

A Mechanism to Structure Mission-Aware Interaction in Mobile Sensor Networks

Michel Charpentier, Radim Bartoš and Swapnil Bhatia

Department of Computer Science,
University of New Hampshire,
Durham, NH, USA
{charpov,rbartos,sbhatia}@cs.unh.edu

Abstract. One of the main appeals of mobile sensors is the variety of environments in which they can operate as an autonomous network. Different environments, however, present different challenges, especially in terms of inter-sensor communication. In sparse environments, it may not be possible to maintain full connectivity at all times, setting off the need for agents to communicate opportunistically when they are close to each other. This, in turn, suggests that communication needs be taken into account in the design of agents' trajectories. In this paper, we introduce a notion of *tour* and *meeting point* as an abstraction of trajectories designed for both sensing and communication, and we study the tradeoffs involved between motion to sense and sample and motion to communicate and interact.

1 Introduction and Motivation

1.1 Motivation

Mobile sensor networks hold tremendous promise as a technology. Thanks to controlled mobility, the network can now be made autonomous and thus, more adaptable to the phenomenon it is deployed to study, making it possible to build and deploy networks that can sense and act in the physical world in programmable ways. Network designers, however, need to consider *sensing*, *motion*, *communication*, and *computation* issues simultaneously, which is a true challenge to mission preparation and deployment. In our broader effort, we are investigating methods that would provide mobile sensor network operators with means to deal with all these features in a unified way.

In this paper, we focus on networks in which mobile agents (or nodes) cooperate towards a common mission and the geographical region in which they operate is vast relative to the number of agents and their communication range. In such networks, inter-agent communication changes from a commodity, used by the system when cooperation is needed, to a limited resource that impacts the network's mission-solving strategy—very much like energy, computational power and the ability to move.

This shift in perspective leads us to reexamine the overall network communication architecture. For instance, mobility makes it difficult to maintain

the structure necessary to the realization of standard communication protocols, while at the same time knowledge of the network’s mission can result in a better exploitation of available communication opportunities. As a result, it may not be as beneficial as it was before to maintain a clear separation between communication and computation. We believe that the time has come to look for communication strategies that are better adapted to the new challenges of mobile computing. This paper presents some of our exploratory steps towards such novel communication architectures.

1.2 Contributions

Consider agents with limited communication capabilities, charged with a mission that requires them to interact with each other while sampling vast areas. We assume that when agents move towards each other, they can rely on a local, efficient form of communication. In this context, agent trajectories cannot be designed in terms of sensing duties alone but must take into account communication needs as well. Agent trajectories are thus constrained by their individual tasks (what parts of the space they must explore) as well as their collaborative behavior (what other agents they need to communicate with and when).

As a possible structure to help design such trajectories, this paper defines and studies the notion of a *tour*. Tours involve a combination of motion and communication and can be used to model requirements and tradeoffs between the individual sensing tasks of mobile agents and their collaboration with respect to the mission. Tours are a simple model that can be used to investigate mission-solving strategies in order to assess their compliance with desired performance and overall network behavior.

2 Background

This paper is motivated in part by our past and current efforts in the field of autonomous underwater vehicles (AUVs) [1]. The underwater environment poses numerous challenges. Submerged AUVs typically utilize acoustic links with limited range and bit rates, high error rates, and propagation latencies many orders of magnitude larger than those of radio frequency links. At the same time, AUVs are typically used for sensing missions that span areas that are vast when compared to the communication range. Since nodes must be relatively close for the acoustic data transmission to be successful, mobility of AUVs is the key enabler of communication in such a sparse environment. The cost of AUVs and the cost of deployment prohibits the use of a large number of vehicles. On the other hand, mobility in the underwater environment is free of some of the constraints typical in the ground and air-based mobile sensor networks. Furthermore, modern AUVs are typically equipped with navigational aids that provide them with their absolute location and with precise clocks that facilitate maintaining synchronicity during a mission. As a result, networks of cooperating AUVs provide an important case for the study presented in this paper.

