# Interaction Patterns for Resilient Intermittently-Connected Static Sensor Networks

Michel Charpentier, Radim Bartoš and Ying Li
Department of Computer Science
University of New Hampshire
Durham, NH 03824
Email: {charpov,rbartos,yws2}@cs.unh.edu

***Abstract*—We study systems of static and mobile sensors in which the participating nodes are disconnected. At any given time, only selected groups of nodes are able to interact. In this paper, we restrict attention to pairwise interactions of static nodes placed on a regular grid. Nodes must agree on a schedule that specifies when and in what order these interactions take place. Our focus is the impact this interaction schedule has on system-wide properties, such as information propagation and resiliency. We show how interaction schedules with varying properties can be built from simple patterns and we discuss the choice of suitable patterns on sample illustrative scenarios.**

## I. Context and Problem Definition

### A. Energy constraints and intermittent connectivity

The lifespan of a sensor network is typically determined by the amount of available energy in the nodes and the rate with which the energy is consumed. Technological constraints and cost make it difficult to increase the amount of energy in the nodes. Communication is one of the most significant energy consuming activities of a sensor node. A common method to extend the lifespan of a sensor network is to have the sensors spent a significant percentage of time in low energy consumption (sleep) mode [1]. Depending on the "depth" of the sleep mode, a node might be capable of sensing but it is not capable of sending or receiving. When a node is fully awake, it can send and transmit but consumes a significant amount of power which drastically reduces its lifespan.

The overall objective is to satisfy the requirements of a network's mission while minimizing the amount of energy consumed. To this end, we aim to reduce the energy cost of communication by reducing the time spent in the fully awake mode and by limiting the overhead and interference. More specifically, we eliminate the need for media access control (MAC) at runtime by orchestrating the exchanges between agents offline before deployment in a way that only two agents take part in any one exchange. Furthermore, these exchanges are scheduled (in time and possibly communication channel) so that no other node within the interference range can transmit at the same time. In such a scheme, nodes can transmit without a prior channel arbitration that is normally required to address the hidden terminal problem. This is especially important in networks with high propagation latency, such as underwater networks that use acoustic communication links, where such arbitration takes significant time or may even be impossible [2]. Nodes may also dissipate lower power because of the reduced interference. Restricting communication to pairs of nodes also allows the use of more energy-efficient and range-extending directional transducers. The use of directional communication may improve security by reducing the area in which the communication can be intercepted.

This research was supported in part by grant N00014-05-1-0666 from the U.S. Office of Naval Research.

### B. A network model based on scheduled meetings

The nature of the networks described above leads to a model in which nodes interact in local groups and do not attempt to reach specific nodes outside their group through long distance mechanisms like multi-hop routing. There are other scenarios in which this form of communication—group based and local only—is appropriate, including cases where nodes are mobile. In this paper, we focus on pairwise meetings of static sensor nodes laid out on a regular grid, but the model we propose is more general and can be applied to non regular topologies, possibly with meetings of more than two nodes [3].

Consider an undirected connected graph of nodes, possibly with cycles. We refer to this graph as the partnership graph of the nodes to emphasize that it is *not* a communication graph. In particular, the fact that this graph is connected does not mean that there are communication paths between all pairs of nodes. Indeed, we rely on local interactions only and no such end-to-end communication paths are used. The existence of an edge between $A$ and $B$ in the partnership graph means that nodes $A$ and $B$ are neighbors or "partners" and will interact (or attempt to interact) on a regular basis through meetings. Meetings of more than two agents make the partnership graph possibly a hypergraph; multiple meetings of the same nodes (for instance, mobile nodes meeting in different locations) make it a multigraph. The partnership graph used in this paper is a regular grid where inner nodes have four neighbors.

When a set $G$ of nodes are connected in a partnership, they need to schedule meetings, which are points in time—and, in the case of mobility, space—at which the nodes will interact. For simplicity, we restrict attention to periodic schedules in which a series of meetings is regularly repeated, i.e., nodes communicate at times $m^k = m^0 + k\delta$, where $m^0$ is the time of they first interaction and $\delta$ is the elapsed time between successive meetings of the two nodes. Like the topology of

the partnership graph, schedules are mostly static (obtained from offline calculations and loaded at or before deployment time), although we can envision dynamic changes to schedules and topologies as part of a strategy for network repair.

