Quick, Decentralized, One-Shot Max Function Computation Using Timer-Based Selection

Arjun Anand and Neelesh B. Mehta, *Senior Member, IEEE*

***Abstract*—In several wireless sensor networks, it is of interest to determine the maximum of the sensor readings and identify the sensor responsible for it. This has been referred to as the max function computation problem in the literature. We propose a novel, decentralized, timer-based max function computation (TMC) algorithm. In it, the sensors do not transmit their readings in a centrally pre-defined sequence. Instead, they are divided into clusters. The computation occurs over two stages. In the first stage, the nodes contend with each other using a decentralized timer scheme to transmit their reading to their cluster heads. In the second stage, the cluster heads contend in a similar manner. The main challenge that arises with the use of the timer scheme is the possibility of collisions, which can make the algorithm fail in finding the maximum. We optimize the algorithm to minimize the average time required to determine the maximum subject to a constraint on the probability that it fails to find it due to collisions. Extensive benchmarking shows that TMC requires lower selection times and far fewer transmissions on average than other approaches proposed in the literature.**

I. Introduction

Wireless sensor networks (WSN) are increasingly being deployed to track temperature, pressure, pollution, and chemical composition [1]. In several of these applications, it is often of interest to determine the maximum of the sensor readings in the network and identify the sensor responsible for it. This helps early detection of an impending event such as a fire or helps monitor and identify an egregious source of pollution. This has been referred to as the max function computation problem in the literature [2]–[8]. The following framework captures the essence of the problem in a wireless system that consists of n nodes. Each node i has a real-valued local *metric* μ_i. The *best node* is defined as the node that has the highest metric. A sink or a fusion node needs to determine the identity of the best node and its metric.

The max function can be written as a composition of itself. For example, $\max\{\mu_1, \mu_2, \mu_3\} = \max\{\max\{\mu_1, \mu_2\}, \mu_3\}$. This leads to bandwidth savings because all the sensors do not need to send their data all the way to the fusion node. Instead, the data can be aggregated en route. Another powerful technique that has been used is block computation in which sensor nodes take measurements for several time instants, and the max function for each of these time instants is computed in one go [2], [3], [5]. The nodes' transmissions occur according to a pre-specified schedule and depend on the contents of the previous transmissions. A pipelined computation model with message passing between the nodes is studied in [4].

The authors are with the Dept. of Electrical Communication Eng., Indian Institute of Science, Bangalore, India.

Emails: arjunanand1989@gmail.com, nbmehta@ece.iisc.ernet.in

This work was partially supported by a research grant from ANRC.

While block computation reduces the number of transmissions significantly, it does have its drawbacks. Firstly, it introduces a large delay since every node has to collect a sufficiently large number of measurements in a block, which can make it impractical [2]. The block size also grows with the number of nodes in the network. Another practically relevant model that avoids the delays associated with block computation is one shot computation, in which the max function is computed once in every coherence interval of the metrics [2], [5]. We shall focus on one-shot computation in this paper.

We propose a novel timer based max function computation (TMC) algorithm for one-shot computation, which differs markedly from the approaches pursued in the literature. Even for one-shot computation, we show that a marked reduction in the average selection time and average number of transmissions can be achieved.

TMC is based on the timer scheme, in which each node sets a timer as a function of its metric, and starts counting its timer down. It transmits a small timer packet when its timer expires. The metric-to-timer mapping is a *monotonic non-increasing* function, which ensures that the node with the maximum metric transmits first. Consequently, only a small fraction of the nodes transmit. While the timer scheme has been employed for opportunistic selection in cooperative relay networks [9] and wireless sensor networks [10], the focus was on selecting the node for data transmission. Our application of the timer scheme for max function computation in a multi-tier network is novel. As we shall see, this leads to new challenges about the optimal design of the timer scheme itself.

Another novelty in our model is the introduction of the failure probability η in max function computation. Such modeling is typical in wireless system design. The choice of η permits a new trade-off between the time required for computing the maximum and the reliability with which it is computed. While the algorithms in [2]–[4] focus on $\eta = 0$, we show that significant reductions in the average selection time can be achieved even when η is close to 0.

Finally, TMC is considerably simpler to implement when compared to the algorithms in the literature, which assume a centralized scheduler that determines the role of each node and which were developed primarily to prove order optimal results [3], [4]. Making the nodes aware of their roles can incur considerable overhead [2].

TMC works as follows. The sensor nodes are grouped into clusters. The nodes in a cluster can sense transmissions by other nodes within the cluster. However, they need not have to sense transmissions by nodes in other clusters. Each cluster

978-1-4799-2003-7/14/$31.00 ©2014 IEEE

has a cluster head, which can sense transmissions by other cluster heads. The computation takes place in two stages:

1) Intra-cluster stage: In each cluster, every node maps its metric to a timer, and starts counting its timer down. When its timer expires, the node transmits a packet containing its identity and metric to the cluster head. The cluster head simply decodes the first packet that it receives to identify the best node within its cluster and its metric. The other nodes in the cluster do not transmit once they sense an ongoing transmission in the cluster. The cluster head uses this metric, which we shall refer to as its *priority*, in the subsequent inter-cluster stage.[1]

2) Inter-cluster stage: In it, each cluster head now maps its priority to a timer value using another metric-to-timer mapping and transmits it to the sink node when its timer expires. Thus, at the end of this stage, the sink node can now determine the identity of the best node and its metric.

