Abstract—We present a system level design methodology for clustered wireless sensor networks based on a semi-random communication protocol called SERAN, a mathematical model that allows to optimize the protocol parameters, and a network initialization and maintenance procedure. SERAN is a two-layer (routing and MAC) protocol. At both layers, SERAN combines a randomized and a deterministic approach. While the randomized component provides robustness over unreliable channels, the deterministic component avoid an explosion of packet collisions and allows our protocol to scale with network size. The combined result is a high reliability and major energy savings when dense clusters are used. Our solution is based on a mathematical model that characterizes performance accurately without resorting to extensive simulations. Thanks to this model, the user needs only to specify the application requirements in terms of end-to-end packet delay and packet loss probability, select the intended hardware platform, and the protocol parameters are set automatically to satisfy latency requirements and optimize for energy consumption.

Index Terms—Industrial wireless, wireless protocol, wireless sensor network.

I. INTRODUCTION

ALTHOUGH wireless sensor network (WSN) technology experienced great advancements in the last years, still its use in real-life applications is very limited. The main reason for the delay in the adoption is the lack of a system level approach, which is a design methodology that, given a set of application constraints, is able to synthesize a design solution that guarantees the required latency and quality of service under unreliable channel conditions. Particularly, it is not always clear what are the latency, reliability, and power consumption performance that the protocol stack offers to the application, and this is critical when control is involved [1].

Designing reliable communication protocols is a difficult task because in many applications, the environment is unpredictable. However, some applications have common characteristics that can be leveraged for an effective protocol design. For example, important classes of applications are characterized by clustered topologies. In building automation, groups of sensors are deployed in specific rooms to observe quantities as temperature, humidity, or chemical leakage and report to a remote central station in a multi-hop fashion. In manufacturing lines, sensors are typically grouped around specific points of interest in a manufacturing cell (i.e., the end of a rail or around some robots). From a network perspective, these are all clustered topologies, and although the size and the position of these clusters may vary significantly for different applications, this similarity allows us to create protocols that can be effective over all these applications.

In [2], we presented SERAN, a routing and MAC protocol for clustered WSNs that provides robustness to environment variability, is energy and storage efficient, can be implemented on a large set of existing hardware platforms, has self-configuration capabilities, supports the addition of new nodes, and can be extended with data aggregation algorithms. In that work, a procedure for the start up and the maintenance of the network that allows the system to react to major changes in topology conditions and clock drifts was also introduced. In this paper, we further extend the solution and improve the mathematical model to characterize the delay and power performance of SERAN given an initial topology estimate and traffic requirements without the need of extensive simulation. Consequently, we are now able to propose a complete system solution that, given the application specifications and a loose description of the topology, synthesizes the protocol parameters so that end-to-end latency requirements of the application are satisfied, energy consumption is minimized, and a procedure for the system to reach the optimal working point is offered. Our solution is different from previous approaches, where single hop performances were optimized and best effort solutions proposed. We do not introduce clustering algorithms because we assume, as it is the case of many practical applications, that the clusters are already formed and identified by the end user.

This is not the first attempt to provide a system level solution for WSN. Both in academia [3]–[5] and industry [6]–[9], several system solutions based on different communication protocols were proposed, and some are expected to be built around standardized low-power protocols such as Zigbee [10] or Bluetooth [11]. However, none of these solutions is based on a communication protocol that clearly leverages distinctive topology features. Despite existing relevant contributions in the literature, none of them presents a comprehensive protocol, which includes all the relevant characteristics of the physical layer, MAC, and routing, and which is able to guarantee latency performance and optimize for energy consumption. Our
Fig. 1. Connectivity graph. Nodes are grouped in clusters, and each node selects at random the node to which transmit within nodes in next cluster.

System level design methodology is an application of the platform-based design (PBD) [12], a methodology that advocates the combination of a top-down approach, where application requirements are refined into system constraints, a bottom up approach, where performance of the implementation options is characterized, and a *meet in the middle* phase where a solution is selected to satisfy constraints and optimize for a given cost function. Although initially developed for classical embedded systems, PBD was later extended to treat more general communication problems [13].

This paper is organized as follows: in Section II, the routing and MAC solution are presented. To motivate our protocol design, we discuss different alternatives at every step of the design flow and indicate how our solution is positioned with respect to previous work. In Section III, we present the mathematical model and show how to optimize the protocol parameters for power efficiency, while in Section IV, we present our initialization algorithm for network self-configuration and our approach to maintain network operations under adverse conditions. In Section V, we present testbed results on a case study, and in Section VI, we give some concluding remarks.

II. SERAN PROTOCOL

We present our protocol stack considering the topology of Fig. 1, where five clusters of sensors, with no cluster head, are deployed to sense data and report to a controller with a latency constraint \( D_{\text{max}} \).

A. Routing Algorithm

Routing over an unpredictable environment is notoriously hard. Our approach leverages density and clusterization and assumes that if there is a set of nodes within transmission range that could be candidate receivers, at least one of them will be likely to offer a good link anytime a transmission is needed. This assumption is reasonable because the spatial diversity found in wireless transmission can be exploited to give diversity gains [31].

In [26], the idea of deciding next hop after an estimation of the links to neighboring nodes is presented. Although the estimation algorithm has very good convergence properties, the protocol shows stress when applied to fast varying links. In [20], the idea of routing through a random sequence of hops instead of a predetermined one is introduced. In [19], the idea is further explored to reduce the overhead caused by the need of coordinating the nodes, and an algorithm is given for determining the optimal shape of the region from which candidate receivers should be selected.

In SERAN, the sender has knowledge of the cluster to which a packet will be forwarded, but the actual choice of forwarding node is made at random. This random choice is not performed at the network layer, but it is a result of an acknowledgment contention scheme performed at the MAC layer by all the candidate receivers (see next subsection).

Consider the cluster connectivity in Fig. 1. An arrow between two clusters means that all the nodes of the two clusters are within transmission range (not necessarily in line of sight). Assume a particular node in Cluster 1 has a packet to forward to the controller. Our cluster-based routing selects at random a node in Cluster 2, so that the node in Cluster 1 forwards the packet to it. The chosen node determines its next hop by choosing a node randomly in Cluster 4, and so on. In other words, packets are forwarded to a randomly chosen node within next-hop cluster in the minimum spanning tree to the controller. Notice that these operations are done without the need of a cluster head node within clusters.