At scheduled meeting times, nodes interact (we say that they *meet*). These interactions can be elaborate (e.g., in-network calculation of complex functions based on the aggregated knowledge of the group) or simple (e.g., a single value sent to and received from everyone else in the group with no guarantees of delivery). Part of the design of a meeting based network is to decide what exactly should take place during node meetings. Such designs can lead to strategies and algorithms substantially different from their counterparts in more standard, routing based models. A full discussion of algorithmic and design issues is beyond the scope of this paper and we limit ourselves here to simple illustrative examples.

### C. Problem definition

Consider a simple event detection mission in which static sensors are deployed to monitor an area of interest and to report detailed information associated with an event once this event is detected. We assume that the topology formed by these sensors is a regular grid in which inside nodes have four neighbors: top, bottom, left and right (abbreviated here as $T$, $B$, $L$ and $R$). This grid serves as the partnership graph of the network and every node performs pairwise interactions with each of its four neighbors according to the network schedule. It should be noted that, in accordance to our model and for the practical reasons listed earlier, these meetings happen in sequence as a series of (at least) four pairwise meetings per period. There are no larger meetings of a node with all its neighbors at the same time, and nodes never interact directly with non-neighbor nodes. In this scenario, the partnership graph is a simple graph and the groups of nodes mentioned in the previous section are all pairs $\{A, B\}$.

We assume that the network includes destination nodes (sinks) to which the information needs to be reported. For instance, destination nodes can represent base stations, nodes with high-power transmission capabilities or the location of human operators. We assume the network contains at least one destination node, but it can also have more. For instance, a corner of the grid can be a single destination, or all four corners, or all the nodes located on one edge of the network, etc. The objective of the event detection mission is to report information to the destination node or to *at least* one such node if the network contains more than one. (It is assumed that destination nodes have robust networking capabilities and can communicate with each other at will.)

In each period of the schedule, a node will interact at least once with each neighbor. These interactions take place at pre-specified times, according to the "pulse" of the network. There are, however, many possible interaction schemes. If we restrict attention to schedules with a period equal to four—a node talks to each neighbor exactly once per period—we can denote by $m_0^i$, $m_1^i$, $m_2^i$ and $m_3^i$ the times at which these meetings take place during period $i$ (where $m_0^i < m_1^i < m_2^i < m_3^i$ and $m_u^i - m_u^j = \delta(i-j)$). A consistent schedule is a labelling of the edges of the partnership graph such that no node is connected to two or more edges with the same label $m_u$. There are many possible such labellings. Fig. 1 shows two possible schedules of a small $2 \times 2$ grid. Note how these schedules use the meeting times $m_0$, $m_1$, $m_2$ and $m_3$ differently. In the first case, two nodes visit their neighbors in the $TLBR$ order and two nodes use the $TRBL$ order; in the second case, all nodes use $TLRB$. Schedules on larger grids can be built by tiling small grids like those of fig. 1, as detailed in the next section.

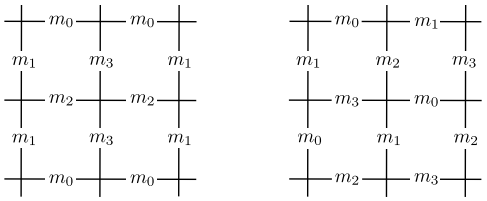


Fig. 1. Possible schedules and corresponding meeting times.

Our paper focuses on the following set of questions: Are all consistent schedules equivalent? If not, how can they be compared and what metrics can be used to compare them? Is there an optimal schedule or does the choice of a good schedule depend on the characteristics of the network's nodes and mission? In particular, what is the impact of the number and locations of destination nodes on the choice of a suitable schedule? Can different schedules improve different broader measures of the mission, such as expected latency between an event detection and its report to a destination node or the lifespan of the entire network given a rate of decay and a maximum acceptable latency?