New Design Challenges and Analysis: A new challenge that arises with the use of the timer scheme is collisions. For example, in the intra-cluster stage, the timer packet transmitted by the best node in a cluster will not be decoded by the cluster head if the timer of the second best node in the cluster expires within a *vulnerability window* Δ of its transmission [9]. Collisions can occur in the inter-cluster stage as well. The parameter Δ depends on the physical layer of the system [9]. It includes the maximum propagation delay, the maximum delay spread in the channels seen by the nodes, receive-to-transmit switching time, and time synchronization errors, if any, among the nodes. Thus, only coarse time synchronization between the nodes is needed to implement TMC.

We determine the optimal parameters of the timer schemes used by the two stages that minimize the average selection time subject to a constraint on the failure probability, which is the probability that the best node is not identified by the sink. We observe that TMC reduces the average number of transmissions by up to two orders of magnitude and the average max function computation time by up to one order of magnitude over adaptations of the tree and the ripple algorithms [4].

The paper is organized as follows. Section II sets up the system model and the problem statement. Section III presents TMC and optimizes it. Simulation results are presented in Section IV followed by our conclusions in Section V.

II. System Model

Consider a system with n sensor nodes and a sink. The nodes are distributed in a square region as shown in the Figure 1. The entire region is divided into k_1 clusters. Each cluster has k_2 nodes located within it. Thus, $n = k_1 k_2$. Every cluster has a cluster head, which can transmit directly to the sink node. It can be located anywhere within the cluster.[2]

[1]For ease of expositions, we assume that the cluster head does not have a metric of its own to report. If it does, then it simply sets its priority as the maximum of its metric and the metric of the best node in its cluster.

[2]The formation of clusters and the choice of the cluster heads can be optimized using algorithms such as LEACH [11]. We do not delve into this aspect in this paper.

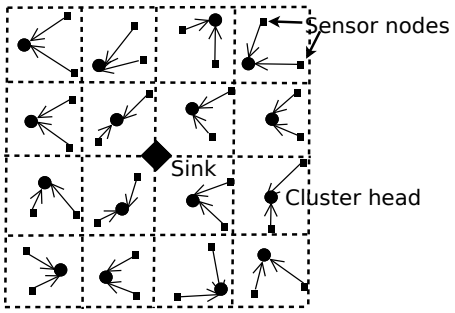

Fig. 1. A two-tier tree model in which the sink is the root, k_1 cluster heads constitute the first level, and $n = k_1 k_2$ sensor nodes constitute the leaves.

The metrics of the nodes are assumed to be independent and identically distributed (i.i.d.). The independence assumption is justified, for example, when the sensor readings decorrelate over distance. The identically distributed assumption helps make the analysis tractable, and is widely used in the literature [9], [12], [13]. The scenario with correlated sensor readings is beyond the scope of this paper. We note that several related papers also make similar assumptions [12], [14]. Further, the proposed scheme works even in the presence of correlation, though it is no longer optimal.

The metric is assumed to be uniformly distributed over $[0, 1]$ in order to simplify the design and analysis. This is justified because any random variable (RV) with a continuous cumulative distribution function (CDF) can be transformed to an RV that is uniformly distributed in $[0, 1]$ as follows [15]. Let C denote the CDF of μ_i. Then the RV $\nu_i = C(\mu_i)$ can be shown to be uniformly distributed in $[0, 1]$. Since the CDF is a monotonically non-decreasing function, if a node i has the highest metric μ_i, then it is also the one with the largest ν_i. Knowing C is practically feasible since it changes at a rate that is several orders of magnitude slower than the instantaneous metrics. We note that this approach is different from [3]–[6], [8], in which prior knowledge of C is not assumed. However, the algorithm in [16] does assume knowledge of C.

Timer Scheme: Before we define the two stages of TMC, we first describe the timer scheme that will be used by them. It uses a discrete metric-to-timer mapping in which the timers expire only at $0, \Delta, \ldots, N\Delta$, where N is called the number of timer levels. This mapping has been proved to be optimal for maximizing the probability of selecting the best node and also for minimizing the average selection time when the metrics are i.i.d. and uniformly distributed over $[0, 1]$ [12]. It is fully characterized by $N + 1$ positive real numbers called *interval lengths* $\alpha[0], \ldots, \alpha[N]$ as follows. If a node's metric lies in the interval $(1 - \alpha[0], 1]$, then its timer expires immediately at 0. In general, if its metric lies in $\left(1 - \sum_{l=0}^{i} \alpha[l], 1 - \sum_{l=0}^{i-1} \alpha[l]\right]$, then its timer expires at time $i\Delta$. When its timer expires, a node transmits a timer packet of duration T_p. If the metric lies in $\left[0, 1 - \sum_{i=0}^{N} \alpha[i]\right)$, then its timer does not expire.

Intra-cluster Stage: Each cluster uses a timer scheme

with N_2 timer levels and interval length vector $\boldsymbol{\beta}_{N_2} = [\beta_{N_2}[0], \beta_{N_2}[1], \ldots, \beta_{N_2}[N_2]]$. A node whose timer expires transmits a packet containing its metric and identity to its cluster head.

In two scenarios the cluster head can fail to select the best node in its cluster, in which case it declares a selection failure.[3] The first scenario occurs when no node's timer expires within the duration $N_2\Delta$. The second scenario occurs when a timer of at least one other node expires at the same time as the best node, as a result of which a collision occurs. In case of a selection failure, the cluster head sets its metric as the least value possible for the metric, which is zero. This ensures that a selection failure in a cluster that does not contain the best node does not affect the chances of the best node being selected in the inter-cell stage.