B. Hybrid MAC

There are several MAC schemes proposed for clustered environments, most of them coupling the MAC with some clusterization algorithm and electing a single node to be the cluster leader and accumulating all the packets of the other nodes in the cluster [16], [17]. We decided to design a MAC where no single node is elected to accumulate all the packets and that is able to support the addition of new nodes for preserving the high level of density required to ensure robustness. This flexibility
is usually obtained by using CSMA-based access schemes that may or may not support collision avoidance, depending on the radio interfaces that are used and the capability of the RF chip to support an effective clear channel assessment (see, e.g., BMAC [28]). High density unfortunately introduces a large number of collisions, even if collision avoidance is supported. To reduce collisions, usually a deterministic MAC is used. A well-known deterministic approach is SMAC [15], where the network is organized in a clustered TDMA scheme.

Our MAC solution is based on a two-level semi-random communication scheme that provides more robustness to topology changes typical of a CSMA-based MAC and more robustness to collision typical of a deterministic MAC. The higher level regulates channel access among clusters. A weighted TDMA scheme is used such that at any point in time, only one cluster is transmitting and only one cluster is receiving. During a TDMA cycle, each cluster is allowed to transmit for a number of TDMA-slots that is proportional to the amount of traffic it has to forward. The introduction of this high-level TDMA structure has the goal of limiting interference between nodes transmitting from different clusters. The time granularity of this level is the TDMA-slot $S$.

The lower level regulates the communication between nodes of the transmitting cluster and nodes of the receiving cluster within a single TDMA-slot. It has to support the semi-random routing protocol presented in II-A, and it has to offer flexibility for the introduction of new nodes. This flexibility is obtained by having the transmitting nodes access the channel in a $p$-persistent CSMA fashion [23]. If collision avoidance (CA) is supported by the hardware platform, it can be used to improve performance. The random selection of the receiving node is obtained by broadcasting the packet over all nodes of the receiving cluster and by implementing in the receiving nodes a random acknowledgment contention scheme to prevent duplication of packets. Assuming that nodes in Cluster 1 transmit to nodes in Cluster 2, the protocol can be summarized as described in the following three steps.

1) Each node of Cluster 1 having a packet tries to multicast the packet to nodes of Cluster 2 at the first CSMA-slot with probability $p$. If CA is supported, a clear channel assessment (CCA) for a random back-off time is performed before the transmission, and in case another transmission is detected, the node aborts the current trial to avoid collisions. If CA is not supported, the node simply transmits the packet.

2) At Cluster 2, if a node receives simultaneously more than one packet, it detects a collision and discards all of them. If it has successfully received a single packet, it starts a back-off time $T_{\text{ack}}$ before transmitting an acknowledgment. The back-off time $T_{\text{ack}}$ is a random variable uniformly distributed between 0 and a maximum value called $T_{\text{ack, max}}$. If in the interval between 0 and $T_{\text{ack}}$ the node hears an acknowledgment coming from another node of Cluster 2, the node discards the packet and does not send the acknowledgment. Note that this random back-off procedure is different from a CA procedure. This is because nodes are already awake and listening to the channel for possible packets, and consequently, such a scheme can be implemented, even on platforms where performing instantaneous CCA is not supported or it is inefficient. In Section III-F, we explain how to deal with inefficiencies in this acknowledgment contention scheme.

3) At the transmitting node of Cluster 1, if no acknowledgment is received (or if only colliding acknowledgments are detected), the node assumes the packet transmission was not successful and it multicasts the packet at next CSMA-slot again with probability $p$. The procedure is repeated until transmission succeeds or the TDMA slots ends.

With the approach outlined with previous steps, nodes need to be awake only of next-hop cluster connectivity and do not need a neighbor list of next-hop nodes. We believe this is a great benefit because, while neighbor lists of nodes are usually time-varying (nodes may run out of power and other nodes may be added) and, hence, their management requires significant overhead, cluster-based connectivity is much more stable. In Section IV, we explain how to deal with permanent fades between clusters within transmission range.

In most of the proposed MAC algorithms for WSN, nodes are turned off whenever their presence is not essential for the network to be operational. GAF [14], SPAN [21], and S-MAC [15] focus on controlling the effective network topology by selecting a connected set of nodes to be active and turning off the rest of the nodes. These approaches require nodes to maintain partial knowledge of the state of their individual neighbors, thus requiring additional communication. Our duty-cycling algorithm leverages the MAC properties and does not require extra communication among nodes. During an entire TDMA cycle, a node has to be awake only when it is in its listening TDMA-slot or when it has a packet to send and it is in its transmitting TDMA-slot. For the remainder of the TDMA cycle, the node radio can be turned off.

### C. Organization of the TDMA-Cycle

Because of the proposed converge-cast routing solution, clusters close to the controller have a larger traffic load since they need to forward packets generated within the cluster as well as packets coming from upstream clusters. Assuming in the example of Fig. 1 that the average traffic generated at each cluster is the same, the average traffic intensity that Cluster 4 experiences is three times the traffic intensity experienced by Cluster 1. Consequently, we can assign one transmitting TDMA-slot per TDMA-cycle to Cluster 1, two transmitting TDMA-slots to Cluster 2, and three transmitting TDMA-slots to Cluster 4. In a similar fashion, on the other path, the number of associated TDMA-slots per cluster can be assigned. Therefore, assuming we have $P$ paths and calling $B_i$ the number of clusters in the $i$th path, we have a total of

\[ T_f = \sum_{i=1}^{P} \frac{B_i(B_i + 1)}{2} \]  

TDMA-slots per TDMA-cycle. For the remainder of the paper, we call $T_f$ the topology factor. As we will see later, $T_f$ is an important parameter that abstracts the network layout and connectivity.

Notice that in case the traffic generated is not the same for each cluster, the relative number of TDMA-slots per TDMA-
cycle for each cluster can be easily recalculated, changing the weights in the TDMA scheme. For the sake of simplicity, we outline our solution for a case with uniform traffic rate. The extension to a more generic traffic pattern is straightforward.

Once we decide the number of TDMA-slots per TDMA-cycle for each cluster, we need to decide the scheduling policy for transmitting and receiving. We select an interleaved schedule (see Fig. 2). For each path, the first cluster to transmit is the closest to the controller (Cluster 4), then Cluster 2 and Cluster 4 again, then Clusters 1, 2, and 4, and similarly on the other path. This scheduling is based on the idea that evacuating the clusters closer to the controller first, we minimize the storage requirement throughout the network.