Much of the work in the area of networks operating in a challenging environment has been done under the umbrella of *Disruption/Delay Tolerant Networks* (DTNs). Fall [2] proposes a non-interactive messaging based overlay architecture for DTNs. Ho et al. extends the DTN architecture to sensor networks [3]. There has been significant work in the area of routing protocols for such networks. Zhao et al. [4, 5] and Burns et al. [6, 7] propose routing protocols that can influence and exploit mobility patterns to improve routing in DTNs. Subramanian et al. propose a utility-driven routing protocol that can optimize a given metric such as delay and deadline violations [8]. The DTN architecture provides a simple message delivery primitive appropriate for intermittent bandwidth-constrained transmission opportunities. Using clever and sophisticated replication heuristics, DTN routing protocols strive to minimize packet delays. However, by design, the DTN architecture is oblivious to the purpose of the network, which is a defining feature of mission-centric networks. We believe that a mission-aware communication subsystem will be able to better utilize the scarce communication opportunities in a mission-centric network.

Underwater communication, an area that has received significant attention in recent years [9–12], represents a compelling example of networking in a challenging environment. Underwater networking problems are studied at many different levels starting from acoustic communication links all the way to multihop routing in connected underwater networks. The results of our field experiments [13] have shown the importance of proper handling of intermittent connectivity during a mission due to the challenging environment. We have already taken the first steps toward addressing this issue at the vehicle code level [14]. This paper presents some of our initial steps to address the difficulty of designing AUV missions in challenging environments at a more fundamental level.

3 Tours and Meeting Points

3.1 Motion, sensing, communication and computation

We consider networks of mobile sensing agents deployed to perform a mission that involves periodic data acquisition and distributed computation. Agents are charged with the task to monitor (sense, scan, collect data) an area of interest in a periodic way and to compute relevant information from this sensed data in a distributed fashion. We assume that the area is vast compared to the communication capabilities of agents and agents will need to move towards each other in order to communicate efficiently. When they are close enough, agents can rely on local communication mechanisms to form groups, exchange information and compute their new states according to some predetermined algorithm.

When solving these missions, in which effective communication involves movement, the motion of agents must be programmed so they all visit their share of the area of interest *and* they form groups often enough to carry out a distributed computation from the acquired data. We propose to design this combination of motion, sensing, computation and local communication in terms of *tours*.

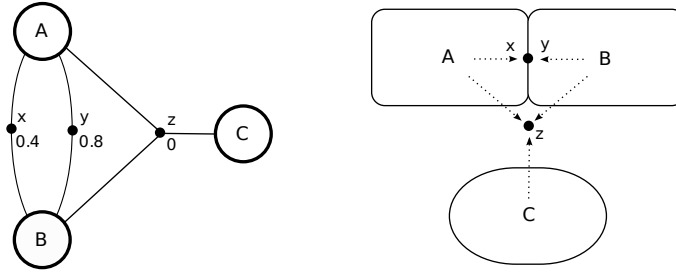


Fig. 1. Tour graph and possible corresponding tour areas and meeting points.

A tour is defined by an *area* (to be repeatedly sampled) and a collection of *meeting points* (to be visited by groups of agents). The overall space to be covered by the network is partitioned into several tours and each tour is the responsibility of a single agent. This agent’s task is, for each period of the cyclical computation, to sense and sample its tour area while maintaining a schedule of visits to its meeting points in order to exchange information and perform joint computational steps with agents from other tours.

A tour is thus implemented by a trajectory that covers the entire area to sample and returns to the meeting points at regular intervals to interact with other agents. The frequency with which a trajectory visits each meeting point is one of the key parameters of a tour implementation. We denote by t the amount of time between consecutive visits to a given meeting point. Low values of t correspond to trajectories that visit their meeting points many times while scanning the tour area. High values of t correspond to trajectories that complete large amounts of sensing between meetings.

3.2 Tours as network building blocks

Tours can be connected through their meeting points and assembled into a network that implements the desired mission. Formally, such a network of tours is a graph in which the vertices are tours and the edges are meeting points. This graph is both a multigraph (the same tours can be connected through several meeting points) and a hypergraph (meeting points can connect more than two tours). Fig. 1 represents a network that consists of three tours (A , B and C) and three meeting points (x , y and z). The figure also shows a possible physical realization of this network. In this realization, agents¹ A and B monitor adjacent rectangular areas and rely on two meeting points x and y that share the same location in the middle of the side common to the two rectangles. Agent C monitors a circular area and interacts with agents A and B through meeting point z . This point is located outside the tour areas and all three agents will need to leave their assigned area to attend meetings at this point. A given tour graph can have several different physical realizations.