Reuse Constraints: All clusters cannot transmit simultaneously because of interference constraints. Let r denote the number of clusters in which transmission can occur simultaneously. Therefore, with k_1 clusters, the intra-cluster stage takes place in k_1/r sub-stages, with the intra-cluster stage in r clusters occurring simultaneously. Since the cluster heads are not required to communicate with each other, each sub-stage in the intra-cluster stage takes $N_2\Delta+T_p$ time. This is because, in the worst case, a node may transmit at time $N_2\Delta$ and T_p duration is required to transmit the timer packet. Hence, the duration $\Gamma_2\left(\boldsymbol{\beta}_{N_2}\right)$ of the intra-cluster stage is $\frac{k_1}{r}\left(N_2\Delta + T_p\right)$.

Inter-cluster Stage: At the start of this stage, each cluster head knows the maximum metric within its cluster, which we shall refer to as its *priority*. The only exceptions are clusters in which a selection failure has occurred, for whom the priority is zero. Now, the cluster heads use a timer scheme with N_1 timer levels and interval length vector $\boldsymbol{\alpha}_{N_1} = [\alpha_{N_2}[0], \ldots, \alpha_{N_2}[N_2]]$. When its timer expires, the cluster head transmits its priority and the identity of the node it selected in the intra-cluster stage.[4]

Since the cluster heads can sense each other's transmissions, the inter-cluster stage ends as soon as the first transmission occurs or the time available for it, which is $N_1\Delta + T_p$, runs out. This is unlike the intra-cluster stage.

Comments:

1. Note that the priority M_j of cluster head j is the maximum of k_1 metrics or is 0, in case of a selection failure in the cluster. Thus, M_j is not uniformly distributed over $[0, 1]$. Its CDF can be determined using order statistics. Further, the probability that M_j is 0 is equal to $p = 1 - k_2 \sum_{l=0}^{N_2} \beta_{N_2}[l] \left(1 - \sum_{i=0}^{l} \beta_{N_2}[l]\right)^{k_2-1}$, which is non-zero. As before, M_j can be transformed into an equivalent metric for the inter-cluster stage such that it is uniformly distributed over $[0, 1]$. This is achieved using the CDF transformation technique, which is outlined at the beginning of this section, and a technique called proportional expansion [14], which randomizes the metric to uniformly take a value between 0 and p in case $M_j = 0$. Doing so simplifies the analysis that follows below for the inter-cluster stage.

2. We assume that the metric is real-valued. In practice, a quantized version of it will be transmitted.

3. Note that the nodes and cluster heads do not need to decode each other's transmissions. Depending on the stage, they only need to sense the presence of transmissions by other nodes or cluster heads. This is unlike [3], [6], [7] in which a node decodes some transmissions that precede its transmission.

[3]We assume classical MAC model for collisions as in [12], [14].

[4]If the priority is zero, then it implies a selection failure, and the identity of the node is irrelevant.

III. Optimal Design of TMC

Our objective is to determine the optimal $\boldsymbol{\alpha}_{N_1} \in \left(\mathbb{R}^+\right)^{N_1+1}$, $\boldsymbol{\beta}_{N_2} \in \left(\mathbb{R}^+\right)^{N_2+1}$, $N_1 \in \mathbb{Z}^+$, and $N_2 \in \mathbb{Z}^+$ that minimize the expected max function computation time $\Gamma\left(\boldsymbol{\alpha}_{N_1}, \boldsymbol{\beta}_{N_2}\right)$ over the network subject to the probability of selection failure $F\left(\boldsymbol{\alpha}_{N_1}, \boldsymbol{\beta}_{N_2}\right)$ not exceeding η.

The optimization problem can be stated as follows:

$$\mathcal{OP}_1 : \text{minimize} \quad \Gamma\left(\boldsymbol{\alpha}_{N_1}, \boldsymbol{\beta}_{N_2}\right), \tag{1}$$

$$\text{s.t.} \quad F\left(\boldsymbol{\alpha}_{N_1}, \boldsymbol{\beta}_{N_2}\right) \leq \eta, \tag{2}$$

$$0 \leq \alpha_{N_1}[i] \leq 1, \quad i = 0, \ldots, N_1, \tag{3}$$

$$0 \leq \beta_{N_2}[j] \leq 1, \quad j = 0, \ldots, N_2, \tag{4}$$

$$\sum_{i=0}^{N_1} \alpha_{N_1}[i] \leq 1 \text{ and } \sum_{j=0}^{N_2} \beta_{N_2}[j] \leq 1, \tag{5}$$

$$N_1, N_2 \in \mathbb{Z}^+. \tag{6}$$

The constraints (3), (4), and (5) ensure that the timer interval lengths are positive and together lie in $[0, 1]$. $\mathcal{OP}_1$ is a combinatorial, non-convex, stochastic optimization problem, and is intractable. We show below that a tighter reliability constraint, which is based on the union bound, leads to a tractable, nearly-optimal and insightful solution. For this, we need to understand when TMC fails to select the best node. This happens when:

1) The inter-cluster stage fails to select the best cluster head. Its probability is denoted by $F_1\left(\boldsymbol{\alpha}_{N_1}\right)$.
2) Or, the cluster with the best node fails to select it in the intra-cluster stage. Its probability is denoted by $F_2\left(\boldsymbol{\beta}_{N_2}\right)$.

It is important to note that a selection failure in a cluster that does not contain the best node does not matter.