III. PROTOCOL PARAMETER DETERMINATION

In this section, we explain how the access probability \( p \) and slot duration \( S \) are determined to satisfy application requirements (maximum delay \( D_{\text{max}} \)) and optimize for power consumption. First, we show how to set the access probability parameters, and then, we set the duration of a TDMA-slot to offer good latency performance and optimize for power consumption. First, we carry on this analysis for the case in which collision avoidance is not supported, and then, we show how the model is modified to account for collision avoidance.

A. Access Probability

Here we would like to determine the access probability that each node should use in order to minimize the time to evacuate a cluster. In particular, such a time is then related to the duration of the TDMA-slot.

Call \( k \) the number of packets that the cluster has to evacuate at the beginning of a transmitting TDMA-slot. We consider the worst-case scenario for collisions; that is when \( k \) packets are distributed over \( k \) different nodes. We abstract the channel behavior by a Bernoulli random variable with parameter \( c \). Notice that this parameter is close to 1 since it abstracts the spatial diversity gain that due to the cluster-based connectivity: \( c \) is the probability that at least one node in forwarding cluster is able to complete a successful communication. Indeed, we assume that appropriate channel and source coding are applied by each node, so that the successful packet loss probability can be met. These are natural assumptions in many WSN applications [29].

When there are \( k \) packets to be forwarded, the probability of having a successful transmission at the first CSMA-slot is

\[
\Pr[\text{success} | k] = c(k) p(1 - p)^{k-1}
\]

Note that this model is conservative and does not account for the “lucky” event of more than one packet transmitted and only a single one successfully received.

Assume the transmission was successful. The cluster now has \( k - 1 \) packets to forward. This time, the probability of a successful transmission at the first CSMA-slot is

\[
\Pr[\text{success} | k - 1] = c(k - 1) p(1 - p)^{k-2}
\]

Again, if the transmission was successful, the cluster has \( k - 2 \) packets to forward, and so on. This allows us to represent the cluster behavior as a discrete time Markov chain (DTMC) where the state is the number of packets that still need to be forwarded (see Fig. 3). The DTMC has an absorbing state in 0, which is the steady-state solution of the chain. This means that the state 0 is eventually reached with probability one. We are interested in calculating the expected time (i.e., expected number of steps) to reach the absorbing state starting from a given state between 1 and \( k \). This is equivalent to the expected number of steps to get from state 1 to state 0, which we denote by \( \lambda_{1,0} \).
to determining the average number of CSMA-slots required for forwarding a number of packets between 1 and \( k \). Since expectation is a linear operator and using the fact that the chain can advance only one step at a time, the expected time to absorption starting from a state \( k \) is equivalent to the sum of the expected time to transition from state \( k \) to state \( k - 1 \) plus the expected time to transition from state \( k - 1 \) to \( k - 2 \) and so on until state 0 is reached. Given that the chain is in state \( j \), the mass distribution of the required number of steps to transition to state \( j - 1 \) follows a geometric distribution of parameter \( 1 - c \eta(j)(1 - p)^{j-1} \). Consequently, the expected time to transition from state \( j \) to state \( j - 1 \) is

\[ \tau(j) = \frac{1}{c \eta(j)(1 - p)^{j-1}}. \]

Calling \( \tau_k \) the expected number of steps to reach the absorption starting from state \( k \), we have

\[ \tau_k = \sum_{j=1}^{k} \tau(j) = \sum_{j=1}^{k} \frac{1}{c \eta(j)(1 - p)^{j-1}}. \]

Considering (2), we notice that the access probability that minimizes the transition time from state \( j \) to \( j - 1 \) is \( p = p_j = 1/j \), meaning that a node in the state \( j \) chooses an access probability given by \( 1/j \). If each node would use the probability \( p_j \) when the system is in state \( j \), the expected number of transmission attempts for each slot would be exactly one. This is the choice that maximizes channel utilization without incurring into excessive collisions. However, this is not the choice of the access probability minimizing the overall (2). Therefore, we now present two strategies for setting up the access probability given the number of packets that need to be transmitted at the beginning of the TDMA-slot. In the following subsections, we present the latency and energy performance of the two strategies, and in Section III-E, we present a comparison of them.

1) Fixed Choice: According to this choice, the access probability is the same for each node, and it remains the same during the whole TDMA-slot duration.

Finding a closed-form expression of the access probability \( p \) that minimizes \( \tau_k \) in (2) is a nontrivial problem. However, the expression is a convex function in \( p \). Indeed, (2) is a nonnegative weighted sum of the functions \( 1/(c \eta(j)(1 - p)^{j-1}) \). These functions are convex, since by taking the first derivative, there is only a critical point in the interval \([0,1]\), which is \( p_j = 1/j \), and the second derivative is strictly positive.

Although (2) is a convex function, its first derivative does not help to compute a closed-form expression of the value of \( p \) that minimizes (2). The convexity allows us to use the bisection algorithm [27], which finds iteratively the numerical value minimizing (2) with any desired precision. Note that the algorithm is not computational demanding and can be easily implemented on sensor nodes. If the initial guess used to feed the algorithm is good, the convergence to the optimal value minimizing (2) is very fast.

However, it can be proved that such an optimal selection of \( p \) would be larger than \( 1/k \). Since the most critical stage in our DTMC model, in terms of collision probability, is from state \( k \) to \( k - 1 \), such an access probability would likely lead to a large number of collisions at the beginning of the TDMA-slot. Consequently, we select the access probability \( p = 1/k \) for the whole duration of the slot, which is suboptimal in terms of expected forwarding time, but it ensures that at the beginning of the TDMA-slot, the expected number of transmission attempts for each CSMA-slot is 1. The result is that the channel is highly utilized, while as time progresses, the channel will be less and less utilized. The expected absorption time is

\[ \tau_k = \frac{k}{c} \sum_{j=1}^{k} \frac{1}{j(1 - \frac{1}{k})^{j-1}}. \]

In this scenario, a simple relation between \( \tau_k \) and \( k \) is not easy to find. Nevertheless, it is important to have knowledge of it in order to minimize the evacuation time and relate it to the slot duration. We can find some useful upper and lower bounds.