## II. Interaction Patterns

### A. Tiling patterns on a regular grid

For an initial study and comparison of interaction schedules, this paper focuses on the grid topology introduced above and makes the following two assumptions:

*1:* Schedules are periodic with a period equal to four, i.e., nodes interact with each one of their four neighbors ($T$, $B$, $L$ and $R$) exactly once per period. There are six possible permutations of these four interactions: $TLBR$, $TRBL$, $TLRB$, $TRLB$, $TBLR$ and $TBRL$, which we denote by T, C, Ur, Ul, Dr and Dl, respectively (Fig. 2). We do not consider schedules of longer periods in which nodes interact more often with some of their neighbors, e.g., $TLTLRB$.

*2:* We restrict attention to regular schedules obtained from tiling a $2 \times 2$ pattern made of four of these possible orderings. For instance, fig. 2 shows a schedule obtained from tiling $\begin{array}{|c|c|}\hline \mathsf{T} & \mathsf{C} \\ \hline \mathsf{C} & \mathsf{T} \\ \hline \end{array}$ and another obtained from tiling $\begin{array}{|c|c|}\hline \mathsf{Ur} & \mathsf{Ur} \\ \hline \mathsf{Ur} & \mathsf{Ur} \\ \hline \end{array}$. The meeting times of these two schedules are those of fig. 1.

If "circling" orderings (T and C) are not mixed with "crossing" orderings (Ur, Ul, Dr and Dl) and symmetries are taken into account, there are exactly 14 possible tilings that result in consistent schedules (Fig. 3). We refer to each pattern by the names of the four orderings that it consists of (from

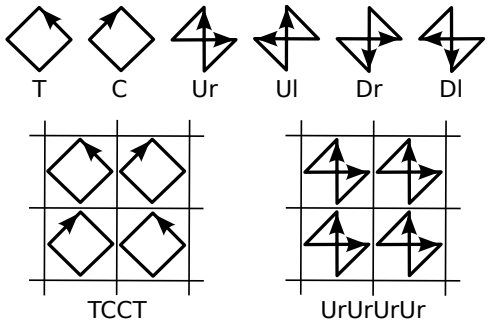


Fig. 2. Elementary schedules and patterns based tilings.

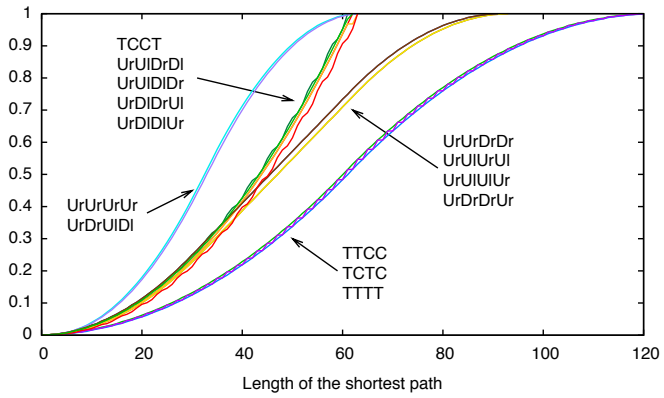


Fig. 5. Cumulative distribution function of the propagation delays from all nodes to the top right corner.

left to right and top to bottom) and each schedule by the name of its generating pattern. Hence, the two example patterns of fig. 1 and 2 are named TCCT and UrUrUrUr.

### B. Characteristics and properties of interaction patterns

Consider the event detection scenario discussed earlier. Fig. 3 shows, for each pattern, a graphical representation of information as it propagates from the center of a $31 \times 31$ grid. In other words, if the event being detected originates in the center of the grid and informed nodes always pass it on to uninformed nodes during meetings, the colors represent the number of interactions (and hence the time) it takes for each node to become aware of the event, from red (low values) to green (high values).