Applying the union bound, we get

$$F\left(\boldsymbol{\alpha}_{N_1}, \boldsymbol{\beta}_{N_2}\right) \leq F_1\left(\boldsymbol{\alpha}_{N_1}\right) + F_2\left(\boldsymbol{\beta}_{N_2}\right). \tag{7}$$

Instead of $\mathcal{OP}_1$, we solve the following constrained optimization problem:

$$\mathcal{OP}_2 : \text{minimize} \quad \Gamma\left(\boldsymbol{\alpha}_{N_1}, \boldsymbol{\beta}_{N_2}\right), \tag{8}$$

$$\text{s.t.} \quad F_1\left(\boldsymbol{\alpha}_{N_1}\right) + F_2\left(\boldsymbol{\beta}_{N_2}\right) \leq \eta, \tag{9}$$

along with the constraints in (3), (4), (5), and (6). The constraint in (9) is tighter because a solution of $\mathcal{OP}_2$ automatically satisfies the constraint in (2).

We first derive expressions for $F_1\left(\boldsymbol{\alpha}_{N_1}\right)$ and $F_2\left(\boldsymbol{\beta}_{N_2}\right)$.

Result *1:* The failure probability $F_2\left(\boldsymbol{\beta}_{N_2}\right)$ in the cluster that contains the best node is given by

$$F_2\left(\boldsymbol{\beta}_{N_2}\right) = 1 - \frac{k_1 k_2}{k_1 k_2 - k_2 + 1} \sum_{l=0}^{N_2} \left(1 - \sum_{i=0}^{l} \beta_{N_2}[i]\right)^{k_2 - 1} \times \left(\left[1 - \sum_{i=0}^{l-1} \beta_{N_2}[i]\right]^{k_1 k_2 - k_2 + 1} - \left[1 - \sum_{i=0}^{l} \beta_{N_2}[i]\right]^{k_1 k_2 - k_2 + 1} \right). \tag{10}$$

The failure probability $F_1\left(\boldsymbol{\alpha}_{N_1}\right)$ in the inter-cluster stage is

$$F_1\left(\boldsymbol{\alpha}_{N_1}\right) = 1 - k_1 \sum_{l=0}^{N_1} \alpha_{N_1}[l] \left(1 - \sum_{j=0}^{l} \alpha_{N_1}[j]\right)^{k_1 - 1}. \tag{11}$$

Proof: The proof is given in Appendix A. ∎

The expected selection time $\Gamma\left(\boldsymbol{\alpha}_{N_1}, \boldsymbol{\beta}_{N_2}\right)$ is the sum of the expected duration $\Gamma_2\left(\boldsymbol{\beta}_{N_2}\right) = \frac{k_1}{r}\left(N_2\Delta + T_p\right)$ of the intra-cluster stage and the expected duration $\Gamma_1\left(\boldsymbol{\alpha}_{N_1}\right)$ of the inter-cluster stage. $\Gamma_1\left(\boldsymbol{\alpha}_{N_1}\right)$ can be shown to be [12]

$$\Gamma_1\left(\boldsymbol{\alpha}_{N_1}\right) = T_p + \Delta \sum_{l=0}^{N_1 - 1} \left(1 - \sum_{j=0}^{l} \alpha_{N_1}[j]\right)^{k_1}. \tag{12}$$

The following important result shows that solving $\mathcal{OP}_2$ is equivalent to solving two simpler problems.

Result *2:* Solving $\mathcal{OP}_2$ is equivalent to separately solving the following two sub-problems:

$$\mathcal{SP}_1 : \text{minimize} \quad \Gamma_1\left(\boldsymbol{\alpha}_{N_1}\right) + \lambda F_1\left(\boldsymbol{\alpha}_{N_1}\right), \tag{13}$$

$$\mathcal{SP}_2 : \text{minimize} \quad \frac{k_1}{r}\left(N_2\Delta + T_p\right) + \lambda F_2\left(\boldsymbol{\beta}_{N_2}\right), \tag{14}$$

both subject to the constraints in (3), (4), (5), and (6). The constant $\lambda \geq 0$, which is common to $\mathcal{SP}_1$ and $\mathcal{SP}_2$, is chosen to meet (9) with equality and such a choice exists.

Proof: The proof is given in Appendix B. ∎

Notice that $\mathcal{SP}_1$ and $\mathcal{SP}_2$ deal with the inter-cluster stage and intra-cluster stage, respectively.

A. Solving $\mathcal{SP}_1$

Given any N_1, the optimal interval lengths $\alpha^*_{N_1}[j]$, $j = 0, \ldots, N_1$, are given by the following recursion:

$$\alpha^*_{N_1}[j] = \begin{cases} \frac{1 + \frac{\lambda}{\Delta} + \frac{1}{\Delta} Q^*_{N_1 - 1}(\lambda)}{1 + \frac{\lambda}{\Delta} k_1 + \frac{1}{\Delta} Q^*_{N_1 - 1}(\lambda)}, & j = 0, \\ \left(1 - \alpha^*_{N_1}[0]\right) \alpha^*_{N_1 - 1}[j-1], & 1 \leq j \leq N_1, \end{cases} \tag{15}$$

where $\alpha^*_0[0] = \frac{1}{k_1}$ and

$$Q^*_{N_1}(\lambda) = \Delta \sum_{l=0}^{N_1 - 1} \left(1 - \sum_{j=0}^{l} \alpha^*_{N_1}[j]\right)^{k_1} - \lambda k_1 \sum_{l=0}^{N_1} \alpha^*_{N_1}[l] \left(1 - \sum_{j=0}^{l} \alpha^*[j]\right)^{k_1 - 1}. \tag{16}$$

The proof follows from [12], and is not repeated here.

