Time and Energy Complexity of Distributed Computation in Wireless Sensor Networks*

Nilesh Khude[1], Anurag Kumar[1], Aditya Karnik [2]

Abstract—We consider a scenario where a wireless sensor network is formed by randomly deploying n sensors to measure some spatial function over a field, with the objective of computing the maximum value of the measurements and communicating it to an operator station. We view the problem as one of message passing distributed computation aver a geometric random graph. The network is assumed to he synchronous; at each sampling instant each sensor measures a value, and then the sensors collaboratively compute and deliver the maximum of these values to the operator station. Computation algorithms differ in the messages they need to exchange, and our formulation focuses on the problem of scheduling of the message exchanges. We do not exploit techniques such as source compression, or block coding of the computations.

For this problem, we study the computation time and energy expenditure for one time maximum computation, and also the pipeline throughput. We show that, for an optimal algorithm, the computation time, energy expenditure and the achievable rate of computation scale as $\Theta\left(\sqrt{\frac{n}{\log n}}\right)$, $\Theta(n)$ and $\Theta\left(\frac{1}{\log n}\right)$ asymptotically (in probability) as the number of sensors $n \to \infty$.

We also analyze the performance of three specific computational algorithms, namely, the tree algorithm, multihop transmission, and the ripple algorithm. and obtain scaling laws for the computation time and energy expenditure as $n \to \infty$. Simulation results are provided to show that our analysis indeed captures the correct scaling; the simulations also yield estimates of the constant multipliers in the scaling laws, Our analyses throughout assume a centralized scheduler and hence our results can be viewed as providing hounds for the performance with a distributed scheduler.

Keywords: distributed maximum **computation, scaling laws for sensor networks**

I. Introduction

A wireless sensor network is formed by a set of small untethered sensor devices that are deployed in an ad hoc fashion and cooperate in sensing the environment and in computing some quantity of global interest (for a survey see [1]). Sensor nodes have limited, and in many cases, irreplaceable power sources. Power consumption occurs due to radio transmission, reception, sensing and computing, typically in decreasing order. As a node spends the maximum energy in communication, it is desirable to have local interactions between the sensors to process the data in the network and, hence, reduce transmissions, rather than to transmit the raw data to the base station. In this paper we focus on the distributed computation approach for sensor information processing.

It is assumed that time is slotted and the n sensors are synchronised at slot boundaries. The sensors periodically (at some multiple of the slot time) sample the environment variable. e.g., temperature. At sampling instant k; each sensor measures a value, yielding a vector of values $(v_1(k), v_2(k), \cdots, v_n(k))$. The objective is to collaboratively compute and deliver $\max\{v_1(k), v_2(k), \cdots, v_n(k)\}$ to an operator station, for each such vector of sampled values. See [3] where the need for a distributed maximum computation arises as a part of a distributed self-tuning algorithm for the optimal operation of a sensor network. If the sensors calculate the local maxima while routing the values to the operator station, we can reduce the traffic in the network and thereby increase the network lifetime. In case of the function max, this is possible because the maximum function is insensitive to the order of computation and can be calculated recursively by using partial results obtained by using subsets of the data, i.e., $\max\{a, b, c, d, e\}$ can be calculated as $\max\{\max\{a, b\}, \max\{c, \max\{d, e\}\}\}$. This means that the function max can be expressed as compositions of itself.

We adopt the message passing distributed computing model. The sensors communicate by sending packets to each other and then performing computations based on the received data and the partial results they have. When successive results for several sampled values need to be computed then separate pipelined computations are performed for each vector of sampled values. Thus we do not exploit block computation, as has been done in [2].

The computation algorithms we consider differ in the way the computations are organised, and hence in the message transmissions that are required to carry out the task. For our underlying assumptions, we provide, for the number of nodes, n, becoming large, optimal scaling results (i.e., the best possible time and energy scaling with the number of nodes), and also the performance of some candidate computation algorithms, thus identifying the best among these.

The following is a summary of our contributions in this paper: All our results are of the nature of providing asymptotic scaling laws (that hold in the "in probability" sense) as the number of nodes $n \to \infty$. Assuming that the transmission range $r(n)$ scales as $\Theta(\sqrt{\frac{\log n}{n}})$, we establish that the time required for one computation (e.g., initiated by a query) by an optimal algorithm is $\Theta\left(\sqrt{\frac{n}{\log n}}\right)$. The minimum energy expended in the network during a computation is $\Theta(n)$, and the maximum achievable rate of pipelined computations is $\Theta\left(\frac{1}{\log n}\right)$. All these orders are right bounds in the sense

*This work was supported by the Indo-French Centre for Promotion of Advanced Research (IFCPAR) under research contract No. 2900-IT.

[1]ECE Department. Indian Institute of Science, Bangalore. INDIA
[2]Tata Institute of Fundamental Research. Mumbai. INDIA

that there exist (centralized) algorithms that achieve these orders. We also obtain scaling laws for the computation time and energy expenditure of the tree algorithm, multihop transmission, and the ripple algorithm, and we conclude that among these the tree algorithm is the best in terms of time and energy complexity. All these analyses assume a centralized, collision-free medium access scheduler. Thus these orders can be viewed as lower bounds when some practical distributed medium access protocol is implemented.

The work we report in this paper is closely related to the one presented in [2]. We will discuss the relationships after formally presenting our model in Section II. We will then discuss some background results in Section III. In Section **IV,** we obtain the optimal order expressions for the performance measures. The performance of some algorithms is analyzed in Section **V.** Simulation results are presented in Section **VI. We** conclude the paper in Section **VII.**

II. The Model and performance Measures

We consider n sensors deployed in a circular field. **A** sensor located at the coordinate $\mathbf{X}$ measures the value of some spatial function (say, temperature) $f(\mathbf{X})$. We assume that the measurements are exact. i.e., the errors due to noise or quantization are not considered. We are interested in obtaining the maximum of the measured values **and** communicating the maximum to an operator station located at the centre of the field.

Sensor Network Model: The two dimensional field in which the n sensors are located is denoted by $\mathcal{A}$. The sensor network **is** characterized by an indexed set of sensors locations $\mathcal{S}$; sensor i has location $\mathbf{X}_i$, $\mathbf{X}_i \in \mathcal{A}$, $1 \leq i \leq n$. Thus, the network $\mathcal{S}$ is a random vector $(\mathbf{X}_1, \ldots, \mathbf{X}_n) \in \mathcal{A}^n$ where $\mathbf{X}_i$s are i.i.d. random variables with uniform distribution over $\mathcal{A}$. The random experiment of deploying a network of sensors is characterized by the probability space $\Sigma^n := (\mathcal{A}^n, \mathcal{F}^n, P^n)$ **where** $\mathcal{A}^n$ **is** the **sample** space, $\mathcal{F}^n$ **is** the event space (a Borel **field)** and P^n is the probability measure. We index the whole experiment **by** n, the number of nodes deployed in the field. **As** n increases, **we** get a sequence **of** experiments, We wish to study asymptotics of certain performance measures as $n \to \infty$.

All radio communication is over a common channel **and** any radio transceiver can either transmit or receive **at** a time. The ***rransmission range*** of the sensors is **fixed** for fixed n **and** is denoted **by** $r(n)$. If **any** two sensors are within a distance $r(n)$ of each other then there is a bidirectional link between them. Thus, the neighbours of a node are nodes within a distance $r(n)$ from that node. The form of $r(n)$ we use is motivated by the results of [5] (see Lemma 3.1).

Definition 2.1: Given a network realisation $\mathcal{S}$, the graph $G(\mathcal{S})$ is a random graph formed by the n nodes at the locations defined by $\mathcal{S}$, and links joining the nodes that are separated **by** a distance not greater than $r(n)$. □

Interference Model: Let $|\mathbf{X}_i - \mathbf{X}_j|$ denote the Euclidean distance between the nodes i and j. We adopt the ***protocol** model* which defines the interference constraints as follows.

***Definition** 2.2 (Gupta **and Kumar** [6]): Protocol Model **of** Interference*: When node i lransmits to node j (i.e., $i \to j$), then the transmission is successful **if**

1) $|\mathbf{X}_i - \mathbf{X}_j| \leq r(n)$
2) For every node k that transmits simultaneously, $|\mathbf{X}_k - \mathbf{X}_j| \geq (1 + \Delta) r(n)$ for some **fixed** $\Delta > 0$.

□

Distributed Computation Model: We work with the model of *message **passing*** distributed computation. Nodes explicitly send packets to each other. and do not exploit any extra information available on the wireless medium by way of listening to the other nodes' transmissions or to collisions. Nodes draw inference based on only the packets that are explicitly sent to them. This necessitates that to complete a computation, each node must *influence* the computation. **As** an example, if there are three sensors with values a, b, and c, such that $a > b > c$, the broadcast of the largest value a suffices to complete the computation. However. in the message passing model, unless the operator station hears at least once from all the nodes it is unable to conclude that the computation is complete. Note that this does not imply that the operator station must explicitly receive each value. only that the computation it receives must have been influenced by every sensor's value. Further, the computation of the maximum for each set of measured values is carried out separately, and block computation is not exploited as in [2].

Formally, suppose that the result delivered to the operator station is $y = \max \mathcal{S}$, **and** is obtained as $y = \max\{y_1, y_2, \cdots, y_m\}$, with $y_i = \max \mathcal{S}_i$, where, for $1 \leq i \leq m$, $\mathcal{S}_i \subset \mathcal{S}$. Now even though $y = y_j$ for some j, $1 \leq j \leq m$ (i.e., the maximum is determined by the subset of sensors $\mathcal{S}_j$), we require that every node k, $1 \leq k \leq n$, belongs to some set $\mathcal{S}_i$, $1 \leq i \leq m$, in the final computation. **We** will then say that every node has had influence on the computation. **This** implies that every node must transmit at least once in order to have influence on the result.

Further, we define $\mathcal{N}_j^{(k)}$, the k hop neighborhood of node j as follows. Let $\mathcal{N}$ denote the set of n nodes. $\mathcal{N}_j^{(0)} := \{j\}$; $\mathcal{N}_j^{(1)} := \{i \in N : |\mathbf{X}_i - \mathbf{X}_j| \leq r(n)\}$; $\mathcal{N}_j^{(k)} := \{i \in \mathcal{N} : |\mathbf{X}_i - \mathbf{X}_l| \leq r(n), l \in \mathcal{N}_j^{(k-1)}\}$. We note that from **the beginning of the slot** in which node j **first transmits** its value, **it** takes at least k hops until the computations in the set $\mathcal{N}_j^{(k)} - \mathcal{N}_j^{(k-1)}$ are influenced by the value of node j, i.e., in each slot the influence ***of*** node j can spread by at most one hop.