Two fundamental aspects of interaction patterns become evident on this picture:

*1:* Different patterns propagate information at different speeds and in different directions. Moreover, some patterns are asymmetric and have preferred directions of propagation. For instance, UrUrUrUr propagates information quickly upward and rightward—which is expected from the definition of the pattern—but also in a bottom-left direction, which is somewhat less obvious at first thought.[1] Patterns have different numbers and angles of preferred directions. For instance, UrUrUrUr has the 3 directions mentioned above, while UrUlUlUr has 5 (top, left, right, bottom-left and bottom-right).

*2:* The second characteristic to notice is that the number of informed nodes grows with time at different rates for different patterns. In other words, given the same amount of time, some patterns inform more nodes than other patterns. (Graphically, the yellow boundary can be thought of as a moment in time and the number of informed nodes at that time is represented by the amount of red within the boundary.) For instance, TTTT propagates information in the same directions as those of UrDrUlDl, but at half the speed. Indeed, calculations show that the number of informed nodes $t$ units of time after the event is in the order of $\frac{1}{2}t^2$ for TTTT and $2t^2$ for UrDrUlDl. Fig. 4 shows how patterns compare to each others in terms of numbers of nodes informed as a function of time: To each pattern corresponds a constant $g$ to represent the fact that $t$ units of time after the initial event, the pattern has informed a number of nodes in the order of $gt^2$. We refer to this constant as the *growth factor* of the pattern (and of the schedule it generates).

These two characteristics of interaction patterns (preferred direction of propagation and growth of the number of informed nodes) allow us to answer our first two questions, namely that not all consistent schedules are equivalent and that there are meaningful metrics to compare them. Furthermore, these two measures constitute important metrics, relevant to the event detection mission used here as an illustration. When deploying a network, a pattern should be chosen with preferred directions consistent with the locations of the destination nodes and a balance of speed and growth that delivers the best performance in terms of delays and power consumption. These questions are discussed more precisely in the following section.

## III. Mission Objectives and Interaction Strategy

### A. Directional speed and propagation delays

In our illustrative scenario, the information obtained by a sensor node needs to propagate to one or more destination nodes. We consider several variations of this scheme in which the destination node is: i) the top right corner; ii) either top corner; iii) any corner; iv) any node on the top edge; v) any node on the top or right edge; vi) any edge node; or vii) a small number (1% in our experiments) of special nodes randomly and uniformly distributed on the grid.

It is expected that, because they tend to propagate information in different directions at different speeds, some patterns will perform better than others for a particular scenario. Fig. 5 shows, for the scenario in which the destination is the top right corner, the cumulative distribution function of the time it takes for every node to transfer information from its location to the target. We observe that UrUrUrUr and UrDrUlDl are the fastest patterns for this scenario (other patterns, like TCCT or UrDlDlUr have similar maximum delays but higher averages). This is consistent with the fact that both patterns propagate information faster in the top and right directions.

[1]To understand this third direction of propagation, consider how meeting times $m_0 < m_1 < m_2 < m_3$ form top-right to bottom-left diagonals of propagating information in the UrUrUrUr pattern on fig. 1.

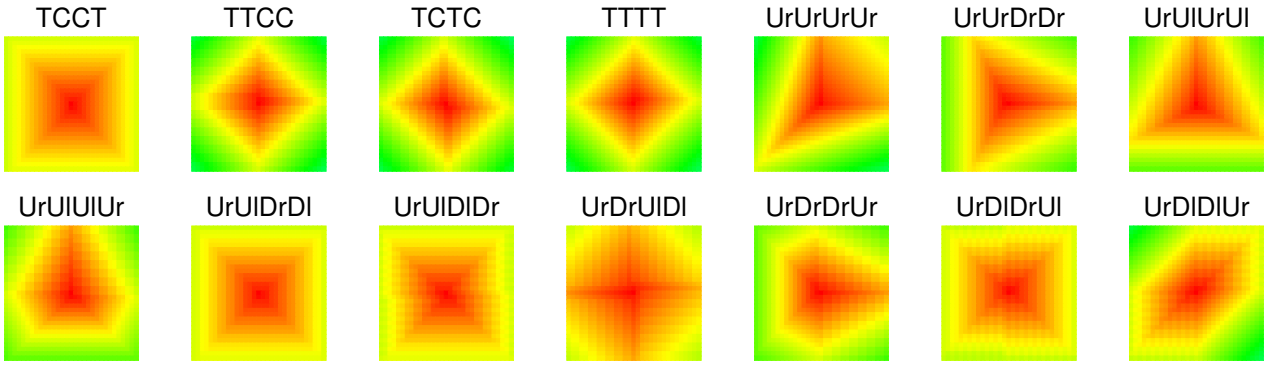


Fig. 3. Possible tilings from $2 \times 2$ patterns.