**Proposition 1:** For the fixed choice, i.e., \( p = 1/k \), for large \( k \), the expected time to forward all the packets is bounded by:

\[ \alpha_{fb} k \leq \tau_k \leq \alpha_{fa} k \ln k, \]

where \( \alpha_{fb} \) and \( \alpha_{fa} \) are positive constants.

**Proof:** Looking at (3), we notice that a lower bound is given by the case in which all the expected transition times are the same as the expected transition time of the first transition (when the channel is optimally utilized). This expected transition time is \( 1/c(1 - 1/k)^{k-1} \), and since \( \lim_{k \to \infty} (1 - 1/k)^k = e^{-1} \), we can find a lower bound \( \tau_{LBL} = (e/c)k \).

The upper bound can be found considering that

\[ \tau_k \leq \frac{k}{c} \left( \frac{k}{k-1} \right)^{k-1} \sum_{j=1}^{k} \frac{1}{j} \leq \frac{e}{c} \left( k - 1 \right) \sum_{j=1}^{k} \frac{1}{j}. \]

The \( k \)th harmonic \( H_k = \sum_{j=1}^{k} 1/j \) grows as fast as \( \ln k \), and it is upper bounded by \( H_k < 1 \ln k \). Consequently, defining any \( \gamma > 1 \) for large \( k \), we have

\[ \tau_k \leq (k - 1)(1 + \ln k) \leq \frac{e}{c} k(1 + \ln k) \leq \frac{e}{c} k \gamma \ln k. \]

Because of the interleaved schedule, each cluster evacuates all locally generated packets before receiving those generated from the one-hop upstream cluster. First, we need to ensure that the expected time for the evacuation of packets in a cluster is less than or equal to the duration of a TDMA-slot. If this does not happen, packets keep accumulating and storage capacity is reached very soon with catastrophic consequences on performance.

We consider the upper bound for the forwarding time, so we can simplify our analysis. As we show in Fig. 4, this is already a good enough upper bound, that is \( \tau_k = (e/c)k \ln k \). Let us denote with \( \Delta \) the duration of a TDMA-cycle and with \( \lambda \) the packet generation rate for each cluster. Since during a TDMA-cycle each cluster generates \( \Delta \) packets, we need to ensure that the TDMA-slot duration is

\[ S > \frac{e}{c} \Delta \ln(\lambda \Delta). \]

Recalling the expression of \( T_f \) in (1), and that \( \Delta = ST_f \), we can simplify previous equation as

\[ S < S_{\text{max}} \Rightarrow \frac{e}{c} = \frac{eT_f(\lambda \Delta)}{X T_f}. \]
which, given a traffic generation $\lambda$, sets a constraint on the maximum duration of a TDMA-slot. Notice that $S_{\text{max}}$ is the maximum TDMA-slot duration due to traffic. As it will be clearer in Section III-E, it is interesting to rewrite previous equation as

$$\lambda \ln(\lambda S T_f) < \frac{c}{c T_f}.$$  

(5)

2) Adaptive Choice: According to this choice, the access probability is increased every time there is a state transition in such way that for each transition from state $j$ to $j - 1$, it goes from $1/j$ to $1/(j - 1)$. Recall that this is the choice that minimizes the forwarding time and, hence, maximizes the throughput of the cluster. The expected time to forward all packets in this case is

$$\tau_k = \frac{1}{c} \sum_{j=1}^{k} \left(1 - \frac{1}{j}\right)^{j+1}.$$  

(6)

Proposition 2: For the adaptive choice, for large $k$, the expected time to forward all packets is bounded by: $\alpha_{adv} k \leq \tau_k \leq \alpha_{adv} k$, where $\alpha_{adv}$ and $\alpha_{adv}$ are positive constants.

Proof: The upper bound can be found as for the similar case in the proof of Proposition 1, so that for large values of $k$, we have $\tau_{U/B} = (c/e) k$.

A lower bound can be found considering a successful transition at every CSMA-slot. This means $\tau_{L/B} = k$.

Considering the upper bound from the previous proposition, we can now derive some design constraints in the same way as we did in the fixed choice case

$$S \geq \frac{c}{c T_f} \lambda \Delta.$$  

Since $\Delta = ST_f$, we can obtain a limit for the maximum sustainable traffic

$$\lambda \leq \frac{c}{c T_f}.$$  

(7)

Notice that, in this case, we get a constraint only on the traffic generation rate, and it depends on the topology and connectivity of the network, abstracted by the topology factor $T_f$, and not on the TDMA-slot duration, as in (5). Note also that, given a number of clusters, the configuration that minimizes $T_f$ (and maximizes the maximum sustainable traffic) is a star topology, where each cluster is a single hop to the controller. Conversely, the worst configuration is a linear topology, where all the clusters are in a single multi-hop chain.

In case the maximum traffic condition is not satisfied even using the adaptive choice, a slot reuse mechanism can be introduced to obtain an operational network. This means to have more than one cluster transmitting and receiving during the same time-slot, provided that they are far enough apart. When the system is facing scalability issues, and this can be abstracted by a large $T_f$, slot reuse is an important mean to cope with it. This solution can significantly increase the throughput of the network, but it is also much more energy expensive.
Consequently, it should be considered only if the stability requirement cannot be satisfied; otherwise, a “lazy” network is preferable.

B. Latency

The clusters experiencing the largest delay are the furthest from the controller. We want to have the delay of packets coming from those clusters less than or equal to a given $D_{\text{max}}$, the requirement set by the application.