A *computation **algorithm*** defines the sequence of message passing transactions, between specified transmitter-receiver pairs, that leads to the function being computed and the results delivered to the operator station. **A** computation algorithm may have associated **with** it a subgraph of $G(\mathcal{S})$ such that only the links in this subgraph are activated. **For** example, in a tree algorithm (see Section **V)** a tree subgraph of $G(\mathcal{S})$ is defined and only the links in **this tree** are activated, progressing from the leaves up to the root.

Recently Giridhar **and** Kumar have addressed the problem of distributed computation (or data fusion) in [2]. The functions they consider are symmetric[1]. They prove a $\Omega\left(\frac{1}{\log n}\right)$ lower bound on the rate of pipelined computation of symmetric functions in a random planar network, For an upper **bound** on the rate, the symmetric functions are divided into two subclasses viz. ***type-sensitive functions*** **and** ***ope-threshold***

[1] A function **is said** *to* be **symmetric if** *the* function **value** *is* **insensitive** *to* **the order in which data is processed or** *to* **the identity of node that measures** the data.

functions[2]. For *type-sensitive functions* the upper bound on rate **is** shown to **be** $O\left(\frac{1}{\log n}\right)$ in **a** random planar network; whereas for *type-threshold functions* the upper bound is shown to be **O** (———). In our work, reported here, we have considered the max function and we find the achievable maximum rate to be $\Theta\left(\frac{1}{\log n}\right)$. Thus the upper bound on the achievable computation rate that we obtain is lower. This is because our message passing computation model does not maximally utilize the information available by virtue of the radio being a broadcast medium. and we do not exploit techniques such as block coding across measurements.

Scheduling Assumptions: A synchronised *time slotted* system is assumed. with a packet transmission between any pair of **nodes** requiring one slot. For the purpose of obtaining our scaling results, we assume perfect scheduling of transmissions? i.e.. in every slot certain links are scheduled and these transmissions are guaranteed to succeed. The perfect scheduler has a set of *maximal activation sets*, i.e., a set of uansmitter-receiver pairs which can communicate simultaneously without violating the interference constraints. The activation sets that are scheduled are maximal in the sense that addition of any transmitter-receiver pair in such a set will violate the interference constraints. **Also,** the perfect scheduler **is** assumed to be optimal in the sense that given the node placements and the set of transmissions to be activated, it chooses the activation sets resulting in the minimum number **of** slots. Owing to these assumptions, our scaling laws should be viewed as lower bounds on what is practically achievable.

Computation and Scheduling Interaction: The computation progresses in ***stages,*** each stage requiring the transmission of messages from certain transmitters to designated receivers (including, possibly, broadcast to a set of receivers in the neighbourhood of each transmitter). Given the transmissions to **be** scheduled at each stage, the scheduler provides a deterministic sequence of maximal activation sets that need to be activated in the successive slots in order to compleie this stage of computation. After the completion of a stage in the computation, the computation algorithm defines the next set of transmissions to be scheduled.

For example, in the Tree algorithm (see Section **V),** each stage corresponds to activation of links at one level in the tree. When a stage of computation involves one transmission from every node, we call such a stage **a** *round.* Note that since a computation algorithm requires each node to transmit its measured value at least once, a computation involves at least one round.

Energy Expenditure Model: We consider the following components of energy expenditure per packet transmission and reception. $E_{xmit-radio}$: the transmit energy radiated. Thus $E_{xmit-radio} = \alpha d^{\eta}$, where d is the distance between the transmitter and the receiver, η is the path loss exponent ($2 \leq \eta \leq 4$), and α is the energy corresponding to the received power level at the receiver required for successful reception in the presence of receiver noise (also sometimes called the receiver sensitivity). $E_{xmit-pkt}$: Energy required in the transmitter's electronics to transmit a packet. $E_{receive-pkt}$: Energy required in the receiver's electronics to receive a **packet.** $E_{proc-pkt}$: Energy required by the on-board computer to perform the computational task triggered by a received packet.

Performance Measures: For a given node placement S, a computation algorithm will result in the maximum being computed in some number of slots. We denote the time required to complete the computation by an optimal scheduler by $\Gamma(\mathcal{S})$. Thus. for a given computation algorithm, Γ is **a** random variable over Σ^n which takes a specific value for every realization of $\mathcal{S}$. **Also** the node placement $\mathcal{S}$ and the computation algorithm (along with the centralised schedule) determine the number of transmissions and receptions by each node; and thereby the total energy spent. Let $E(\mathcal{S})$ be the total energy spent in the network while performing one computation and E denote the random variable over Σ^n, akin to Γ.

III. Background Results

The presentation of these results is interspersed **with** remarks about the intuition behind them. In writing these remarks we use the notation $\Theta(\cdot)$ loosely; it only means "of the order:' the "in probability" qualification being implied.

Bounds on the number of hops in the shortest path: Consider a network realization $\mathcal{S}$ **and** consider all the pairs of points in the field ***A*** separated by a distance d; these points need not **be** locations of nodes. If a node at one such point were to communicate with a node at the other point at a disiance d, the packets will be transmitted along a multihop path using some intermediate nodes. The number of hops in the shortest path joining these points **and** using the intermediate nodes **and** links in $G(\mathcal{S})$ will be finite and will depend on the distance d. **We** define $\overline{H}(d)$ ($\underline{H}(d)$, resp.) as the supremum (infimum. resp.) over the number of hops in the shortest paths connecting all such pairs of points. Thus $\overline{H}(d)$ and $\underline{H}(d)$ are random variables over Σ^n defined for the distance *d.* We need probabilistic bounds on $\overline{H}(d)$ and $\underline{H}(d)$ as a function of d and $r(n)$.

Lemma 3.1: For a *circular* field of unit radius: (1) If $r(n) = \sqrt{\frac{\pi p^2}{2\sqrt{p^2-4}}\frac{\log n}{n}}$, with $p > 2$, then $\lim_{n\to\infty} \mathcal{P}^n$ (G is connected) = 1.
(2) In addition

$$\lim_{n\to\infty} \mathcal{P}^n\left(\frac{d}{r(n)} \leq \underline{H}(d) \leq \overline{H}(d) \leq p\frac{d}{r(n)}\right) = 1$$

Remark: This result has the obvious intuition.' The transmission range of a node is $r(n)$. Hence, the number of hops in the shortest path between any two nodes separated **by** a distance d should be $\Theta\left(\frac{d}{r(n)}\right)$.

Proof: The first **part** follows easily from the results in [5]. The proof of the second part is provided in the Appendix. □

Corollary 3.1: In a *square* field of unit area if n nodes are deployed and $r(n) = \sqrt{\frac{p^2}{2\sqrt{p^2-4}}\frac{\log n}{n}}$, with $p > 2$ then the following hold
(1)

$$\lim_{n\to\infty} \mathcal{P}^n(G \text{ is connected }) = 1$$

(2)

$$\lim_{n\to\infty} \mathcal{P}^n\left(\frac{d}{r(n)} \leq \underline{H}(d) \leq \overline{H}(d) \leq p\frac{d}{r(n)}\right) = 1$$

[2]Informally, type-sensitive **functions** are those **which require almost the entire** data **for computation.** e.g., the **mean value** of **the measurements. On the** other **hand** type-threshold **functions can be computed with some part of the measurements,** i.e., the entire **data is not required for computation,** e.g., **an indicator function.**

□

Remark: In this paper, we have chosen $p = 2\sqrt{2}$, which implies $r(n) = \sqrt{\frac{2\pi \log n}{n}}$ for a circular field of unit radius. This choice of form of $r(n)$ is motivated by the necessity to maximize the spatial reuse in the network while retaining the connectivity of the network. □

Bounds on the number of simultaneous transmissions: Corresponding to every sample point S, and given an $r(n)$, we have a set of activation sets. We are interested in upper and lower bounds that uniformly bound the cardinalities of all the activation sets, i.e., these are bounds on the number of simultaneous transmissions in the network. We denote the minimum and maximum of the cardinalities of these activation sets by $\underline{\gamma}(\mathcal{S})$ and $\overline{\gamma}(\mathcal{S})$ respectively. The quantities $\underline{\gamma}$ and $\overline{\gamma}$ also are random variables that take specific values for a given network realization.

Lemma 3.2: With the protocol model of interference, and with perfect scheduling of transmissions, if the transmission range $r(n) = \sqrt{\frac{2\pi \log n}{n}}$, there exist some positive constants a_1 and a_2 such that

$$\lim_{n\to\infty} \mathcal{P}^n \left(a_1 \frac{n}{\log n} \leq \underline{\gamma} \leq \overline{\gamma} \leq a_2 \frac{n}{\log n} \right) = 1$$

Remark: This is what we would intuitively expect since the area of the field ***A*** divided by $\pi(r(n))^2$ is of the order $\frac{n}{\log n}$.

Proof: See the Appendix. □

Bounds on the time required for a round: Recall the notion of round from Section II. Though each node transmits once in **a** round. the time required to complete **a** round will depend on **the** transmitter-receiver combinations. Let $\overline{T}(\mathcal{S})$ ($\underline{T}(\mathcal{S})$, resp) denote the maximum (minimum, resp.) of the lime required to complete a round, with maximum (minimum. resp.) taken over all possible transmitter-receiver combinations. We note that for a **given** network realization $\mathcal{S}$ **and optimal link** scheduling algorithm. the bounds $\overline{T}(\mathcal{S})$ **and** $\underline{T}(\mathcal{S})$ are **fixed. Thus,** $\overline{T}$ **and** $\underline{T}$ are **random variables** over Σ^n.