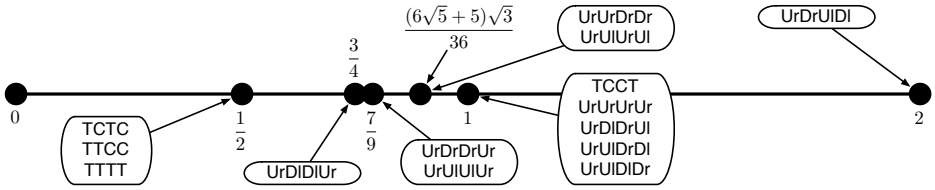


Fig. 4. Growth factors of tiled interaction patterns.

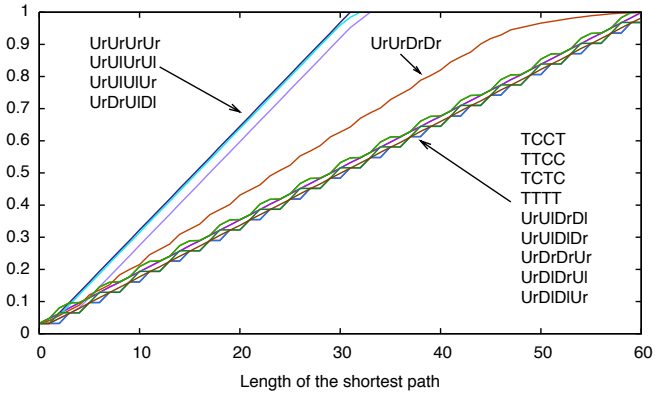


Fig. 6. Cumulative distribution function of the propagation delays from all nodes to the top edge.

If we consider instead the case where any node on the top edge is a valid destination (Fig. 6), patterns cluster in three groups. The fastest patterns are now UrUrUrUr and UrDrUlDl, as before, together with UrUlUrUl and UrUlUlUr, which both favor an upward direction. These two patterns were no contenders in the previous scenario because of their weak rightward propagation.

### *B. Pattern growth and energy consumption*

If the only metric were the time it takes to carry information from the node where it is produced to an acceptable destination, UrDrUlDl would tend to outperform all other interaction patterns for many scenarios. This is because it tends to propagate information rapidly in all directions. If the mission benefits from such an omnidirectional propagation (for instance, when there are destination nodes on all edges), this pattern may indeed be the right choice. If, however, the destination nodes are all located on the top edge of the field, the downward propagation of information of the pattern is wasted. This is an important consideration if we assume that interactions between uninformed nodes and informed nodes have higher costs in terms of energy resources (for instance, if they take more time and require the transfer of larger data).

In order to minimize flooding areas of a network with information that is not needed there, sensor nodes can use a classic cutoff technique. Given their known location on the grid, they can stamp the information they generate with a time-to-live value that is sufficient to reach the destination nodes. Propagation of this information will stop once it reaches this limit, which will limit the number of actual transfers between uninformed and informed nodes. Suitable time-to-live values depend not only on the relative position of a sensor node to the destination nodes, but also on the directional speeds of the chosen interaction pattern. A pattern that propagates slowly in some direction will require larger time-to-live values if nodes expect to reach a destination in this direction.