Consider the packets generated in Cluster 1. These packets have to wait, in the worst case, a TDMA-cycle before the first opportunity to be forwarded to Cluster 2. Assuming for now that all the packets of a cluster are forwarded within a single TDMA-slot, then it takes three additional TDMA-slots to reach the controller. Generalizing to the case of $P$ paths and $B_i$ clusters per path, the worst case delay is

$$D = \Delta + S \max_{i=1}^{P} B_i = S(B + T_f)$$

where $B = \max_{i=1}^{P} B_i$. Consequently, the requirement on $S$ is

$$S \leq S_{\text{max}} = \frac{D_{\text{max}}}{B + T_f}$$

(8)

where $S_{\text{max}}$ is the maximum TDMA-slot duration due to latency. If during a TDMA-slot not all the packets are forwarded, a latency over the deadline is observed. We can model this phenomenon using the DTMC model introduced in Section III-A. We want to evaluate the probability that the time to forward $\lambda\Delta$ packets exceeds the duration of a TDMA-slot. Using the Central Limit Theorem, we model the distribution of the time to forward $\lambda\Delta$ packets as a Gaussian variable whose mean and variance is given by the sum of the expected times and variances to advance in a step in the chain. Call $T_{\text{ev}}$, the time to evacuate $\lambda\Delta$ packets, and call $m_{\text{ev}}$ and $\text{var}_{\text{ev}}$ its mean and variance. Consequently, the time $T_{\text{ev}}$ to evacuate a cluster can be modeled as $T_{\text{ev}} \in \mathcal{N}(m_{\text{ev}}, \text{var}_{\text{ev}})$, where in case there is no collision avoidance, we have

$$m_{\text{ev}} = m_{\text{ev,ca}} = \frac{\lambda\Delta}{c p j (1 - p)^{j-1}}$$

$$\text{var}_{\text{ev}} = \text{var}_{\text{ev,ca}} = \frac{\lambda\Delta}{c p j (1 - p)^{j-1}} - \frac{1}{[1 - c p j (1 - p)^{j-1}]^2}.$$ 

Therefore, the probability of not forwarding all the packets during a given TDMA-slot, which we define as the outage probability of packets, can be approximated by

$$\Pr[T_{\text{ev}} \geq S] \approx \frac{1}{2} \text{erfc} \left( \frac{S - m_{\text{ev}}}{\sqrt{\text{var}_{\text{ev}}}} \right)$$

(9)

where $\text{erfc}(\cdot)$ is the complementary error function. Although it is not possible to find a closed-form solution to (9), the requirement expressed in (8) usually ensures an outage probability well below 5%, as we will show in Section V.

C. Energy Consumption

We are now interested in determining the total energy consumed by the network over a period of time. The energy cost is given by the contribution of the energy spent for transmissions $E_{T_f}$, the energy spent to wake up and listen during the listening cluster-slots $R$, and the energy spent for the clear channel assessment procedure $E_{\text{CCA}}$ in case collision avoidance is supported. We consider the energy spent for receiving a packet together with the energy consumption for listening. The energy consumption for listening for a time $\delta$ is given by the sum of a fixed cost (the wake-up cost $R$) plus a time-dependent cost (listening cost power $W$): $E_{\text{lw}} = R + W\delta$.

During a TDMA-cycle, nodes in Cluster 1 never wake up for listening, nodes in Cluster 2 wake up once for listening, nodes in Cluster 4 wake up twice for listening, and so on (see Fig. 2). Assume that there are $N$ nodes per cluster and that all nodes wake up in their listening TDMA-slot. During a given TDMA-cycle, the total number of wake ups is

$$N_{\text{wu}} = N \sum_{i=1}^{P} \frac{B_i (B_i - 1)}{2}.$$ 

(10)

To determine the energy spent for transmissions, we need to derive the average number of attempted packet transmissions during a TDMA-cycle. In case collision avoidance is not supported, we can use the DTMC introduced in Section III-A.

Proposition 3: For large values of the number of packets accumulated in a cluster during a TDMA-cycle (i.e., the expected number of attempted transmission is a linear function with the respect to the number of packets to transmit.

Proof: We model the number of attempted transmissions for each transition as the average number of nodes attempting to transmit during a slot multiplied by the average number of slots required for that transition. Assuming $k$ packets to forward, the average number of attempted transmission during a TDMA-cycle can be modeled as

$$N_{T_{\text{ev,ca}}} = \left( \frac{\text{numslots}}{\text{cycle}} \right) \sum_{j=1}^{k} \frac{p j}{c p j (1 - p)^{j-1}}.$$ 

(11)

where $(\text{numslots/\text{cycle}})$ is the number of TDMA slots in a TDMA cycle (see Section II-B). We then have the following cases.

1. Case Fixed Choice. Since $p = 1/k$, (11) can be simplified as

$$N_{T_{\text{ev,ca}}} = T_f \frac{1}{k} (k - 1) \left[ \left( 1 - \frac{1}{k} \right)^{k-1} - 1 \right].$$ 

For large values of $k$, $\left(1 - 1/k\right)^{k-1} \approx (e - 1)$ and

$$N_{T_{\text{ev,ca}}} = T_f (e - 1) \lambda\Delta.$$ 

(12)

2. Case Adaptive Choice. In this case, recall that $p_j = 1/j$, so that the product $p_j$ at the numerator of (11) is always equal to 1, and the expected number of attempted transmission is equal to the expected number of steps required to forward
all the packets. In Section III-A-II, we showed that this number is a linear function on the number of initial packets whose slope is between 1 and \( e/c \). Consequently, we can write \( N_{txNca} = A_{nca} \lambda \Delta \).

The number of acknowledgment transmissions is equal to the number of successful packets \( N_{ack} = T_f \lambda \Delta \). Recalling that \( E_{Tx} \) is the energy consumption for the transmission of a packet and \( E_{Ack} \) is the energy consumption for the transmission of an acknowledgment, the total energy consumption during a time \( T \gg \Delta \) for the noncollision avoidance case is

\[
E_{tot}(S) = \frac{T}{\Delta} [N_{txNca} E_{Tx} + N_{Ack} E_{Ack} + N_{txNca} R + N_{txNca} W]
= T A_{nca} \lambda E_{Tx} + T X T_f E_{Ack} + \frac{T N_{txNca} R}{T_f}
+ \frac{T N_{txNca} W}{T_f}.
\]  

(13)

Notice that \( E_{Tx}, E_{Ack}, R, \) and \( W \) are parameters that characterize the physical layer, \( \lambda, N \) are given by the application, and \( T_f, N_{txNca} \) depend only on the network topology, so that the only variable in (13) is \( S \). Since, from (5) and (8), we have \( S \leq S_{\text{max}} = \min \{ S_{\text{trans}}, S_{\text{trans}} + d \} \), \( E_{tot}(S) \) is a monotonically decreasing function of \( S \), the optimal working point is \( S = S_{\text{max}} \).

D. Impact of Collision Avoidance

1) Maximum Sustainable Traffic With CA: Once again, the behavior of the transmitting cluster can be characterized by a DTMC whose state is given by the number of packets that remain to be forwarded. Differently from the non-CA case, when collision avoidance is used, the transition probability from a state \( j \) to a state \( j - 1 \) is larger. This results from the fact that even if two or more nodes decide to transmit, the CA procedure is likely to avoid collisions and allows at least one packet to be transmitted successfully.