Lemma 3.3: With the protocol model of interference, with perfect scheduling of transmissions, **and** with the transmission range $r(n) = \sqrt{\frac{2\pi \log n}{n}}$, the **bounds** on the time required to schedule transmission of all the nodes $\overline{T}$ and $\underline{T}$ satisfy the relation

$$\lim_{n\to\infty} \mathcal{P}^n \left(\frac{\pi\Delta^2}{2} \log n \leq \underline{T} \leq \overline{T} \leq 8\pi(1+\mu)(2+\Delta)^2 \log n \right) = 1$$

Proof: See the Appendix. □

Remark: The intuition behind the above **two** results is the following. If we assume that the interference is zero outside the range of the transmitter, i.e., **A** $= 0$, then the number of simultaneous transmissions is equal to the number disjoint disks of radius $r(n)$. Each disk occupies an area of $\Theta(\pi r^2(n))$. Hence the number of simultaneous transmissions should be $\Theta\left(\frac{1}{r^2(n)}\right) = \Theta\left(\frac{n}{\log n}\right)$. This implies that the number of slots required to schedule all the **nodes'** transmissions once is $\Theta(\log n)$. □

Farthest nodes: We will need the following observations.

Lemma 3.4: Consider a square field of unit area. For a given $\epsilon > 0$, probability that the farthest node from the center of the field lies at a distance greater than $\left(\frac{1}{\sqrt{2}} - \epsilon\right)$ goes to 1 as $n \to \infty$. □

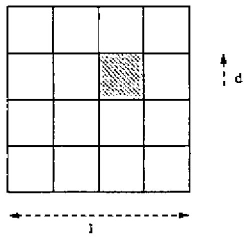

Fig. 1. **Tessellation** to *get* the **bounds** on the number **of** neighbours **of** a node. $d = \sqrt{\frac{K \log n}{n}}$ where $K = 8$.

Lemma **3.5:** In the circular field of unit radius, for a given $\epsilon > 0$, probability that the farthest node from the center of the field lies least at a distance of $1 - \epsilon$ goes to 1 as $n \to \infty$. □

IV. OPTIMAL ORDERS FOR PERFORMANCE MEASURES

In this section, we obtain the optimal order relations for the performance measures such as computation time. energy expenditure and the rate (throughput) of maximum calculation. Initially we shall assume that the field is a square of unit area and obtain a bound on the computation time. We then extend this to a circular field.

For computation time and energy expenditure, we shall first obtain absolute lower bounds in order sense (i.e., **we** establish $\Omega(\cdot)$ relations). These bounds are absolute in the sense that no algorithm can do better than these bounds. We then construct centralized algorithms that achieve the same order **as** that of the lower bounds (but with a different leading constant). This gives an upper bound on computation time and energy expenditure for an optimal algorithm (i.e., $O(\cdot)$ relation). Thus we **obtain** an exact order (i.e., $\Theta(\cdot)$ relation) for an optimal algorithm.

Optimal order for computation time:

Theorem 4.1: If n nodes are uniformly *distributed in* **the** square field of unit **area,** then there exist positive constants u_1 and u_2 such that the number of slots required for an optimal algorithm to calculate the maximum measured value under the assumption of perfect scheduling obeys the following relation

$$\lim_{n\to\infty} \mathcal{P}^n \left(u_1 \sqrt{\frac{n}{\log n}} \leq \Gamma_{opt} \leq u_2 \sqrt{\frac{n}{\log n}} \right) = 1$$

Remark Thus we see that the optimum scaling of computation time Γ_{opt} with n is $\Theta\left(\sqrt{\frac{n}{\log n}}\right)$. □

Proof: **We** note that for the value which we report as the **maximum** value to **indeed be** maximum value, all the nodes' values must be considered while calculating the maximum. Hence, there **should** be at least one node which has received all the values or **the** maximum value computed by this node has the *influence* of all the values on it. Probability **that** the influence of the farthest node's value has to propagate a distance $\left(\frac{1}{\sqrt{2}} - \epsilon\right)$ goes to 1 as $n \to \infty$ for all $\epsilon > 0$ (Lemma 3.4). **As** seen before the influence of a node's value can propagate at most one hop in a slot. Hence, the computation time has to be at least the time taken **by** the value of the farthest node to influence the result received by the center. This gives the lower bound on the computation time that $\Gamma_{opt} \geq \underline{H}(d_{max})$. From Lemma **3.1,** for $\epsilon > 0$

$$\lim_{n\to\infty} \mathcal{P}^n \left(\Gamma_{opt} \geq \left(\frac{1}{\sqrt{2}} - \epsilon\right) \frac{1}{r(n)} \right) = 1 \qquad (1)$$

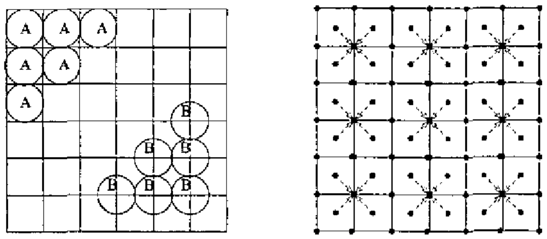

Fig. 2. The construction for an upper bound on the computation time.

(Here the transmission range $r(n)$ should be considered for the square field as per Corollary 3.1).

The upper bound on the computation time can be obtained by giving an actual computation algorithm. We assume A = 0 in obtaining the upper bound on computation time. This assumption does not affect the order of the computation time. (See Remarks 4.1.) Here we consider a centralized algorithm and obtain its computation time. The computation time of an optimal algorithm will be less than this time.

We divide the field into the square cells of size $\sqrt{\frac{K\log n}{n}}$ where $K = 8$; see Figure 1. The number of cells in the field $M_n = \frac{n}{K\log n}$. We need an upper hound on the number of nodes falling in a cell. Xue and Kumar in [7] have given a bound (see Theorem 8.1 in the Appendix) that for $K > \frac{1}{\log\frac{4}{e}}$, if a square field of unit area is tessellated into square cells by a grid of size $\sqrt{\frac{K\log n}{n}}$, where n is the number of nodes uniformly deployed in the field then the probability that the number of nodes falling in a cell is bounded by $(1-\mu)K\log n$ and $(1+\mu)K\log n$ goes 1 as $n\to\infty$ (here $\mu\in(0,1)$ satisfies some conditions). We note that the average number of nodes in a cell is $K\log n$ and μ captures the variation. Since in our case, $K = 8 > \frac{1}{\log\frac{4}{e}}$, the result applies. Hence, with probability 1. the number of nodes in a cell is bounded by $(1+\mu)K\log n$.

Now, we draw circles as shown in the left part of Figure 2. Type A and B circles together cover the whole area. Also,

$$\begin{aligned}&\mathcal{P}^n(\text{Number of nodes in any circle} \le (1+\mu)K\log n)\\ \ge\ &\mathcal{P}^n(\text{Number of nodes in any cell} \le (1+\mu)K\log n)\\ =\ &1 \text{ as } n\to\infty\end{aligned}$$

Hence,

$$\lim_{n\to\infty}\mathcal{P}^n(\text{Number of nodes in any circle} \le (1+\mu)K\log n) = 1$$

We elect the node which is nearest to the center of the circle as the cluster-head. All nodes in type A circles form clusters and the nodes in type B circles that do not lie in the type A circles form clusters. Type B cluster-heads actually cover smaller area.

It can be shown similar to the Lemma 3.4 that the probability that the cluster-head lies in the circle of radius ϵ at the center of the circle goes to 1 as $n\to\infty$ for all values of t and hence for the case when $\epsilon\to 0$.

Hence, all nodes within a circle can reach their cluster-heads in one hop. Since we have assumed zero interference outside the range (*See* Remarks 4.1), non overlapping circles can have simultaneous transmissions. This means all type A clusters can have one transmission in each cluster simultaneously. After type A cluster-heads have received the values from the cluster members (which will take at most $(1+\mu)K\log n$ slots). the same procedure can be repeated for the type B clusters. Hence. the cluster-heads will get all the values in their clusters in less than $2(1+\mu)K\log n$. considering both type A and B clusters.

Now, we have to consider only the cluster-heads. The type A cluster-heads will report their values to the type B cluster-heads as shown in the right side of Figure 2. Each cluster-head has to send the value to a node which is at a distance $\sqrt{2}r(n)$ apart. Hence, the path will have ≤ 4 hops high probability as $n\to\infty$ (Lemma 3.1). Thus it wilt require at most 16 slots. Note that all type B cluster-heads are $2r(n)$ apart. Hence there is no interference and all the assigned type B cluster-heads can receive simultaneously.

Now only type B cluster-heads are remaining. They are aligned in the straight lines. The type B cluster-heads which are near the left and right edge transmit their values towards the central part (of the line) horizontally. As the values reach other cluster-heads in the path, the new maximum value is propagated ahead. These transmissions can occur simultaneously as the paths are confined in the rectangles (Lemma 3.1), and the minimum distance between any two rectangles is $2r(n) - 2\frac{r(n)}{2\sqrt{2}} > r(n)$. Hence there is no interference.

Since the range is $r(n)$ and the values have to propagate a distance 1/2 units, the probability that the time required for the propagation $\le \frac{2\sqrt{2}}{2r(n)}$ slots approaches 1 as $n\to\infty$.

Once these values merge on the central vertical line, the same procedure can be used to get all the values at the center. Considering that the probability of occurrence of all the events approaches 1 as $n\to\infty$ we can say that

$$\lim_{n\to\infty}\mathcal{P}^n\left(\Gamma_{opt} \le 2(1+\mu)K\log n + 16 + \frac{\sqrt{2}}{r(n)} + \frac{\sqrt{2}}{r(n)}\right) = 1$$

$$\lim_{n\to\infty}\mathcal{P}^n\left(\Gamma_{opt} \le 16(1+\mu)\log n + 16 + 2\sqrt{\frac{n}{\log n}}\right) = 1 \qquad (2)$$

From Equations 1 and 2,

$$\begin{aligned}\lim_{n\to\infty}\mathcal{P}^n\Bigg(\left(\frac{1}{\sqrt{2}}-\epsilon\right)\sqrt{\frac{n}{2\log n}} &\le \Gamma_{opt}\\ &\le 16(1+\mu)\log n + 16 + 2\sqrt{\frac{n}{\log n}}\Bigg) = 1\end{aligned}$$

□

Remarks 4.1:
1) The assumption of A = 0 is to simplify the presentation, and does not change the order of the computation time. If A > 0, then the transmissions in a Type A cluster will interfere with some constant number of other type A clusters and this constant depends only on A. Thus, in this case. all the type A clusters can be activated in a constant number of slots (which depends only on A). rather than in 1 slot as assumed in the proof. Thus, we obtain the same scaling order even if $\Delta > 0$.
2) Intuition about the logn term in the upper bound expression can be obtained as follows. Since we form clusters of nodes that are one hop neighbours of the cluster-heads, the number of cluster members is the number of nodes that lie in a circle of radius $r(n)$ drawn around the cluster-head. The node density is $\frac{n}{\pi}$. Hence, the number of cluster members would be $\Theta(\pi r^2(n)\frac{n}{\pi}) = \Theta(\log n)$. Only one node in a cluster

can transmit in a slot; hence $\Theta(\log n)$ slots are needed to complete the collection of values from a cluster. The clusterheads need to transmit the values to the operator station. Since the clusterheads are sparsely distributed. simultaneous transmissions are possible **and** value of the farthest clusterhead needs to travel unit distance which requires $\Theta(\frac{1}{r(n)})$ slots.