Consider again the scenario with a single destination node in the top right corner of the grid. We saw that UrUrUrUr and UrDrUlDl give equally good performance in terms of propagation delays (fig. 5). Because information propagation in the top and right directions is identical for these two patterns, they induce the same time-to-live values for this scenario. Recall, however, that UrDrUlDl has a growth factor that is twice that of UrUrUrUr (fig. 4). Given the same time-to-live values, there will be more costly transfers from uninformed to informed nodes with UrDrUlDl than with UrUrUrUr for no

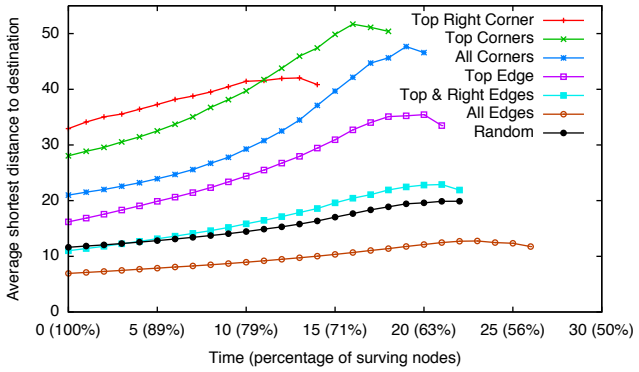


Fig. 7. Delay increase and node reachability of the UrUrUrUr pattern as sensors cease to operate.

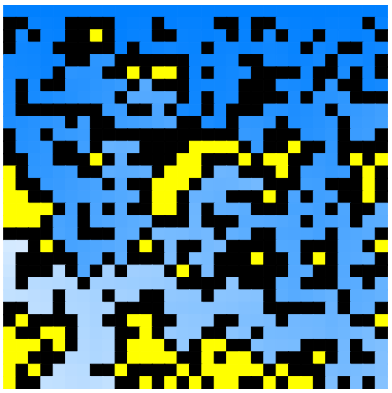

Fig. 8. Lengths of shortest paths from all nodes to the top edge for the UrUlUlUr pattern. Black nodes are disabled; yellow nodes are isolated.

benefit in terms of propagation delays. Therefore, UrUrUrUr is the better interaction pattern for this scenario. In the scenario where the destination is the entire top edge, the four best patterns delay-wise have growth factors of 0.778 (UrUlUlUr), 0.886 (UrUlUrUl), 1 (UrUrUrUr) and 2 (UrDrUlDl). Therefore UrUlUlUr should be the preferred interaction pattern. Indeed, with its upward "spaceship" shape, this pattern naturally limits propagation of information in all other directions.

## IV. Robustness and Network Longevity

This paper proposes to use pairwise, local interactions between nodes as the sole communication mechanism used by a network. This means that any end-to-end propagation of information between nodes is the result of multiple interactions as data is passed from node to node along a shortest path. If nodes on this shortest path cease to operate, the information will need to follow alternate (possibly longer) paths. This section discusses the impact of node failure on the length of these communication paths. In the event detection illustration, these lengths correspond to the time it takes to transfer data from the location of the event to one of the destination nodes.

Fig. 7 shows, for the UrUrUrUr interaction pattern and the seven mission scenarios considered in the paper, how the performance of the network evolves as nodes begin to fail. Failures follow a standard exponential law, from all nodes active to about half the nodes still operating. We consider the length of a shortest path between a sensor node and at least one of the destination nodes of a particular scenario. We only consider those nodes for which such a path exists (that is, nodes that cannot reach any destination are ignored) and we average the lengths of all shortest paths. Assuming that the event being detected happens at any location with uniform probability, this average corresponds to the expected latency of the mission.

As one would anticipate, the latency is lowest in scenarios with more destination nodes, the worst case being that of a single listener in the top right corner. As nodes cease to operate, this latency begins to increase because information follows longer paths around the failed nodes. This increase is minimal in scenarios with many destinations because it takes many failed nodes to impact all the shortest paths to all the destinations. Scenarios with fewer destinations are more sensitive to failures but still benefit from the fact that, most of the time, multiple paths exist between a node and a destination. The graphs are not monotonic because we only consider nodes from which at least a destination is reachable. Low latency can be the result of having destinations reachable from only a small number of nodes, albeit with fast propagation. This is especially true of the top right corner scenario, for which the single destination is more likely to become isolated from large areas of the network. Each plot ends when more than half the nodes are either dead or isolated from all destination nodes, which happens sooner in scenarios with fewer destinations.