Although very small, the probability of a collision is still nonzero, and it is associated to a failure of the CA mechanism. For instance, if TinyOS [18] is used to program the hardware platform, such a failure may happen if between the posting and the execution of a sending task of a node, another node starts its transmission. We call \( \phi \) the probability of such a failure when two nodes are involved. Consequently, the transition probability from state \( j \) to \( j - 1 \) in a given CSMA-slot can be modeled by the probability of having at least one node trying to access the channel multiplied by the probability that other nodes do not interfere with the first that transmits, namely,

\[
\text{Pr}\{\text{Success}[j]\} = e^{[1-(1-p)^j] - (1-p)^{j-1}}
\]

Following the same steps as the non-CA case, we can model the average time to empty the cluster using

\[
\tau_k = \frac{1}{k} \sum_{j=1}^{k} \frac{1}{e^{[1-(1-p)^j] - (1-p)^{j-1}}}
\]

(14)

In case the fixed choice is selected, the access probability is \( 1/k \). Using the same reasoning as in the non-CA case, we can find an upper and lower bound for \( \tau_k \). The lower bound becomes

\[
\tau_{kLB} = \frac{ek}{c e^{-\phi}(e-1)}
\]

whereas the upper bound does not improve with respect to the non-CA case. Consequently, the constraint on maximum sustainable traffic and duration of a TDMA-slot remains the same.

In case the adaptive choice is selected, the upper bound becomes \( \tau_{kUB} = (e/c)[1-(1-p)(e-1)]k \), whereas the lower bound remains the same. As a consequence, the constraint on the maximum sustainable traffic is slightly relaxed

\[
\lambda_{\text{max}} \leq \frac{c(1-\phi)(e-1)}{eT_f}.
\]

2) Delay With CA: The constraint on the maximum duration of the TDMA-slot does not change. What changes is the mean and standard deviation of the Gaussian distribution that abstracts the distribution of the time to empty a cluster. Specifically, we have that

\[
m_{ev} = m_{evca} = \sum_{j=1}^{\lambda\Delta} \frac{1}{c[1-(1-p)^j](1-\phi)^{j-1}}
\]

\[
\text{var}_{ev} = \text{var}_{evca} = \sum_{j=1}^{\lambda\Delta} \frac{1}{c[1-(1-p)^j](1-\phi)^{j-1}}
\]

3) Energy Consumption With CA: The difference with respect to the non-CA case is only in the number of attempted transmissions and in the number of clear channel assessments. Ignoring the collision events, the number of attempted transmissions can be easily modeled with the number of successful transmissions \( N_{txNca} = T_f \lambda \Delta \).

Modeling the number of channel assessments is similar to modeling the number of attempted transmissions when collision avoidance is not used, and a similar reasoning may be used involving the manipulation of the relative DTMC. However, a more simple model can be obtained, neglecting the collisions and considering the average number of transmission tries for each step. Consequently, we can write

\[
N_{cca} = \frac{T_f}{c} \sum_{j=1}^{\lambda\Delta} pj.
\]

(15)

Equation (15) becomes \( N_{cca} = (T_f/2c)(\lambda\Delta + 1) = (T_f/2c)\lambda \Delta \) in case of fixed choice and \( N_{cca} = (T_f/c)\Delta \) in case of adaptive choice. In any case, we can model the expected number of clear channel assessments as a linear function \( N_{cca} = A_{cca} \lambda \Delta \). Calling \( t \) the fixed duration of a CSMA-slot, we have

\[
E_{tot}(S) = \frac{T}{\Delta} \left[N_{txNca} E_{Tx} + N_{Ack} E_{Ack} + N_{txNca} (R + W)\right]
+ \frac{T N_{txNca} R}{T_f}
+ \frac{T N_{txNca} W}{T_f}
+ T A_{nca} (R + W t)
\]

(16)
Also in this case, we see that the energy consumption is a monotonically decreasing function of $S$; hence, the optimal working point is $S = S_{\text{max}}$.

E. Comparison of Access Strategies

Comparing the traffic constraints in (5) and (7), it can be seen that the constraint relative to the fixed choice is more stringent. One way to interpret these results is that in a network, there is a limit on the sustainable traffic that is given by the network topology and represented by the topology factor $T_f$. Furthermore, if the fixed choice is selected, the constraint becomes more stringent as the TDMA-slot increases. Consequently, given a traffic to support, the selection of the fixed choice may limit the capability of extending the duration of the TDMA-slot (unless the constraint imposed by the latency requirement is the most stringent). As we showed in Section III-C, this has a reverse impact on the power performance of the overall solution.

The adaptive choice is more efficient, and it allows for a larger throughput. However, such a strategy is more difficult to implement in a distributed fashion because nodes may not be aware of the fact that other nodes completed a successful transmission, and there is no way to tell them without incurring major overhead costs. The best way to implement this strategy is to have each node automatically update its access probability, evaluating the expected time to complete a transition in the chain. To do this, the node must be able to compute each term of the summation in (2). Failure to compute those fractions or lack of synchronization among the nodes may have a reverse impact on the efficiency of the solution and create either too many accesses, hence having more collisions, or too few accesses, hence wasting bandwidth. In the mathematical analysis, we did not consider these events.

Since we decided to set the access probability in such a way that the expected number of attempted transmissions for a CSMA-slot is at most 1, collision avoidance procedures do not improve performance dramatically. However, the greatest benefit is given by the extra robustness against inefficiencies in the implementation of the adaptive choice. This is a result of the fact that collision avoidance notoriously helps stabilizing CSMA protocols when bandwidth utilization approaches the limit.

For all these reasons, we recommend to use the adaptive choice only when the fixed choice is not good enough to serve the application requirements, and the selected hardware platforms support an effective collision avoidance.

F. Optimizing the Protocol

In Section II-A, we mentioned the problem of duplicate packets that can happen in our multicast scheme. This phenomenon can be simply modeled by introducing a variable $\nu$ that represents the probability of having a duplicate packet in each transmission. To consider this effect, we just need to substitute $\lambda$ with $\lambda(1 + \nu)$ in the previous equations. In [19], our acknowledgment contention scheme is proved to reduce $\nu$ to 0.1.