3) The above result can be very easily extended to the circular field. We skip the proof and state the result as a corollary. □

Corollary 4.1: If n nodes are uniformly distributed in a circular field of unit radius, then there exist positive constants v_1 and v_2 such that the number of slots required for an opumal algorithm to calculate the maximum measured value under the assumption of perfect scheduling obeys the following relation

$$\lim_{n\to\infty} \mathcal{P}^n \left(v_1\sqrt{\frac{n}{\log n}} \le \Gamma_{opt} \le v_2\sqrt{\frac{n}{\log n}} \right)$$

□

Optimal Order for Energy Expenditure:

Theorem 4.2: If n **nodes** are uniformly distributed in a circular field of unit radius, then the total energy expenditure in the network by an optimal algorithm to calculate the maximum measured value, under the assumption of perfect scheduling, is $\Theta(n)$.

Proof: In any algorithm, every node **has** to transmit at least once **and** at least to its one hop neighbour. Similarly, every transmission must **be** received. The number of comparisons required can vary depending on how many values we report to the operator station which will be compared at the operator station. Hence, we get a lower bound as

$$E_{opt} \ge n(E_{xmit-radio} + E_{xmit-pkt} + E_{receive-pkt})$$

We shall see that the tree algorithm (see Section V) has energy expenditure

$$E_{Tree} = n(E_{xmit-radio} + E_{xmit-pkt} + E_{receive-pkt} + E_{proc-pkt})$$

An optimal algorithm will have energy expenditure at most equal to the free algorithm. Hence,

$$E_{opt} \le n(E_{xmit-radio} + E_{xmit-pkt} + E_{receive-pkt} + E_{proc-pkt})$$

Hence. $E = \Theta(n)$ for all n. □

Optimal Order for the Achievable Pipelined Throughput: The network normally will perform the computations continuously. **This** can be viewed as **a** complex queueing system in which a batch of measurements arrives at sampling instants in the sampling buffers of the nodes with the arrival rate of the batches being equal to the sampling rate. The batch leaves the system when corresponding maximum computation **is** over. The computations are pipelined in the network. It **is** of interest to obtain the saturation throughput of this system, which **will** dictate the permissible sampling rate of the sensors. That is. our interest **is** to characterize the interdeparture time of **the** batches of the measurements when the nodes are infinite. We denote this inter-departure time (also called as pipelined computation time) **by** $\Gamma_{pipeline}(\mathcal{S})$. $\Gamma_{pipeline}$ is a random variable over Σ^n.

We derive the result for the square field of unit area [with the transmission range **modified** as per Corollary 3.1) **and** then the result for the circular field ***A* follows.**

Theorem 4.3: **If** n nodes are uniformly distributed in a square **field** of unit area, then there exist positive constants w_1 **and** w_2 such that the number of slots required for an optimal algorithm to continuously compute the maximum measured value under the assumption of perfect scheduling obeys the following relation

$$\lim_{n\to\infty} \mathcal{P}^n \left(w_1 \log n < \Gamma_{pipeline} < w_2 \log n \right) = 1$$

Proof: In the message passing paradigm (see II), each node has to uansmit at least once to complete a computation. From **Lemma** 3.1, **we** know that the time required to schedule a round ***in*** a *square field* is lower bounded as

$$T > \frac{\Delta^2}{2}\log n$$

This gives the following lower bound on pipelined computation time $\Gamma_{pipeline}$

$$\mathcal{P}^n \left(\Gamma_{pipeline} > \frac{\Delta^2}{2} \log n \right) = 1 \text{ for all } n$$

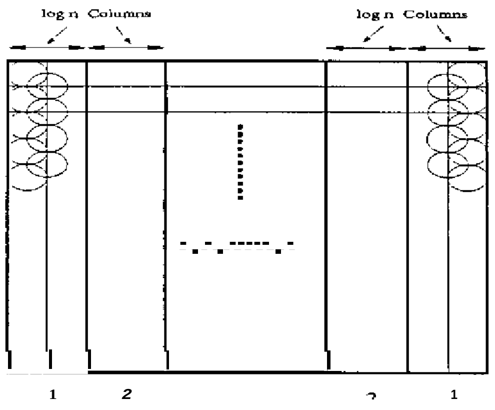

Fig. 3. Transmission scheme for an upper bound on the pipelined computation time. First the computation is done in the leftmost and rightmost logn columns (marked with 1), and then in the next set of logn columns (marked with 2), and so on.

An upper bound on the pipelined computation time required for an optimal algorithm can be obtained by actually constructing a computation scheme. Here we follow the scheme used to obtain the upper bound on the computation time, where the **field** is tessellated in the small cells of size $\sqrt{\frac{K\log n}{n}}$ with $K = 8$ and the nodes are divided into type **A** and B clusters. The transmission schedule is as follows. First, the nodes in the rightmost and leftmost logn columns of type **A** clusters transmit their values to their respective cluster **heads** (these columns are **marked** as 1 in the Figure **3).** Different type **A** clusters can have simultaneous transmissions. Since there is no spatial reuse within a cluster **and** the number of cluster members is bounded , this will take at most $(1+\mu)K$ logn slots with high probability as $n \to \infty$ (see [7]).

Next. these type **A** cluster heads transmit the effective maximum values to the nearest type **B** cluster heads that are towards the central region. Each value has to travel $\sqrt{2}r(n)$ units which require at most 4 slots. Since. the adjacent type **A** clusterheads cannot transmit simultaneously, this requires at most $8(=4+4)$ slots. In the next $(1+\mu)K$ **log** n slots these $\log n$ columns of type B cluster-heads collect the values from their cluster members and compute the effective maximum value.

These maximum values are propagated horizontally to the next type B cluster heads located at a distance of $2\log n r(n)$ from the farthest of the logn type B cluster-heads. This requires $< 4\sqrt{2}\log n$ slots with high probability as $n \to \infty$. (Lemma 3.1).

Thus after $2(1+\mu)K\log n + 8 + 4\sqrt{2}\log n$ slots. the first measurements in the leftmost and rightmost logn columns have been transported outside the leftmost and rightmost columns and now are nearer to the central region. Now the logn type A clusters that are adjacent to the first batch of type A clusters (marked as 2 in Figure 3), can collect their first measurements whereas the rightmost and leftmost $\log n$ type A cluster-heads (marked as 1 in Figure 3) can collect the second measurements.

Thus, in steady state, when the leftmost logn columns are computing with their m^{th} measurements, the next set of $\log n$ columns to the right are computing with their $(m-1)^{th}$ measurements and so on. These measurements, propagating horizontally with the time lag of $2(1+\mu)K\log n + 8 + 4\sqrt{2}\log n$ slots are collected at type B cluster-heads located along the central vertical line. These values should propagate vertically towards the operator station. However, due to interference constraints the transmissions along the vertical and horizontal directions are not possible simultaneously. Hence additional $4\sqrt{2}\log n$ slots are required for this vertical propagation of length $2r(n)\log n$. Thus the results of the successive batches of the measurements arrive at the operator station with the lag of at most $2(1+\mu)K\log n + 8 + 8\sqrt{2}\log n$. Thus

$$\lim_{n\to\infty} \mathcal{P}^n \left(\Gamma_{pipeline} < 2(1+\mu)K\log n + 8 + 8\sqrt{2}\right) = 1$$

Defining K' appropriately, we have

$$\lim_{n\to\infty} \mathcal{P}^n \left(\Gamma_{pipeline} < K'\log n\right) = 1$$

This completes the proof. □

Remarks 4.2: 1) In at most $K'\log n$ slots, the measurements have propagated by a distance of $2r(n)\log n$. The total distance to travel is 1 unit ($\frac{1}{2}$ units in horizontal and vertical directions respectively): This requires at most $K'\log n \frac{1}{2r(n)\log n} = \frac{K'}{2r(n)} = K''\sqrt{\frac{n}{\log n}}$ slots. Thus, the sojourn time of a batch in this algorithm also is of the order $\frac{1}{r(n)}$, same as that of the computation time.

2) A very similar result can be established for the circular field also. This result means that the throughput of the pipelined computations scales as $\Theta\left(\frac{1}{\log n}\right)$. We note that this is also the round rate for scheduling n nodes.

3) This result can also be independently obtained as a consequence of Theorem 2 in [2]. □

V. Performance of Specific Protocols with Perfect Scheduling

We analyze the performance of some computation algorithms in this section. Let the distance of a node from the centre of the field be denoted by a random variable D. In the accompanying analysis we will need the probability density function $f(\cdot)$ of D. For a circular field of unit radius, this is easily seen to be $f(s) = 2s, 0 \leq s \leq 1$ (To see this, note that $F(s) = \frac{\pi s^2}{\pi . 1^2}$ and hence $f(s)ds = \frac{2\pi s ds}{\pi} = 2sds$).

In this section, we will provide the scaling results and the intuition behind them. The detailed proofs are omitted, and can be found in [4].

Tree Algorithm: The sensors form a spanning tree with the operator station as the root of the tree. Here the children of a sensor are amongst its one hop neighbours. Each sensor gets values from its children, compares with its own value and forwards only the maximum value to its parent. So. for each maximum computation each sensor transmits only once. The slowest computation will be over a tree that is a string. For fast computation, we need a shallow tree. Hence, we take all the neighbours of the operator station as its children.

We also note that the nodes with different parents are not assured to have simultaneous transmissions. Simultaneous transmissions occur only if the nodes have different parents and the interference constraints are met.