When the entire top edge consists of destination nodes, the most efficient pattern, as argued in the previous section, is UrUlUlUr because of its combination of speed and growth factor. Fig. 8 shows a stage where 42% of the nodes have failed (in black) and another 9% are isolated from all destinations (in yellow). Lighter shades of blue indicate longer shortest paths. In this snapshot, we can see several isolated areas, as well as an area towards the bottom left from which the destination edge is still reachable through long shortest paths.

## V. Related Work

Wireless sensor networks (WSNs) have been extensively studied and to some extent deployed in practice [4]. Energy conservation has been identified as one of main issues in WSN research and active power management through scheduling of modes of operation has become one of the main approaches to address it [1].

The concept of a network where end-to-end connectivity is not assumed has been explored in sensor networks [5] and other settings such as Delay Tolerant Networks (DTNs) [6], [7]. DTNs constitute an example of disconnected mobile networks in which encounters among nodes are driven by forces unrelated to their mission. The main objective of DTN research is to ensure information delivery despite the periods of disconnectivity. Our approach differs insofar as communication opportunities here are orchestrated to serve the mission of the network.

At a more formal level, results on asymptotic capacity of static wireless networks [8] have been applied to mobile cases [9], [10]. In contrast to this paper, these works assume either random node placement or random mobility and exploit opportunistic communication.

While omnipresent communication infrastructure is becoming reality in many places, there are cases where a WSN would not or could not tap into it. Underwater acoustic networks is one of such cases [11]. Many of the traditional networking approaches are not applicable in the underwater environment and new solutions are required for MAC [12] and routing [13]. Furthermore, severe challenges of the underwater environment make it a logical place to use physical data ferrying [14] and orchestrated communication.

The need for synchronization is common to all sensor networks where nodes switch between active and sleeping modes. If precise onboard clocks are not available, time synchronization can be achieved through standard protocols like NTP or IEEE 1588. Alternatively, self-synchronizing schedules can be achieved using data traffic associated with the mission without the need for explicit synchronization messages [15]. As part of our research, we are exploring energy efficient algorithms capable of maintaining synchronization in low traffic missions.

## VI. Conclusions and Future Work

In many settings, networks are an inexpensive commodity that allows processes in a system to communicate when interaction is needed. Yet, there are and will remain scenarios in which such networks are not available (or desirable) and communication between participants becomes a challenge. Much research has been done to equip such challenging environments with communication facilities that rely on conventional networking abstractions and offer acceptable performance and security. Another line of research, to which this paper contributes, is to forgo the standard abstractions and to explore different ways for system entities to interact. This approach, however, changes design strategies and impacts many decisions related to mission implementation.

In this paper, we explore the possibility of making energy constrained sensors rely exclusively on local, group-based, short-lived interactions while maintaining the desired global behavior of a network. We focus on the fundamental issue of scheduling meetings of nodes in ways that balance performance and resource usage. We show how simple patterns of pairwise interactions on a regular grid can be used as building blocks and how they become important parameters in a design driven by the goals and constraints of a particular mission.

Most of the data presented in this paper are the result of simulations. We have implemented a general simulator that we plan to apply to other topologies, regular (hexagonal grids, square grids with corner meetings of four nodes) and non regular. We are also interested in cases where some or all of the nodes are mobile because these cases bring different challenges—in addition to *when* interactions take place, nodes must also decide *where* they take place, in ways that are consistent with their mission driven motion—and different opportunities—mobility can be used to dynamically modify topologies in an effort to repair a network. We have started to explore schemes that allow mobile agents to detect damage to a network and initiate effective repairs in a local way without central coordination.

In addition to simulations, we are developing analytical models that will allow us to study the properties of meeting-based computations more rigorously. In this, we are following earlier work on population protocols [16] and self-similar algorithms [17], from which we also draw inspiration. A preliminary model of schedules generated from patterns was used to compare growth factors precisely (fig. 4) and to confirm simulated data.