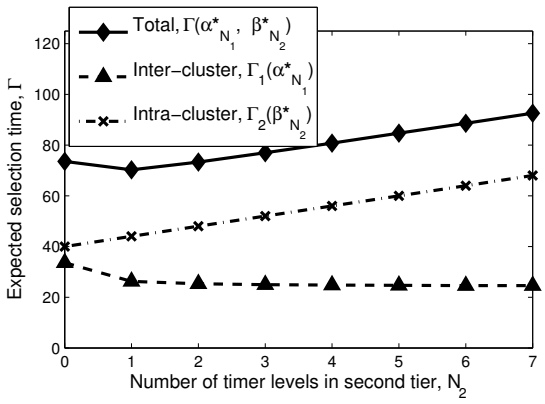

Fig. 2. Total expected selection time $\Gamma\left(\boldsymbol{\alpha}^*_{N_1}, \boldsymbol{\beta}^*_{N_2}\right)$, expected inter-cluster and intra-cluster selection times, $\Gamma_1\left(\boldsymbol{\alpha}^*_{N_1}\right)$ and $\Gamma_2\left(\boldsymbol{\beta}^*_{N_2}\right)$, respectively, as a function of N_2 ($\eta = 0.05$, $k_1 = 200$, $k_2 = 10$, $r = \frac{k_1}{4} = 50$, and $T_p = 10\Delta$).

Thus, only N_1 remains to be optimized in $\mathcal{SP}_1$. It is easy to see that $\Gamma_1\left(\boldsymbol{\alpha}_{N_1}\right)$ is a monotone non-increasing function of N_1 given η because increasing N_1 gives more variables for solving $\mathcal{SP}_1$. Therefore, given η, N_1 should be made as large as possible. In this asymptotic regime, the optimal interval lengths even simplify considerably [17], which makes implementing this stage straight forward.

B. Solving $\mathcal{SP}_2$

Result *3:* Given any N_2, the optimal interval lengths $\beta^*_{N_2}[j]$, $j = 0, \ldots, N_2$, are recursively given by

$$\beta^*_{N_2}[j] = \begin{cases} 1 - \left(\frac{\kappa(k_2 - 1)}{k_1 k_2 \left(\kappa - 1 + F_2\left(\boldsymbol{\beta}^*_{N_2 - 1}\right)\right)}\right)^{\frac{1}{k_1 k_2 - k_2 + 1}}, & j = 0, \\ \left(1 - \beta^*_{N_2}[0]\right) \beta^*_{N_2 - 1}[j-1], & 1 \leq j \leq N_2, \end{cases} \tag{17}$$

where $\beta^*_0[0] = 1 - \left(\frac{k_2 - 1}{k_1 k_2}\right)^{\frac{1}{k_1 k_2 - k_2 + 1}}$ and $\kappa = \frac{k_1 k_2}{k_1 k_2 - k_2 + 1}$.

Proof: The proof is given in Appendix C. ∎

Unlike N_1, a larger value of N_2 need not reduce $\Gamma\left(\boldsymbol{\alpha}^*_{N_1}, \boldsymbol{\beta}^*_{N_2}\right)$. This is shown in Figure 2, which plots $\Gamma\left(\boldsymbol{\alpha}^*_{N_1}, \boldsymbol{\beta}^*_{N_2}\right)$, $\Gamma_1\left(\boldsymbol{\alpha}^*_{N_1}\right)$, and $\Gamma_2\left(\boldsymbol{\beta}^*_{N_2}\right)$ as a function of N_2 for $\eta = 0.05$. We see that the optimal value for N_2 is one in this example. As N_2 increases, $F_2\left(\boldsymbol{\beta}^*_{N_2}\right)$ decreases and $\Gamma_2\left(\boldsymbol{\beta}^*_{N_2}\right)$ increases linearly. Given η, decreasing $F_2\left(\boldsymbol{\beta}^*_{N_2}\right)$ increases $F_1\left(\boldsymbol{\alpha}^*_{N_1}\right)$ since they must sum to η. However, increasing $F_1\left(\boldsymbol{\alpha}^*_{N_1}\right)$ reduces $\Gamma_1\left(\boldsymbol{\alpha}^*_{N_1}\right)$ [12]. Thus, a trade-off exists between increasing $\Gamma_1\left(\boldsymbol{\alpha}^*_{N_1}\right)$ and decreasing $\Gamma_2\left(\boldsymbol{\beta}^*_{N_2}\right)$.

IV. Performance Evaluation

For the simulations, a square field of unit area is considered, as shown in the Figure 1. Reuse constraints are such that any four neighboring clusters cannot transmit simultaneously. Hence, $r = \frac{k_1}{4}$. Further, $T_p = 10\Delta$. We benchmark TMC with the following schemes proposed in [4]. We study them because they can be adapted for one-shot computation.

1) Tree algorithm with direct transmission: A two stage computation is used. In the intra-cluster stage, the k_2 nodes

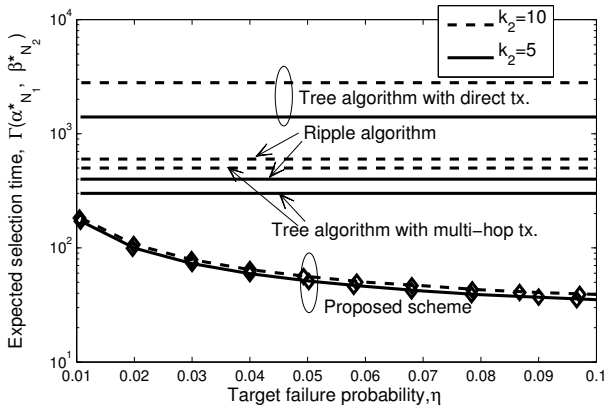

Fig. 3. Expected selection time $\Gamma\left(\boldsymbol{\alpha}^*_{N_1}, \boldsymbol{\beta}^*_{N_2}\right)$ as a function of the target failure probability η for different values of k_2 ($k_1 = 20$).