The following result provides the asymptotic order for the computation time and energy expenditure for maximum computation over a tree. The number of hops required for the farthest node to reach the centre is nothing but the depth of the tree. From Lemma 3.1, we know upper and lower bounds on the number of hops. We analyse Γ_{Tree} in both the cases which combined together provide the displayed result.

Proposition 5.1: For the tree algorithm. if n sensors are distributed uniformly over a circular field of unit radius, then there exist positive constants a_1, b_1 and b_2 such that the energy expenditure E and the computation time Γ required to compute the maximum of the measured values satisfy the relations.

$$E_{Tree} = a_1 n \quad \text{as} \, n \to \infty$$
$$\lim_{n\to\infty} \mathcal{P}^n \left(b_1\sqrt{n\log n} \leq \Gamma_{Tree} \leq b_2\sqrt{n\log n}\right) = 1$$

□

Remark: The following is the intuition for the above result. In the tree algorithm every node transmits only once. There are n transmissions, n receptions and n comparisons. Hence the energy expenditure is $\Theta(n)$. For computation time we know that the number of hops between the farthest node and the operator station is of the order $\frac{1}{r(n)}$. It can be shown ([4]) that the time required to schedule the nodes at a level is of the order logn. Hence the computation time is of the order $\frac{1}{r(n)}\log n$, i.e. $\sqrt{n\log n}$. This is because the nodes at a level cannot be scheduled before all the descendant nodes of this level are scheduled.

Multi-Hop Transmission: n this computation algorithm the value of each node is transported via multihop transmissions to the operator station. No computations are performed at the intermediate nodes. Each transmission follows a shortest path from a node to the operator station. Thus, this computation algorithm also basically involves a tree rooted at the operator station.

We recall that the transmission range $r(n) = \sqrt{\frac{2\pi\log n}{n}}$. Let $H(d)$ denote the random variable denoting the number of hops in the shortest path from a node at a distance d from the centre to the operator station. From Lemma 3.1, if a node is at a distance s from the center of the field, then

$$\lim_{n\to\infty} \mathcal{P}^n \left(\frac{s}{r(n)} \leq \underline{H}(s) \leq \overline{H}(s) \leq \frac{2\sqrt{2}s}{r(n)}\right) = 1$$

where $\underline{H}(s)$ *and* $\overline{H}(s)$ are lower and upper bounds the number of hops in the shortest path from that node to the operator station. We define

$$E_{hop} = E_{xmit-pkt} + E_{xmit-radio} + E_{receive-pkt}$$

where, $E_{xmit-radio} = \alpha\left(\sqrt{\frac{2\pi \log n}{n}}\right)^{\eta}$

The average energy spent in the node's transmission to the operator station is calculated as follows

$$\mathsf{E}(H(D)\, E_{hop}) \quad = \quad E_{hop}\int_0^1 \mathsf{E}(H(s))\, 2s ds \qquad (3)$$

From the definition of $\overline{H}(s)$ and $\underline{H}(s)$, it follows that

$$\mathsf{E}(\underline{H}(s)) \le \mathsf{E}(H(s)) \le \mathsf{E}(\overline{H}(s))$$

Consider

$$\begin{aligned}\mathsf{E}(\underline{H}(s)) &\ge \mathsf{E}\left(\underline{H}(s).I_{\{\underline{H}(s)\ge\frac{s}{r(n)}\}}\right)\\ &\ge \frac{s}{r(n)}.\mathcal{P}^n\left(\underline{H}(s)\ge\frac{s}{r(n)}\right)\end{aligned}$$

where $I_{\{.\}}$ is and indicator function. Now consider

$$\begin{aligned}\mathsf{E}(\overline{H}(s)) &= \mathsf{E}\left(\overline{H}(s).I_{\{\overline{H}(s)\le\frac{2\sqrt{2}s}{r(n)}\}}\right)+\mathsf{E}\left(\overline{H}(s).I_{\{\overline{H}(s)>\frac{2\sqrt{2}s}{r(n)}\}}\right)\\ &\le \frac{2\sqrt{2}s}{r(n)}\mathcal{P}^n\left(\overline{H}(s)\le\frac{2\sqrt{2}s}{r(n)}\right)+n\mathcal{P}^n\left(\overline{H}(s)>\frac{2\sqrt{2}s}{r(n)}\right)\end{aligned}$$

The above equation uses the fact that the maximum number of hops in any shortest path cannot be more than n, the number of nodes. It has been shown in [4] that $\mathcal{P}^n\left(\overline{H}(s) > \frac{2\sqrt{2}s}{r(n)}\right) = O\left(\frac{1}{n}\right)$. Hence

$$\frac{s}{r(n)}\mathcal{P}^n\left(\underline{H}(s)\ge\frac{s}{r(n)}\right)\le\mathsf{E}(H(s))\le\frac{2\sqrt{2}s}{r(n)}\mathcal{P}^n\left(\overline{H}(s)\le\frac{2\sqrt{2}s}{r(n)}\right)+x_n$$

where $x_n = O(1)$. Hence as $n\to\infty$, $\mathsf{E}(H(s)) = \Theta\left(\frac{s}{r(n)}\right)$. Thus substituting in Equation 3 and simplifying, we get $\mathsf{E}(E) = \Theta\left(\frac{1}{r(n)}\right)$. This gives the total energy, as $n\to\infty$, as

$$\mathsf{E}(E_{Multi-Hop}) = n\mathsf{E}(E) = \Theta\left(\frac{n}{r(n)}\right) \quad = \quad \Theta\left(n\sqrt{\frac{n}{\log n}}\right)$$

We now obtain bounds on the computation time $\Gamma_{Multi-Hop}$. Each sensor sends its value to the operator station via the shortest path. These paths can be found by constructing a breadth first tree with operator station as root. The transmissions can then be viewed in the form of tree as in the tree algorithm but with the difference that the values do not merge. Computation is complete when all the n measurements are received at the operator station. Since, the operator station can receive at most one packet in a slot, it will take at least n slots to complete the computation. Thus, we get the obvious lower bound on the computation time as

$$\Gamma_{Multi-Hop} \ge n \quad \text{for all } n$$

We now obtain an upper bound. The computation algorithm progresses in stages. In each stage. all nodes that have packets to send to the operator station transmit one packet to their parents in the tree. Thus in the first stage. all the nodes transmit. Let the number of neighbours of the operator station be n_0. (n_0 is a random variable defined over Σ^n that takes specific value for each network realisation.) Then in the first round the operator station receives n_0 new values; these are the values of its neighbours. In the next stage, the leaf nodes have no packets, but the next level of nodes will have one or more packets. During this stage again, the operator station receives n_0 new values. Then the number of stages are $\frac{n}{n_0}$ and the number of slots required for each stage is bounded by the number of slots required for a full round which is provided by Lemma 3.3. Note that all the nodes do not transmit in all the stages.

It can be shown that the number of neighbours of the operator station is of the order $logn$. More precisely it can be shown that ([4])

$$\lim_{n\to\infty}\mathcal{P}^n(\pi\log n < n_0 < 3\pi\log n) = 1 \qquad (4)$$

Thus using Lemma 3.3 and Equation 4, we get bounds on the computation time as

$$\lim_{n\to\infty}\mathcal{P}^n\left(\frac{\Delta^2}{6}n < \Gamma_{Multi-Hop} < 8(1+\mu)(2+\Delta)n\right) = 1$$

We summarize the above results as follows.

Proposition 5.2: For multi-hop transmission, as $n\to\infty$

$$\mathsf{E}(E_{Multi-Hop}) \quad = \quad \Theta\left(n\sqrt{\frac{n}{\log n}}\right)$$

and there exists a positive constant d_1 such that the computation time $\Gamma_{Multi-Hop}$ required to compute the maximum satisfies the relation.

$$\lim_{n\to\infty}\mathcal{P}^n\left(n \le \Gamma_{Multi-Hop} \le d_1 n\right) = 1$$

□

Remark: Here is the main intuition behind the arguments leading to the result. We know that the time required to schedule the transmissions in an area is of the order $\frac{n}{\frac{1}{r^2(n)}}$, i.e., logn. After each round, the operator station receives one value from each of its one hop neighbours, the number of which is of the order $nr^2(n)$, i.e., logn. Hence the number of rounds required is of the order $\frac{n}{\log n}$. This shows that the computation time is of the order n. For, energy expenditure, we notice that each node induces on an average $\Theta\left(\frac{1}{r(n)}\right)$ transmissions. Each transmission requires fixed energy. Hence, the energy expenditure is of the order $\frac{n}{r(n)}$, i.e., $n\sqrt{\frac{n}{\log n}}$. □

Ripple Algorithm: In this algorithm, sensors keep on exchanging their current estimates of the maximum values and eventually all the sensors know the maximum value. The transmissions take place in stages that are rounds as defined in Section II. In a round. every node transmits its current estimate of the maximum value, receives the estimates from its neighbours and then updates its own estimate of the maximum value at the end of the round. We note that the values of all the sensors propagates one hop distance in every round, like a ripple. Hence, once the influence of the value of the farthest node reaches the centre, the computation is over. We need not continue till all the nodes know the actual maximum. The probability that the farthest node has a distance of $(1-\epsilon)$ unit, for a given $\epsilon > 0$, from the centre approaches 1 as $n \rightarrow \infty$ (Lemma 3.5). In Lemma 3.3. we have obtained bounds on the number of slots required to schedule a round. Since any node's value propagates by one hop in a round. the number of rounds required to complete the computation is calculated using Lemma 3.1,

$$\lim_{n\rightarrow\infty} \mathcal{P}^n \left(\frac{(1-\epsilon)}{\sqrt{2\pi}} \sqrt{\frac{n}{\log n}} \leq \text{number of rounds} \right. \quad (5)$$

$$\left. \leq \frac{2}{\sqrt{\pi}} \sqrt{\frac{n}{\log n}} \right) = 1$$

This combined with Lemma *3.3* gives bounds on the computation time

$$\lim_{n\rightarrow\infty} \mathcal{P}^n \left(\frac{\sqrt{\pi}\Delta^2}{2\sqrt{2}} \sqrt{n \log n} \leq \Gamma_{Ripple} \right.$$

$$\left. \leq 16\sqrt{\pi}(1+\mu)(2+\Delta/2)\sqrt{n\log n} \right) = 1$$

To calculate the energy expenditure, we need to know bounds on the number of neighbours. For this we use the result from [7] to bound the number of neighbours.