## References


[1] L. Wang and Y. Xiao, "A survey of energy-efficient scheduling mechanisms in sensor networks," *Mob. Netw. Appl.*, vol. 11, no. 5, pp. 723–740, 2006.

[2] V. Rodoplu and M. K. Park, "An energy-efficient MAC protocol for underwater wireless acoustic networks," in *Proc. of IEEE/MTS Oceans 2005*, Washington, DC, Sep. 2005.

[3] M. Charpentier, R. Bartoš, and S. Bhatia, "A mechanism to structure mission-aware interaction in mobile sensor networks," in *Proc. of the 10th Intl. Conf. on Distributed Computing and Networking (ICDCN 2009)*, Jan. 2009, pp. 425–436.

[4] Y. S. Ian F. Akyildiz, Weilian Su and E. Cayirci, "A survey on sensor networks," *IEEE Communications Magazine*, vol. 40, no. 8, pp. 102–114, Aug. 2002.

[5] O. Dousse, P. Mannersalo, and P. Thiran, "Latency of wireless sensor networks with uncoordinated power saving mechanisms," in *MobiHoc'04*. New York, NY, USA: ACM, 2004, pp. 109–120.

[6] K. Fall, "A delay tolerant networking architecture for challenged internets," in *SIGCOMM 2003*, Aug. 2003.

[7] A. Lindgren and P. Hui, "The quest for a killer app for opportunistic and delay tolerant networks: (invited paper)," in *CHANTS '09: Proceedings of the 4th ACM workshop on Challenged networks*. New York, NY, USA: ACM, 2009, pp. 59–66.

[8] P. Gupta and P. R. Kumar, "The capacity of wireless networks," *IEEE Trans. on Information Theory*, vol. 46, no. 2, pp. 388–404, Mar. 2000.

[9] M. Grossglauser and D. Tse, "Mobility increases the capacity of adhoc wireless networks," *IEEE/ACM Transactions on Networking*, vol. 10, no. 4, pp. 477–486, Aug. 2002.

[10] T. Spyropoulos, K. Psounis, and C. Raghavendra, "Efficient routing in intermittently connected mobile networks: The multi-copy case," *IEEE/ACM Trans. on Networking*, vol. 16, no. 1, Feb. 2008.

[11] D. Pompili and I. F. Akyildiz, "Overview of networking protocols for underwater communications," *IEEE Communications Magazine*, vol. 47, no. 1, pp. 97–103, Jan. 2009.

[12] K. B. Kredo and P. Mohapatra, "A hybrid medium access control protocol for underwater wireless networks," in *ACM International Workshop on UnderWater Networks (WUWNet)*, Montreal, Canada, Sep. 2007.

[13] D. Pompili, T. Melodia, and I. F. Akyildiz, "Routing algorithms for delay-insensitive and delay-sensitive applications in underwater sensor networks," in *MobiCom '06: Proceedings of the 12th annual international conference on Mobile computing and networking*. Los Angeles, CA: ACM, 2006, pp. 298–309.

[14] R. Bartoš, S. G. Chappell, R. J. Komerska, M. M. Haag, S. Mupparapu, E. Agu, and I. Katz, "Development of routing protocols for the solar-powered autonomous underwater vehicle (SAUV) platform," *Wireless Communications and Mobile Computing*, vol. 8, no. 8, pp. 1075–1088, Aug. 2008.

[15] K. Xu, O. Dousse, and P. Thiran, "Self-synchronizing properties of CSMA wireless multi-hop networks," in *ACM SIGMETRICS'10*. New York, NY, USA: ACM, Jun. 2010.

[16] D. Angluin, J. Aspnes, Z. Diamadi, M. Fischer, and R. Peralta, "Computation in networks of passively mobile finite-state sensors," *Distributed Computing*, vol. 18, no. 4, pp. 235–253, Mar. 2006.

[17] K. M. Chandy and M. Charpentier, "Self-similar algorithms for dynamic distributed systems," in *27th International Conference on Distributed Computing Systems (ICDCS'2007)*, Jun. 2007.