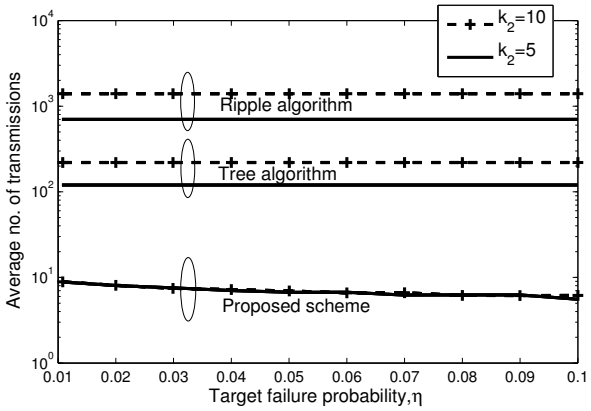

Fig. 4. Average number of transmissions as a function of the target failure probability η ($k_1 = 20$).

in a cluster transmit to their cluster head in a round-robin manner. This requires $4k_2T_p$ time after accounting for reuse constraints. Next, in the inter-cluster stage, the k_1 cluster heads transmit their priorities directly to the sink in a round-robin manner. This requires an additional k_1T_p time.

2) Tree algorithm with multi-hop transmission: In it, the cluster heads instead route their priorities column-wise and then row-wise to the sink, which turns out to be faster. First, the cluster heads of the left-most and right-most columns transmit to the cluster heads in the same row of the neighboring columns. Each such neighboring cluster head transmits the maximum of its priority and the priority it received to its inner neighbor, and so on. This process repeats until the priorities are routed to the central column. Thereafter, they are routed row-wise to the central sink node.

3) Ripple algorithm: This algorithm does not use clusters. The computation is instead divided into rounds. In a round, each node broadcasts a packet containing its metric and identity to its neighbors. Then, the metric of the best node propagates by one hop in each round. Due to interference constraints, a round is of duration $4k_2T_p$. The algorithm stops when the metric from the farthest node reaches the sink.

Figure 3 plots the expected selection time as the function of the target probability of selection failure η for $k_1 = 20$ and two different k_2. For $\eta = 0.04$, $k_1 = 20$, and $k_2 = 5$, it is 1.4, 18.8, and 7.2 times faster than the tree algorithm with multi-hop transmission, tree algorithm with direct transmission, and ripple algorithm, respectively. The corresponding numbers increase to 7.7, 9.2, and 43.3 for $k_2 = 10$. The figure also illustrates a trade-off between $\Gamma\left(\boldsymbol{\alpha}_{N_1}, \boldsymbol{\beta}_{N_2}\right)$ and η for TMC. A smaller value of η means a tighter constraint, which increases the total time required to select the best node. Only when η is less than 0.005 are any of the benchmark algorithms quicker.

Figure 4 plots the average number of transmissions, which is related to the bandwidth and energy consumed by the network in computing the maximum, as a function of η. Both variants of the tree algorithm require the same number of transmissions since all nodes transmit exactly once in a round of computation. We see that the TMC requires several orders of magnitude fewer transmissions, on average, than the benchmark schemes. For example, for $\eta = 0.04$, $k_1 = 20$, and $k_2 = 10$, TMC requires 31.4 and 199.1 times fewer transmissions than the tree and ripple algorithms, respectively. The reason for this marked reduction is that TMC ensures that often at most one transmission occurs per cluster in the intra-cluster stage and one transmission occurs in the inter-cluster stage.

V. Conclusions

We proposed a fast, decentralized algorithm called TMC for max function computation in a wireless network. By making nodes with higher metrics to transmit earlier, TMC achieves significant reductions in the average selection time and the average number of transmissions. Optimizing the parameters of the timer schemes for the two stages turned out to be a combinatorial, non-convex, stochastic constrained optimization problem. However, by replacing the failure probability with its union bound, the problem decomposed into two simpler sub-problems. The first one dealt with the intra-cluster stage and the second with the inter-cluster stage. These were then solved to find the optimal parameters. We saw that once the nodes are organized into clusters, only three constants N_1, N_2, and λ need to be conveyed to them and the cluster heads to fully specify the algorithm. Further, this signaling needs to be done only once in the beginning.

Appendix

A. Brief Proof of Result 1

1) Expression for $F_2\left(\boldsymbol{\beta}_{N_2}\right)$: Let the nodes be labeled such that the first k_2 nodes form the first cluster, the next set of k_2 nodes form the second cluster, and so on. Without loss of generality, let node 1 be the best node. Let S_1 denote the event that node 1 is selected by the inter-cluster stage. Therefore,

$$F_2\left(\boldsymbol{\beta}_{N_2}\right) = 1 - \Pr\left(S_1 | \mu_1 = \max\left\{\mu_1, \ldots, \mu_{k_1k_2}\right\}\right). \quad (18)$$

As the metrics are i.i.d., the probability that node 1 is the best node is $\frac{1}{k_1 k_2}$. Therefore, from Bayes' rule, we get

$$F_2\left(\boldsymbol{\beta}_{N_2}\right) = 1 - k_1 k_2 \Pr\left(S_1, \mu_1 \geq \mu_2, \ldots, \mu_1 \geq \mu_{k_1 k_2}\right). \quad (19)$$