In each round, every node broadcasts its current maximum value and receives its neighbours' values. It calculates the new maximum value and broadcasts it in the next round, i.e., only after it has got values from all neighbours. So, in a round each node has one transmission and receives and compares its neighbours' values to compute the current maximum. To find the maximum of m values, we need $(m-1)$ computations. If N_i is the number of neighbours of node i, then node i compares $(N_i + 1)$ values (neighbours' values and its own value). Hence, node i does N_i computations in each round.

The energy spent by node i in a round,

$$E_i = E_{xmit-radio} + E_{xmit-pkt} + N_i.E_{receive-pkt} + N_i.E_{proc-pkt}$$

where, $E_{xmit-radio} = a\left(\sqrt{\frac{2\pi\log n}{n}}\right)^{\eta}$. Let $E_{xmitter} := E_{xmit-radio} + E_{xmit-pkt}$ and $E_{receiver} := E_{receive-pkt} + E_{proc-pkt}$.

From the bound on the number of neighbours from [7] (see Theorem 8.1 in Appendix), we obtain the following result for E_i

$$\lim_{n\rightarrow\infty} \mathcal{P}^n \left(E_{xmitter} + (1-\mu)2\pi^2 \log n (E_{receiver}) \leq \right.$$

$$\left. E_i \leq E_{xmitter} + (1+\mu)2\pi^2 \log n (E_{receiver}) \right) = 1$$

where μ satisfies the condition given in [7]. This equation combined with the Equation 6 gives the bounds on E_i for any i which in turn gives the bound on $E(\mathcal{S})$

$$\lim_{n\rightarrow\infty} \mathcal{P}^n \left(\frac{(1-\epsilon)}{\sqrt{2\pi}} n \sqrt{\frac{n}{\log n}} \left(E_{xmitter} + (1-\mu)2\pi^2 \log n.(E_{receiver})\right) \leq \right.$$

$$\left. E_{Ripple} \leq \frac{2}{\sqrt{\pi}} n \sqrt{\frac{n}{\log n}} \left(E_{xmitter} + (1+\mu)2\pi^2 \log n.(E_{receiver})\right) \right) = 1$$

We summarize the above results as a proposition.

Algorithm	Energy Expenditure	Computation Time
Optimum Algorithms	$\Theta(n)$	$\Theta\left(\sqrt{\frac{n}{\log n}}\right)$
Multi-hop transmission	$\Theta\left(n\sqrt{\frac{n}{\log n}}\right)$	$\Theta(n)$
Tree Algorithm	$\Theta(n)$	$\Theta(\sqrt{n}$ **low** $n)$
Ripple Algorithm	$\Theta(n\sqrt{n} \log n)$	$\Theta(\sqrt{n} \log n)$

Proposition 5.3: For the ripple algorithm. if n sensors are distributed uniformly over a field of unit radius. then there exist positive constants k_1, k_2, l_1 and l_2 such that the energy expendihire $E(\mathcal{S})$ and the computation time $\Gamma(\mathcal{S})$ required to compute the maximum satisfy the relation.

$$\lim_{n\rightarrow\infty} \mathcal{P}^n \left(k_1 n\sqrt{n\log n} \leq E_{Ripple} \leq k_2 n\sqrt{n\log n} \right) = 1$$

$$\lim_{n\rightarrow\infty} \mathcal{P}^n \left(l_1\sqrt{n\log n} \leq \Gamma_{Ripple} \leq l_2\sqrt{n\log n} \right) = 1$$

□

Remark: The following is the intuition behind the arguments leading to the above result. We know that the number of slots required to schedule the transmissions in an area is of the order $\frac{n}{r^2(n)}$, i.e., logn. After each **round,** the values can influence the estimates of nodes at one hop. The number of hops between the farthest node and the operator station are of the order $\frac{1}{r(n)}$. Hence the rounds required are of the order $\frac{1}{r(n)}$. This shows that the computation time **is** of *the* order $\frac{1}{r(n)} \log n$, i.e. $\sqrt{n\log n}$. For energy expenditure, we notice that for each node in each round ihere is one transmission **and** $\Theta(\log n)$ receptions. **As** $n \rightarrow \infty$, the energy spent in reception is dominant over that of radio transmissions. Hence, the energy expenditure of the network is of the order $n \log n \frac{1}{r(n)}$ i.e., $n\sqrt{n\log n}$.

VI. Simulation Results

Table I summarizes the order expressions obtained in the previous section for all the algorithms. However, the constants multiplying these expressions are not known. In this section we validate these order results from a simulation, and as a by-product also obtain estimates **of** the constants.

The simulations are conducted as follows.

1) For the tree algorithm. we build a breadth first **tree** rooted at the operator station. To calculate the computation time, we schedule the nodes at the same levels **by** building activation sets. **We** use the protocol model for interference. The schedule is suboptimal as we ensure that the transmissions are successful irrespective of the location of the receivers, i.e., all the transmitters are at least $(2+\Delta)r(n)$ distance **apart** from any other transmitter. We start from the leaf level **and** go on scheduling the nodes up the tree. This gives the computation time. Since. each node has only one

fixed energy transmission, the energy can be readily calculated.

2) For multi-hop transmissions, the breadth first tree used in the tree algorithm gives the shortest path to the root for each node. We note that all nodes do not always have packets to transmit. Hence. while scheduling, we consider only those nodes which have packets to send. The transmissions are carried on until all the n values reach the operator station. This gives the computation time. Each transmission requires fixed energy. Hence the total number of transmissions give the total energy expenditure.
3) In the ripple algorithm, we have rounds in which each node transmits once. We schedule all the nodes using activation sets as before. In a round, each node has one transmission and as many receptions as the number of its neighbours. This gives the round time and energy per round which are the same for all rounds. Since the number of rounds required is the number of hops required for the farthest node to reach the operator station. the depth of the tree is the number of rounds required. This gives the total computation time and energy.

The simulation plots are obtained by taking an average over ten realizations of the node locations. The parameters used for the simulations are as follows: $E_{xmit-pkt} = 0.25$, $E_{receive-pkt} = 0.5$, $E_{proc-pkt} = 0.00001$, $A = 0.5$, $E_{xmit-radio} = 100r^3(n)$ and $r(n) = \sqrt{\frac{2\pi \log n}{n}}$. The energy values can be viewed as scaled versions of practical values.

Suppose $G(n)$ is the value of some performance measure obtained from the simulation as described above and $f(n)$ is the scaling law we obtain (e.g., $f(n) = \sqrt{n \log n}$ for the computation time for the tree algorithm), then we plot $\frac{G(n)}{f(n)}$. The theoretical results suggest the existence of constants $\underline{a}$ and $\overline{a}$ and functions $l(n) = o(f(n))$ and $u(n) = o(f(n))$ such that

$$\lim_{n\to\infty} \mathcal{P}^n \left(\underline{a}f(n) + l(n) \le G(n) \le \overline{a}f(n) + u(n)\right) = 1$$

$$\lim_{n\to\infty} \mathcal{P}^n \left(\underline{a} + \frac{l(n)}{f(n)} \le \frac{G(n)}{f(n)} \le \overline{a} + \frac{u(n)}{f(n)}\right) = 1$$

$$\lim_{n\to\infty} \mathcal{P}^n \left(\underline{a} \le \frac{G(n)}{f(n)} \le \overline{a}\right) = 1$$

Thus for large n, the values $\frac{G(n)}{f(n)}$ should be confined between the two limits $\underline{a}$ and $\overline{a}$.

In Figures 4, 5 and 6 we plot the ratios of the observed computation times and the asymptotic orders for the three algorithms. The plots show that in each case the interval within which these ratios should lie appears to collapse to a constant, an estimate of which is shown by the flat lines. The computation time has converged in all the cases. The energy curves in case of Tree and Multi-Hop algorithms have not converged because for any finite n the $E_{xmit-radio}$ is finite. Since $E_{xmit-radio}$ decays with n, we observe that the energy curves are decaying. The energy curve in case of Ripple algorithm has converged because unlike the other two algorithm. Ripple has broadcast transmissions. Hence the total energy spent in reception dominates over $E_{xmit-radio}$. Of course. these observations also help to corroborate our scaling results. In Table II we display the order results along with the constant multipliers estimated from the simulation.

Figure 7 compares time and energy requirements of all the three algorithms. From these figures, it is clear that a tree is the preferable way to arrange the computations in the case of computing the maximum.

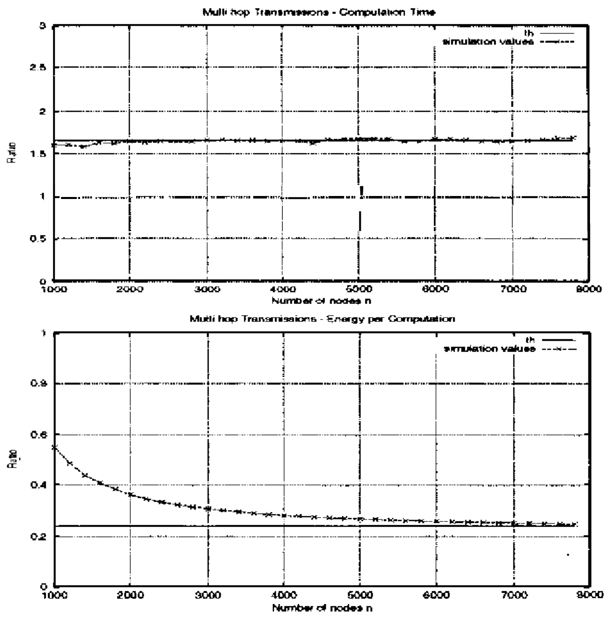

Fig. 4. Multihop transmission: (upper panel) Ratio of observed computation time to the corresponding asymptotic order, and (lower panel) the ratio of the observed energy expenditure per computation and the corresponding asymptotic order. The flat lines show the estimate of the constant multiplying the asymptotic order.