Further power savings can be obtained having only a subset of nodes per cluster waking up for their listening duty. The savings come from three factors: 1) The impact of the energy consumption due to listening decreases. 2) If the packets are forwarded to a smaller number of nodes, then the number of collisions in the following transmitting TDMA-slot is reduced. Assume only $M$ out of $N$ nodes wake up. Then, the number of attempted transmissions is no longer a constant but now is a monotonically decreasing function of $M$. 3) Since only a few nodes are accumulating upstream packets, it is possible to implement efficient data-aggregation algorithms.

As already mentioned, nodes closer to the data collector have a higher workload. As a consequence, these nodes would be subject to early energy depletion with catastrophic consequences for the network lifetime. This problem is typical of single sink networks and not specifically related to our solution. The best way to deal with this issue is implementing some sort of packet aggregation algorithm. Because of its modularity, SERAN can be extended and integrated with existing packet aggregation algorithms. Note that they would affect the construction of the TDMA schedule and not other parts of SERAN.

Since having a packet aggregation procedure decreases the increment of traffic for clusters closer to the data collector, the number of TDMA-slots dedicated to those cluster decreases as well, making the design of the final SERAN solution even simpler. Furthermore, a reduced number of TDMA-slots per TDMA-cycle will increase the maximum sustainable traffic for that topology. Having more nodes awake ensures robustness against fades, and, if the number of nodes per cluster is large enough, this extra optimization can be explored. Assume we need to wake up an average of $M$ out of $N$ nodes, an efficient and distributed implementation is obtained by having each node waking up at the beginning of its listening TDMA-slot with probability $M/N$. In [14], [15], [19], and [21], alternative solutions are proposed to obtain this level of optimization. The flexibility of SERAN allows once more the integration of those techniques.

IV. OPERATION OF THE NETWORK

In this section, we introduce a procedure that allows the network to initialize and self-configure to the optimal working point calculated in Section III, ensures robustness against clock drift of the nodes, and allows for the addition of new nodes.

In SERAN, a token is a particular message that carries information on the duration of a TDMA-slot and TDMA-cycle ($S$ and $\Delta$), the transmitting and receiving schedule of a TDMA-cycle, the transmitting probabilities, and a synchronization message carrying the current execution state of the TDMA-cycle. Note that once information on cluster location is given to the controller, the controller has all information to calculate the optimal set of parameters as in Section III. Consequently, the controller is able to generate a token before the network starts operating. Notice also that once a node receives a token message, it has all the necessary information to work properly.

When the network starts working, each node is awake and listening. The node remains in this state and cannot transmit until it receives a token. The first transmission comes from the controller, which multicasts a token to all the nodes of one of the connected cluster. In our example, assume the selected cluster is
Cluster 4. Nodes of Cluster 4 read information on scheduling and duration of TDMA-slot and TDMA-cycle. Assume the scheduling is the one in Fig. 2. Nodes of Cluster 4 start transmitting their packets to the controller with the modalities indicated in the token. At the end of the TDMA-slot, all the nodes of Cluster 4 listen to the channel and start a random back-off counter. When the counter expires, if no other node sent a token, they broadcast it. Nodes in Cluster 2 see the token and start behaving according to the scheduling algorithm. After they transmit their packets to Cluster 4, one of them broadcasts a token so that nodes in Cluster 1 can hear it. After the first branch of the routing tree is explored, the controller sends a token to Cluster 5, the new branch is explored, and so on. The token passing procedure continues, even in the following TDMA-cycles.

The routing solution described in Section II-A is designed to cope with fast time-varying channels and not with permanent fades between clusters. This is a case when, e.g., a metal object is interposed for a long time between two clusters, hence cutting off their communication. This phenomenon is detected by the controller that does not receive packets (or receives too little) coming from a particular cluster (or set of clusters). When this event occurs, the controller recomputes a minimum spanning tree, without considering the corrupted link and generates a new scheduling and protocol parameters, then for a limited number of TDMA-cycles (typically from 2 to 5), it sends a token with a message to void the current scheduling, and finally, it reinitializes the network sending a token with the new optimal parameters. This re-initialization happens in general more often at the beginning of the network life-cycle, but once the corrupted links are detected, it is less and less frequent.

In [2], we have shown that clock drift conditions that were an order of magnitude worse than the ones reported in literature for off-the-shelf platforms do not influence the protocol stability.

V. CASE STUDY

Industrial monitoring applications offer a great opportunity to exploit natural clusterization and node density [30]. We consider a case study of a typical robot monitoring and maintenance application. Sensors are placed around robots in a manufacturing line to sense vibrations, and data must be delivered to a controller with a latency constraint. The controller is usually placed to sense vibrations, and data must be delivered to a controller with a latency constraint. The controller is usually placed close to the manufacturing line.

We recreated the topology of Fig. 1 in a testbed environment, where five MICA2dot motes [22] are placed in each cluster, and “disconnected” clusters are separated by metal walls. The main goal of our validation is to test the validity of our mathematical analysis. In particular, prove that the analysis of Section III drives to a solution that satisfies latency constraints and minimizes power consumption.

We considered a latency constraint on the end-to-end delay of $D_{\text{max}} = 190$ s. This value was selected to allow for an implementation over the available platform and generate interesting synthesis scenarios. We implemented the power saving techniques described in Section III-F. Specifically, nodes wake up for listening with probability $2/5$, and if a node has in its buffer packets generated by the same cluster, it calculates the average of the data and forwards a single packet. We abstracted the physical layer using a CSMA-slot duration $t = 0.1$ s, which we assume to be enough for two nodes to exchange a packet and an acknowledgment. We consider two scenarios. In the first one, we selected a packet generation rate per cluster $\lambda_1 = 1 \text{pkt/5s}$, and for the second, we have $\lambda_2 = 1 \text{pkt/10s}$. In both cases, we used the fixed choice for the channel access probability calculating the access probability $p$ as explained in Section III-A-I, and we do not consider collision avoidance (notice that this means to turn off the collision avoidance mechanism present in the common distribution of TinyOS). In Table I, we summarize the synthesized parameters. During initialization, all nodes were operational after the first TDMA-cycle. The introduction of a permanent fade between Cluster 2 and Cluster 5 forced the minimum spanning tree to the shortest path tree of Fig. 1. The re-initialization of the network was successful after two TDMA-cycles in both scenarios. For the first scenario, since the most stringent constraint was due to the traffic sustainability [$S_{\text{max}} = \{4\}$], the solution was fast, and the delay constraint always satisfied. In the second scenario, the most stringent constraint was due to the delay requirement, and more interesting results were obtained.