We condition with respect to μ_1. Selection of node 1 in the intra-cluster stage depends only on the metrics of the k_2 nodes that are in its cluster. Since the metrics are i.i.d. and are uniformly distributed in $[0,1]$, we get

$$F_2\left(\boldsymbol{\beta}_{N_2}\right) = 1 - k_1 k_2 \times \mathbb{E}_{\mu_1}\left[\mu_1^{k_1 k_2 - k_2} \Pr\left(S_1, \mu_2 \leq \mu_1, \ldots, \mu_{k_2} \leq \mu_1 | \mu_1\right)\right]. \quad (20)$$

If $\mu_1 \in \left(1 - \sum_{i=0}^{l} \beta_{N_2}[i], 1 - \sum_{i=0}^{l-1} \beta_{N_2}[i]\right]$, then node 1 is selected only if $\mu_i \leq 1 - \sum_{i=0}^{l} \beta_{N_2}[i]$, for $i = 2, \ldots, k_2$. Summing up the probabilities for all the timer intervals in which μ_1 can lie and simplifying yields (10).

2) Evaluation of $F_1\left(\boldsymbol{\alpha}_{N_1}\right)$: If the priority of the cluster head whose cluster contains the best node lies in $\left(1 - \sum_{l=0}^{i} \alpha_{N_1}, 1 - \sum_{l=0}^{i-1} \alpha_{N_1}\right]$, then the priorities of all the other cluster heads must be less than $1 - \sum_{l=0}^{i} \alpha_{N_1}$ in order to avoid a collision. Summing over $i = 0, \ldots, N_1$ yields (11).

B. Brief Proof of Result 2

For a constant $\lambda \geq 0$, define

$$L^{\lambda}\left(\boldsymbol{\alpha}_{N_1}, \boldsymbol{\beta}_{N_2}\right) = \Gamma\left(\boldsymbol{\alpha}_{N_1}, \boldsymbol{\beta}_{N_2}\right) + \lambda\left(F_1\left(\boldsymbol{\alpha}_{N_1}\right) + F_2\left(\boldsymbol{\beta}_{N_2}\right)\right). \quad (21)$$

Let $\boldsymbol{\alpha}^*_{N_1}$ and $\boldsymbol{\beta}^*_{N_2}$ minimize (21) for a given λ and let L^{λ}_* denote the corresponding minimum value. Further, choose λ such that $F_1\left(\boldsymbol{\alpha}^*_{N_1}\right) + F_2\left(\boldsymbol{\beta}^*_{N_2}\right) = \eta$.[5] Clearly, $L^{\lambda}_* \leq L^{\lambda}\left(\boldsymbol{\alpha}_{N_1}, \boldsymbol{\beta}_{N_2}\right)$ for any $\boldsymbol{\alpha}_{N_1}$ and $\boldsymbol{\beta}_{N_2}$. Therefore,

$$\Gamma\left(\boldsymbol{\alpha}_{N_1}, \boldsymbol{\beta}_{N_2}\right) - \Gamma\left(\boldsymbol{\alpha}^*_{N_1}, \boldsymbol{\beta}^*_{N_2}\right) \geq \lambda\left(\eta - F_1\left(\boldsymbol{\alpha}_{N_1}\right) - F_2\left(\boldsymbol{\beta}_{N_2}\right)\right). \quad (22)$$

If $\boldsymbol{\alpha}_{N_1}$ and $\boldsymbol{\beta}_{N_2}$ are feasible solutions of $\mathcal{OP}_2$, then $F_1\left(\boldsymbol{\alpha}_{N_1}\right) + F_2\left(\boldsymbol{\beta}_{N_2}\right) \leq \eta$. Therefore, $\Gamma\left(\boldsymbol{\alpha}^*_{N_1}, \boldsymbol{\beta}^*_{N_2}\right) \leq \Gamma\left(\boldsymbol{\alpha}_{N_1}, \boldsymbol{\beta}_{N_2}\right)$. Hence, $\boldsymbol{\alpha}^*_{N_1}$ and $\boldsymbol{\beta}^*_{N_2}$ is the optimal solution. The decomposition into the two sub-problems $\mathcal{SP}_1$ and $\mathcal{SP}_2$ then follows because the objective function $L^{\lambda}\left(\boldsymbol{\alpha}_{N_1}, \boldsymbol{\beta}_{N_2}\right)$ can be written as a sum of $\frac{k_1}{r}\left(N_2\Delta + T_p\right) + \lambda F_2\left(\boldsymbol{\beta}_{N_2}\right)$ and $\Gamma_1\left(\boldsymbol{\alpha}_{N_1}\right) + \lambda F_1\left(\boldsymbol{\alpha}_{N_1}\right)$, which are only coupled through λ.

C. Proof of Result 3

We now prove that the solution in (17) achieves a lower bound on $F_2\left(\boldsymbol{\beta}_{N_2}\right)$, which proves that it is optimal. Taking out the common factor $\left(1 - \beta_{N_2}[0]\right)^{k_1 k_2}$ from the terms indexed by $l = 1, \ldots, N_2$ in (10), we get

$$F_2\left(\boldsymbol{\beta}_{N_2}\right) = 1 - \kappa\left(1 - \beta_{N_2}[0]\right)^{k_2 - 1}\left[1 - \left(1 - \beta_{N_2}[0]\right)^{\frac{k_1 k_2}{\kappa}}\right] - \left(1 - \beta_{N_2}[0]\right)^{k_1 k_2}\left[1 - F_2\left(\frac{\beta_{N_2}[1]}{1 - \beta_{N_2}[0]}, \ldots, \frac{\beta_{N_2}[N_2]}{1 - \beta_{N_2}[0]}\right)\right],$$

where $\kappa = \frac{k_1 k_2}{k_1 k_2 - k_2 + 1}$. If $\boldsymbol{\beta}^*_{N_2 - 1}$ is the optimum solution for $N_2 - 1$ timer levels, then

$$F_2\left(\boldsymbol{\beta}_{N_2}\right) \geq 1 - \left(1 - \beta_{N_2}[0]\right)^{k_1 k_2}\left[1 - F_2\left(\boldsymbol{\beta}^*_{N_2 - 1}\right)\right] - \kappa\left(1 - \beta_{N_2}[0]\right)^{k_2 - 1}\left[1 - \left(1 - \beta_{N_2}[0]\right)^{k_1 k_2 - k_2 + 1}\right]. \quad (23)$$