VII. Conclusions

We have addressed the issue of scheduling the processing of information, i.e., the sequence of message passing and computations, in sensor networks. We considered the function max, and obtained optimal order (in probability) expressions as functions of n, the number of nodes, for certain performance measures. namely, computation time, energy expenditure and the rate of computation. These orders provide measures to calibrate the performance of any specific algorithm. We then analyzed some specific computational algorithms and obtained scaling laws for the computation time and energy expenditure with these algorithms. We saw that the tree algorithm arranges the computations most efficiently in terms of computation delay and energy expenditure.

In the present work we have assumed a scheduler with global knowledge of the network topology. Hence, the results can be viewed as bounds on the performance when some distributed scheduler is implemented. The performance of distributed computing with a distributed medium access protocol (such as random access), which does not assume any global

TABLE II

EXPRESSIONS FOR ENERGY EXPENDITURE AND COMPUTATIOF TIME OBTAINED FROM SIMULATIONS WITH $\Delta = 0.5$

Algorithm	Energy Expenditure	Computation Time
Multi-hop transmission	$0.24\ n\sqrt{\frac{n}{\log n}}$	$1.64(n)$
Tree Algorithm	$0.75(n)$	$3.15(\sqrt{n}\log n)$
Ripple Algorithm	$1.44(n\sqrt{n}\ \log n)$	$9.6(\sqrt{n}\ \log n)$

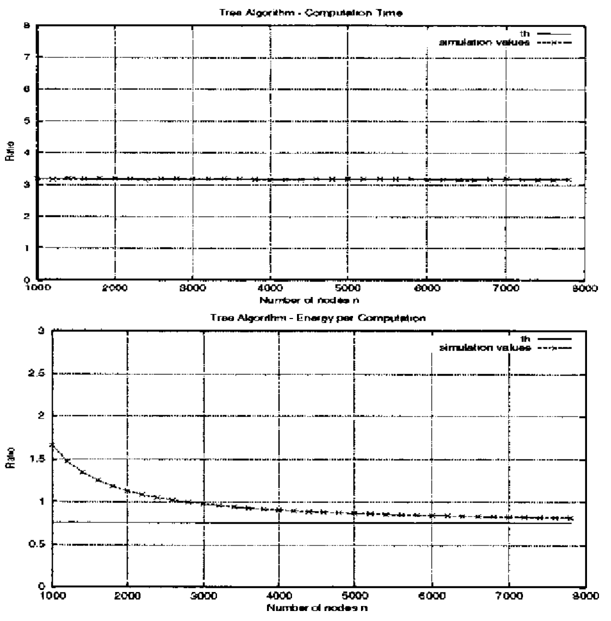

Fig. 5. Tree Algorithm: (upper panel) Ratio of observed computation time to the corresponding asymptotic order, and flower panel) the ratio of the observed energy expenditure per computation and the corresponding asymptotic order. The flat lines show the estimate of the constant multiplying the asymptotic order.

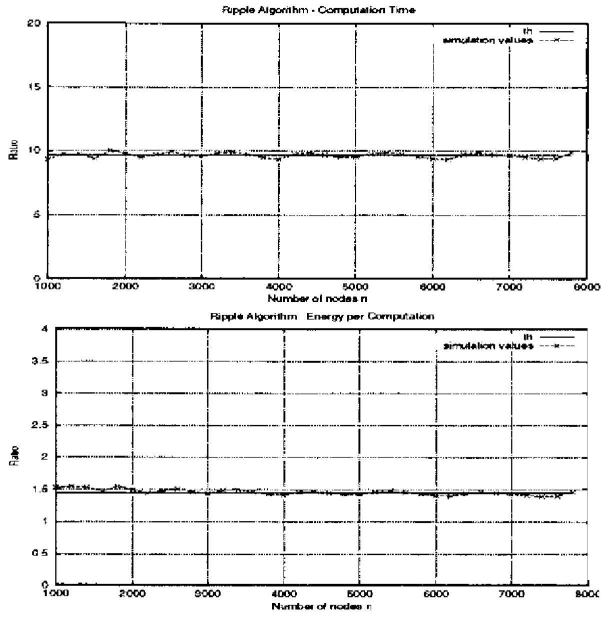

Fig. 6 Ripple Algorithm. (upper panel) Ratio of observed computation bme to the corresponding asymptotic order, and flower panel) the ratio of the observed energy expenditure per computation and the corresponding asymptotic order The flat lines show the estimate of the constant multiplying the asymptotic order.

knowledge, is a topic of some of our ongoing work in this direction.

VIII. APPENDIX

Proof: (Lemma 3.1): Part (2): Consider two nodes separated by distance d. Since the transmission range is $r(n)$, one hop can cover the distance of at most $r(n)$. Clearly,

$$\mathcal{P}^n\left(\underline{H}(d) \geq \frac{d}{r(n)}\right) = 1 \quad \text{for all } n \qquad (6)$$

To obtain the upper bound on the number of hops, we make the construction shown in Figure 8. We draw a rectangle of sides d and h joining the points **A** and B as shown in the figure. We divide the rectangle into bins of sides $b = \frac{r(n)}{p}$ and h. There are $p\frac{d}{r(n)}$ bins [3]. The bins are indexed. We select h such that any node in a bin can communicate with any other node in the adjacent bins. Hence, $h^2 + \left(\frac{2r(n)}{p}\right)^2 = r^2(n)$. This gives $h = \frac{\sqrt{p^2-4}}{p} r(n)$. The area of a bin is $h\frac{r(n)}{p} = \frac{\sqrt{p^2-4}}{p^2} r^2(n)$.

We note that the nodes in the adjacent bins are in each other's range. Further, if every bin contains at least one node, then there exists a path contained in the rectangle. However, for the path to be in the rectangle, it is not necessary to have a node in every bin. Hence,

$$\begin{aligned}
1 &\geq \mathcal{P}^n(\text{ a path exists in the rectangle})\\
&\geq \mathcal{P}^n\left(\cap_{i=1}^{p\frac{d}{r(n)}} \text{bin } i \text{ in the rectangle has at least one node}\right)\\
&= 1-\mathcal{P}^n\left(\cup_i^{p\frac{d}{r(n)}} \text{bin } i \text{ in the rectangle has no node}\right)\\
&\geq 1-\sum_i^{p\frac{d}{r(n)}} \mathcal{P}^n(\text{bin } i \text{ in the rectangle has no node})\\
&= 1-p\frac{d}{r(n)}\left(1-\frac{\sqrt{p^2-4}}{\pi p^2} r^2(n)\right)^n
\end{aligned}$$

[3]In the entire proof we have used $p\frac{d}{r(n)}$ as an integer which may not be the case. Precisely, it should be $\lfloor p\frac{d}{r(n)}\rfloor$. However, asymptotically (as $n \rightarrow$ w), $r << d$ and $p\frac{d}{r(n)}$ can be approximated as a large integer.

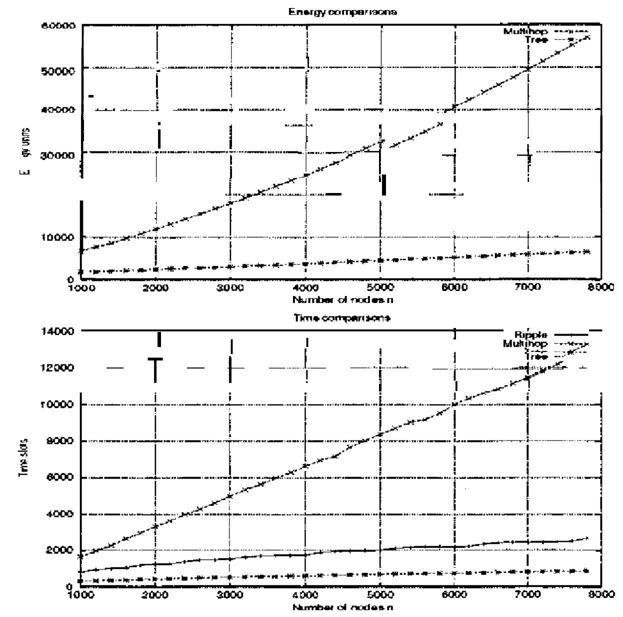

Fig. 7. Comparison of energy consumption (upper panel) and computation **time flower panel) per computation for the three algorithms. The energy required** for **the ripple algorithm is not shown because of** the **very large values.**

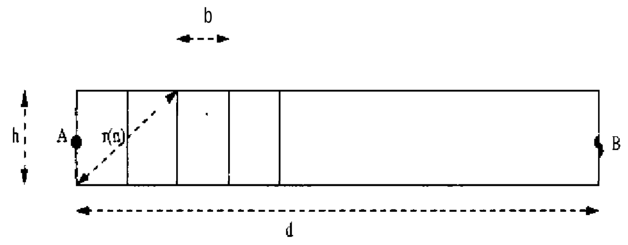

Fig. 8. Construction for an upper bound on hop number between the **nodes A and B**

The last **step** uses **the** facts that the area of a bin **is** $\frac{\sqrt{p^2-4}}{p^2}r^2(n)$, that of the entire field is π and uniform distribution of the **n** nodes.

Since the radius of **the** field is 1, we have $0 \le$ d $\le$ **2.** We note **that** p is finite. **Also,** $r(n) \to 0$ as $n \to \infty$. Thus,

$$\lim_{n\to\infty}\left(1-\frac{\sqrt{p^2-4}}{\pi p^2}r^2(n)\right)^{-\frac{\pi p^2}{(\sqrt{p^2-4})r^2(n)}} = e$$

Hence, for large n,

$$p\frac{d}{r(n)}\left(1-\frac{\sqrt{p^2-4}}{\pi p^2}r^2(n)\right)^n \approx pd\,e^{-\left(\frac{\sqrt{p^2-4}}{\pi p^2}nr^2(n)+\ \log r(n)\right)}$$

Thus, in order to show that $p\frac{d}{r(n)}\left(1-\frac{\sqrt{p^2-4}}{\pi p^2}r^2(n)\right)^n \to 0$ as $n \to \infty$, **we** need

$$\left(\frac{\sqrt{p^2-4}}{\pi p^2}nr^2(n)+\ \log r(n)\right) \to \infty \ \text{ as } \ n \to \infty$$

If we choose $r(n) = \sqrt{\frac{k\log n}{n}}$, $k > 0$, it can be shown that the above expression holds only if $k \ge \frac{\pi p^2}{2\sqrt{p^2-4}}$ Thus, if **we** choose $r(n) = \sqrt{\frac{\pi p^2}{2\sqrt{p^2-4}}\frac{\log n}{n}}$, then,

$$\lim_{n\to\infty} \mathcal{P}^n \text{ (a path from node } \mathbf{A} \text{ to B } \textbf{exists} \text{ in the rectangle)} = 1$$

We note that if a **path** lies in the rectangle, an upper bound on the number of hops in that path is $p\frac{d}{r(n)}$ and hence the upper **bound** on the number of hops in the shortest path $\overline{H}(d) \le p\frac{d}{r(n)}$. However. the converse is not true. Hence,

$$\mathcal{P}^n\left(\overline{H}(d) \le p\frac{d}{r(n)}\right) \ge$$
$$\mathcal{P}^n \text{ (a path from point } \mathbf{A} \text{ to } \mathbf{B} \text{ exists in the rectangle)}$$

Hence,

$$\mathcal{P}^n\left(\overline{H}(d) \le p\frac{d}{r(n)}\right) = 1 \ \text{ as } \ n \to \infty \qquad (7)$$

Equations 6 and 7 prove the Lemma. □

We need the following result established by Xue **and** Kumar [7]. In this set up, the square **field** of unit **area** is split into small squares (cells) of size $\sqrt{\frac{K\log n}{n}}$ x $\sqrt{\frac{K\log n}{n}}$ by a grid. The cells are indexed by $i \in (1,\dots,\frac{n}{K\log n}\}$.