As it is shown in Fig. 5, the optimal solution offered an outage probability around 2%. At this working point, we observed an average node duty-cycle around 1.4% that would project a network lifetime of several months. As it can be seen in Fig. 6, better power performance can be obtained having longer TDMA-slot, but this would have the side effect of entering a region of exponential growth of the outage probability. For this reason, we consider the optimal calculated solution as the one that offers the best tradeoff. The reason for the perfect match of the results is that a CSMA-slot duration of 100 ms is enough for a complete packet acknowledgment exchange, and no unexpected problems appeared. When trying to replicate the same example using a much smaller CSMA-slot duration (less than 50 ms), we noticed a high level of unacknowledged packets that results in a network instability. We believe that a CSMA-slot duration of 50 ms is the limit of the proposed solution when implemented over such a node platform.

VI. CONCLUSION AND FUTURE WORK

We presented a system level approach for the design of wireless sensor networks in clustered environments. We believe clustered topologies abstracts very important application classes, and the capability of leveraging their characteristics as well as providing a methodology to effectively design these systems is what distinguishes our approach from the previous system level solutions. Our solution is based on a semi-random communication protocol called SERAN, a mathematical model that describes the performance and tradeoffs of the protocol as well
as an algorithm to initialize and maintain the network. We validated our mathematical model with some testbed experiments. Future work includes performing a large set of experiments with a wide experimental evaluation of the SERAN protocol.
in several scenarios of traffic load, node clustering, cluster size, channel conditions, and performance requirements. Furthermore, we will include in the framework aggregation algorithms to increase the effective throughput at no power cost, whenever aggregate information is required.

REFERENCES


Alvise Bonivento (S’03) received the Laurea degree in telecommunication engineering from the University of Padova, Padova, Italy, in 2002 and the M.S. and Ph.D. degrees in electrical engineering and computer science from the University of California, Berkeley, in 2004 and 2007, respectively. He is currently a private consultant for industrial automation and logistic systems. His research interest is in communication protocols for embedded systems, with emphasis on system level design and protocol synthesis for wireless embedded networks and wireless industrial networks. He is coauthor of several technical papers on major communication and embedded systems conferences.

Carlo Fischione (S’02–M’06) received the Laurea degree in electronic engineering (cum laude) and the Ph.D. degree in electrical and information engineering from the University of L’Aquila, L’Aquila, Italy, in 2001 and 2005, respectively. Currently, he is a research fellow at the University of California, Berkeley, Department of Electrical Engineering. He has been a research associate at the School of Electrical Engineering of the Royal Institute of Technology, Stockholm, Sweden (May 2005 to May 2007). His research interests are in the area of wireless networks and networked embedded systems, with particular reference to communication theory, radio resource control, and distributed computation. He has been in the TPC of conferences related to wireless sensor networks. Dr. Fischione is a member of the IEEE Communications Society and serves as a reviewer for technical journals and conferences.

Luca Necchi received the M.S. and Ph.D. degrees in electrical engineering from Politecnico di Torino, Torino, Italy, in 2003 and 2007, respectively. His Ph.D. dissertation was on “Architectural design for wireless sensor networks.” In 2005, he spent five months at the Electrical Engineering Department of the University of California, Berkeley, as a visiting scholar. His research interests include wireless sensor networks, embedded digital systems, and logic synthesis, with special emphasis on asynchronous circuits and ultra-low energy design.
Fernando Pianegiani (M’06) received the M.S. degree in electronic engineering from the University of Perugia, Perugia, Italy, in 2001 and the Ph.D. degree in information and communication technologies (ICT) from the International Doctorate School in ICT of the University of Trento, Trento, Italy, in 2005. He joined the Department of Electronic and Information Engineering of the University of Perugia, as a part-time researcher in 2001. During the summer of 2002, he did an internship at Marconi Mobile, Firenze, Italy. From 2004 to 2005, he was a Visiting Student Researcher at the Donald O. Pederson Center for Electronic Systems Design, Electrical Engineering and Computer Sciences Department (EECS) of the University of California, Berkeley, on leave from the University of Trento. From 2006 to 2007, he was a Visiting Scholar at the same center on leave from ArsLogica IT Laboratories, Mezzolombardo (TN), Italy. Currently, he is a PostDoc at the Department of Information and Communication Technology of the University of Trento and a Researcher for ArsLogica IT Laboratories. His main research interests are focused on the design and characterization of distributed measurement systems, wireless sensor networks, hardware/software co-design of embedded systems, analysis and control of nonlinear systems, signal processing, and pattern recognition techniques.

Alberto Sangiovanni-Vincentelli (F’82) holds the Buttner Chair of Electrical Engineering and Computer Sciences at the University of California, Berkeley. He was a cofounder of Cadence and Synopsys, the two leading companies in the area of electronic design automation (EDA). He is the Chief Technology Adviser of Cadence. He is a member of the Board of Directors of Cadence, UPEK, a company he helped spinning off from ST Microelectronics, Sonics, Gradient, and Accent, an ST Microelectronics-Cadence joint venture he helped in founding. He is a member of the HP Strategic Technology Advisory Board and of the Science and Technology Advisory Board of General Motors. He consulted for many companies including Bell Labs, IBM, Intel, United Technology, COMAU, Magneti Marelli, Pirelli, BMW, Daimler-Chrysler, Fujitsu, Kawasaki Steel, Sony, and Hitachi. He is the founder and Scientific Director of PARADES, a European Group of Economic Interest supported by Cadence and ST Microelectronics. He is a member of the High-Level Group and of the Steering Committee of the EU Artemis Technology Platform. He is an author of over 700 papers and 15 books in the area of design tools and methodologies, large-scale systems, embedded controllers, hybrid systems, and innovation.

Dr. Sangiovanni-Vincentelli received the Distinguished Teaching Award of the University of California in 1981. He received the worldwide 1995 Graduate Teaching Award of the IEEE for “inspirational teaching of graduate students.” In 2002, he was the recipient of the Aristotle Award of the Semiconductor Research Corporation. In 2001, he was given the prestigious Kaufman Award of the Electronic Design Automation Council for pioneering contributions to EDA. He has been a member of the National Academy of Engineering since 1998.