The lower bound in (23) is achieved when $\frac{\beta_{N_2}[1]}{1 - \beta_{N_2}[0]} = \beta^*_{N_2 - 1}[0], \ldots, \frac{\beta_{N_2}[N_2]}{1 - \beta_{N_2}[0]} = \beta^*_{N_2 - 1}[N_2 - 1]$.

The right hand side in (23) is minimized when

$$\beta^*_{N_2}[0] = 1 - \left(\frac{\kappa\left(k_2 - 1\right)}{\left[\kappa - 1 + F_2\left(\boldsymbol{\beta}^*_{N_2 - 1}\right)\right] k_1 k_2}\right)^{\frac{1}{k_1 k_2 - k_2 + 1}}. \quad (24)$$

Base Case of $N_2 = 0$: We have $F_2\left(\boldsymbol{\beta}_0\right) = 1 - \left(1 - \beta_0[0]\right)^{k_2 - 1}\left[1 - \left(1 - \beta_0[0]\right)^{k_1 k_2 - k_2 + 1}\right]$. Using the first order condition, we get $\beta^*_0[0] = 1 - \left(\frac{k_2 - 1}{k_1 k_2}\right)^{\frac{1}{k_1 k_2 - k_2 + 1}}$.

[5] That such a choice exists can be shown by applying the intermediate value theorem and proving that $F_1\left(\boldsymbol{\alpha}^*_{N_1}\right)$ and $F_2\left(\boldsymbol{\beta}^*_{N_2}\right)$ are continuous functions of λ.

References

[1] M. Ilya and I. Mahgoub, *Handbook of Sensor Networks: Compact Wireless and Wired Sensing Systems*. CRC Press, 2004.

[2] A. Giridhar and P. R. Kumar, "Toward a theory of in-network computation in wireless sensor networks," *IEEE Commun. Mag.*, vol. 44, no. 4, pp. 98–107, Apr. 2006.

[3] ——, "Computing and communicating functions over sensor networks," *IEEE J. Sel. Areas Commun.*, vol. 23, no. 4, pp. 755–764, Apr. 2005.

[4] N. Khude, A. Kumar, and A. Karnik, "Time and energy complexity of distributed computation of a class of functions in wireless sensor networks," *IEEE Trans. Mob. Comput.*, vol. 7, no. 5, pp. 617–632, May 2008.

[5] Y. Kanoria and D. Manjunath, "On distributed computation in noisy random planar networks," in *Proc. IEEE Int. Symp. Inf. Theory*, Jun. 2007, pp. 626–630.

[6] S. Kamath and D. Manjunath, "On distributed function computation in structure-free random networks," in *Proc. IEEE Int. Symp. Inf. Theory*, Jul. 2008, pp. 647–651.

[7] H. Kowshik and P. R. Kumar, "Optimal computation of symmetric boolean functions in collocated networks," *IEEE J. Sel. Areas Commun.*, vol. 31, no. 4, pp. 639–654, Mar. 2013.

[8] L. Ying, R. Srikant, and G. E. Dullerud, "Distributed symmetric function computation in noisy wireless sensor networks with binary data," in *Proc. WIOPT*, Apr. 2006, pp. 336–344.

[9] A. Bletsas, A. Khisti, D. P. Reed, and A. Lippman, "A simple cooperative diversity method based on network path selection," *IEEE J. Sel. Areas Commun.*, vol. 24, pp. 659–672, Mar. 2006.

[10] Q. Zhao and L. Tong, "Opportunistic carrier sensing for energy-efficient information retrieval in sensor networks," *EURASIP J. Wireless Commun. Netw.*, vol. 2005, pp. 231–241, Apr. 2005.

[11] W. B. Heinzelman, "An application-specific protocol architecture for wireless networks," Ph.D. dissertation, Massachusetts Inst. of Technology, Jun. 2000.

[12] V. Shah, N. B. Mehta, and R. Yim, "Optimal timer based selection schemes," *IEEE Trans. Commun.*, vol. 58, pp. 1814–1823, Jun. 2010.

[13] X. Qin and R. Berry, "Opportunistic splitting algorithms for wireless networks," in *Proc. INFOCOM*, Mar. 2004, pp. 1662–1672.

[14] V. Shah, N. B. Mehta, and R. Yim, "Splitting algorithms for fast relay selection: Generalizations, analysis, and a unified view," *IEEE Trans. Wireless Commun.*, vol. 9, no. 4, pp. 1525–1535, Apr. 2010.

[15] A. Papoulis, *Probability, Random Variables and Stochastic Processes*, 3rd ed. McGraw Hill, 1991.

[16] H. Kowshik and P. R. Kumar, "Zero-error function computation in sensor networks," in *Proc. IEEE Conf. Decision and Control (CDC)*, Dec. 2009, pp. 3787–3792.

[17] A. Anand and N. B. Mehta, "Quick, energy-efficient, and decentralized max function computation in wireless networks using timer-based selection," *To be submitted to IEEE Trans. Commun.*, 2014.