Theorem 8.1 (Xue and Kumar [7]): Lei $K > \frac{1}{\log(\frac{4}{e})}$, and let $\mu^* \in (0,1)$ be the only root of the equation

$$-\mu^* + (1+\mu^*)\log(1+\mu^*) = 1/K$$

We tessellate the square field of unit area as mentioned above. There are n **nodes deployed uniformly** in the **square field.** Let N_i be the number of nodes in the i^{th} cell; **and** M_n be the total number of cells $\left(M_n = \frac{n}{K\log n}\right)$. Then the following holds for any $\mu > \mu^*$

$$\lim_{n\to\infty} \mathcal{P}^n\left(\max_{1\le i\le M_n} |N_i - K\log n| \le \mu K\log n\right) = 1$$

□

Remark: This implies that the **number** of nodes lying in any cell is uniformly bounded between $(1-\mu)K\log n$ **and** $(1+\mu)K$ **log**n w.p. 1 as $n \to \infty$. We note that the expected number of nodes in a cell **is** $K\log n$; thus μ captures the range of variation from the mean. We note that this result also holds in our case of circular field of unit radius. D

Proof: (Lemma 3.2): In order to prove $\gamma = \Theta\left(\frac{n}{\log n}\right)$ asymptotically, we need two inequalities which **we** obtain in two parts(a) **and** (b).

(a): Consider **the** two simultaneous transmissions $a \to j$ and $k \to$ *1.* By the triangle inequality,

$$\begin{aligned} |X_j - X_l| + |X_l - X_k| &\ge |X_j - X_k| \\ |X_j - X_l| + r(n) \ge |X_j - X_l| + |X_l - X_k| &\ge |X_j - X_k| \ge (1+\Delta)r(n \\ |X_j - X_l| &\ge \Delta r(n) \end{aligned}$$

This has been shown in [6].

Thus, the simultaneous transmissions $i \rightarrow j$ **and** $k \rightarrow l$ necessitate the condition $|X_j - X_l| \geq \Delta r(n)$, i.e., the minimum distance between any two receivers receiving simultaneously must be at least $\Delta r(n)$. This necessary condition is equivalent to the statement that the disks of radius $\frac{\Delta}{2} r(n)$ centered around the receivers should be disjoint.

Note that this is the most compact arrangement of receivers possible. For this condition to suffice for the successful receptions at all the receivers, the transmitters and receivers must be located in a specific manner; any arbitrary location of the transmitters will not permit simultaneous transmissions.

The area of a disk of radius $\frac{\Delta}{2} r(n)$ is $\frac{\pi\Delta^2}{4} r^2(n)$. Hence, **an** upper bound on the number of simultaneous transmissions in the field A for all $\mathcal{S}$,

$$\begin{aligned} \overline{\gamma}(\mathcal{S}) &< \frac{\pi}{\frac{\pi\Delta^2}{4} r^2(n)} = \frac{4}{\Delta^2 r^2(n)} \\ \overline{\gamma}(\mathcal{S}) &< \frac{2}{\pi\Delta^2} \frac{n}{\log n} \end{aligned} \qquad (8)$$

The bound above is an unachievable upper bound since the actual number of the disks that can be accommodated **in** the field will be less than the above bound.

(b): To obtain the lower bound, we consider the construction shown in the Figure 9. We inscribe a square field inside the given circular field $\boldsymbol{A}$ **and** partition the square field into small squares of size $(2 + \Delta) r(n)$ by a grid, We shall find a lower bound on the number of simultaneous transmissions possible only inside the **square.** Clearly, This is **also** a lower bound on $\underline{\gamma}(\mathcal{S})$. This is to avoid the cells which are at the boundary **and** **are** not completely within the field.

Note that Theorem 8.1 holds as $K = 2\pi(2 + \mathbf{A})' >$

Hence the probability that the number of nodes in any cell is between $2\pi(1-\mu)(2+\Delta)^2 \log n$ and $2\pi(1+\mu)(2+\Delta)^2 \log n$ goes to 1 as $n \rightarrow \infty$. Now we number the cells by their X and $\boldsymbol{Y}$ coordinates. The pair (i, j), where i and j are integers, denotes a cell. The origin can be any cell. We mark the cells whose coordinates are of the form $(2l, 2k)$ **(say,** the hashed cells in Figure 9). We note that **my** node from one such marked cell is at least $(2 + \Delta) r(n)$ distance away from any node in some other marked cell. Thus, if we choose one node from each of the marked cells, they can have simultaneous transmissions irrespective of the locations of their receivers. After scheduling these cells, **we** can schedule the transmissions of **the** nodes in **the** cells **with** coordinates of the form ***(21*** $+ 1, 2k + 1)$ (say, the filled cells), $(2l, 2k + 1)$ **and** $(2l + 1, 2k)$ (unfilled cells in last two cases) in three **slots.** Thus in four **slots,** a **node** from each cell is scheduled. We note that the fact that the transmissions are successful irrespective of *the* location of the receivers gives a lower bound on the number of simultaneous transmissions possible.

The size of the square is $\sqrt{2}$. Thus the total number of cells that can be accommodated in the square is

$$\frac{2}{(2+\Delta)^2 r^2(n)} = \frac{1}{\pi(2+\Delta)^2} \frac{n}{\log n}$$

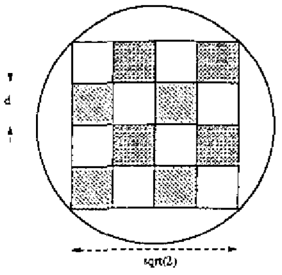

Fig. 9. **Construction** for **lower bound on** hop number of **simultaneous transmissions.** $d = (2 + \Delta) r(n)$.

In one slots, nodes from only $\frac{1}{4}^{th}$ of the cells can uansmit. We need to show that the probability of any cell being empty is zero as $n - \infty$. This is evident from the result of **Xue** and Kumar [7], stated earlier in the paper. Thus

$$\lim_{n\rightarrow\infty} \mathcal{P}^n \left(\frac{1}{4\pi(2+\Delta)^2} \frac{n}{\log n} < \underline{\gamma} \right) = 1 \qquad (9)$$

Thus, Equations **8 and** 9 prove **the** lemma. □

Proof: (Lemma **3.3):** In a round, each node transmits once to one of its neighbours. There are n transmissions in a round. **A** lower bound can be obtained from Equation **8.** We know that the number of simultaneous transmissions is bounded by $\overline{\gamma}(\mathcal{S}) < \frac{2}{\pi\Delta^2} \frac{n}{\log n}$. Hence a lower bound on the round time $\underline{T}(\mathcal{S})$ for all $\mathcal{S}$ is

$$\begin{aligned} \underline{T}(\mathcal{S}) &\geq \frac{n}{\overline{\gamma}(\mathcal{S})} \\ \underline{T}(\mathcal{S}) &\geq \frac{\pi}{\eta} \Delta^2 \log n \end{aligned} \qquad (10)$$

To find an upper bound, we again split the **field** $\boldsymbol{A}$ into squares of size $(2 + \Delta) r(n)$. We know that **in** four slots all the cells **get** one transmission scheduled, hence the time required to complete a round is determined by the maximum number of nodes a cell can have, From the previous lemma, we know that $2\pi(1 + \mu)(2 + \Delta)^2 \log n$ bounds this number **with** high probability. Hence we get an upper bound on the round time.

$$\lim_{n\rightarrow\infty} \mathcal{P}^n (\overline{T} \leq 8\pi(1+\mu)(2 + \Delta)^2 \log n) = 1 \qquad (11)$$

Equations 10 and 11 together prove the lemma. □

REFERENCES

[1] 1. Akyildiz, W. Su, Y. Sankarasubramanian, and E. Cayirci. **Wireless** sensor **networks: a survey.** *Computer Networks*, 38:393–422, 2002.

[2] **Arvind Giridhar and I? R. Kumar. Data fusion over sensor networks: Computiug and communicating functions of measurements.** *IEEE Journal on Selected Areas* **in** *Communications*, **submitted.**

[3] Aditya Karnik and Anurag **Kumar. Distributed optimal** self-organisation **in a class of ad** hoc sensor **networks. In** *IEEE Infocom*, 2004.

[4] Nilesh Khude. **Distributed computation** on **wireless 'sensor networks. Master's thesis, ECE Department. Indian Institute of Science, Bangalore,** June 2004.

[5] **Piyush Gupta and** P.R. Kumar. **Critical Power for Asymptotic** Connectivity **in Wireless Networks. A Volume in** Honour **of** W.H.Fleming **in Stochastic Analysis, Control, Optimisation and Applications. 1998.**

[6] **Piyush Gupta and** P.R. **Kumar. The Capacity of Wireless Networks.** *IEEE Transactions* **on** *Information Theory*, IT46(2):388–404, **March** 2000.

[7] **Feng** Xue **and P.** R. Kumar. **The** number **of neighbors needed** for **connectivity** of wireless networks. *Wireless Networks*, to **appear.**