¹ Here, “agent A ” is used as a shortcut to mean “the agent in charge of tour A ”.

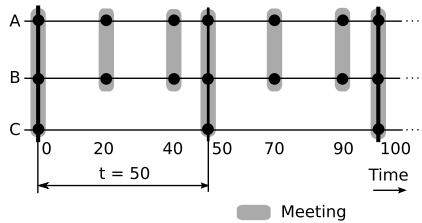


Fig. 2. Schedule of meetings.

In a tour connecting graph, each meeting point is labeled with a number between 0 and 1 that specifies when, during the tours, this meeting point must be visited (the labels 0 and 1 are equivalent and both denote the edge of a cycle). This graph is associated with a time-between-visits t that is common to all tours (considerations involved in choosing t are discussed later in the paper). This time-between-visits and the labels

together define the schedules that the tour trajectories must implement in order to guarantee that groups of agents will form at meeting points.

For instance, suppose the desired time-between-visits t of the network of Fig. 1 is 50 minutes. This means that all tour trajectories are based on a repetition of a 50 minute cycle of visits. Each 50 minute time slot includes sensing (of some given part of the tour area) as well as visits to the meeting points as specified by the labels. The label 0.4 associated with meeting point x means that this point will be visited by agents A and B at times $0.4 \times 50 = 20$, $(1 + 0.4) \times 50 = 70$, $(2 + 0.4) \times 50 = 120$, etc. Fig. 2 shows what groups of agents are formed at what times during the first 100 minutes in this network.

3.3 Implementing tours

In our vision, networks are built by assembling tours in accordance with the mission to solve and the strategy and algorithms chosen to solve it. The task of programming the movement of agents so they achieve the desired sensing and interaction is pushed into the tour implementation. Although the details of tour implementation are not the focus of this paper, this section aims to define the implementation problem precisely and present some of the difficulties and tradeoffs that such implementations involve.

A tour-based design of a network results in a desired time-between-visits t and a meeting schedule (in the form of numbers associated with each meeting point). Together with the geometrical definition of the area to be sensed during the tour, these represent the constraints to be satisfied by a tour implementation. The output of a tour implementation process consists of a trajectory that satisfies the following requirements:

- It visits all the meeting points at their specified times. If a meeting point x is labeled with $v_x \in [0, 1]$ and the desired time-between-visits is t , the trajectory must visit this point at times $v_x \times t$, $(1 + v_x) \times t$, $(2 + v_x) \times t$, etc.
- It covers (for sensing and data gathering) the entire tour area as quickly as possible. In other words, the area A of the tour should be visited entirely every $R \times t$, where R is as small as possible.

This trajectory design problem can be thought of as an optimization problem, in which the goal is to minimize R given the other parameters. A tour implemen-

tation should attempt to cover as much of the area as possible between meetings because it will allow agents to use more recent samples when they interact with other agents in the course of the distributed computation performed by the network. If, given the best possible R , the duration $R \times t$ between samples is still too high for the given mission, the network will have to use smaller tours and hence more agents, as expected.

3.4 Tradeoff between agent interaction and individual activities

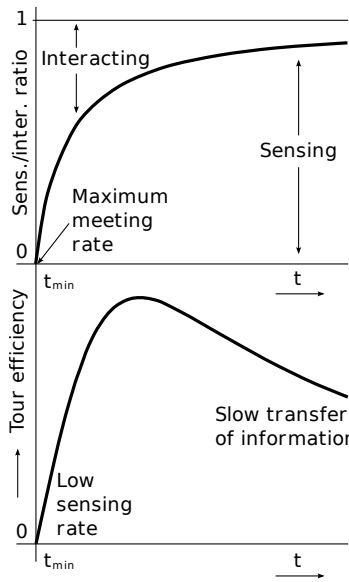


Fig. 3. Sensing / meeting tradeoff.

The tradeoff between desired sensing rates, the overall area to monitor and the number of agents is unavoidable and has little to do with tours. Tours, however, involve another, more interesting tradeoff due to the fact that agents have to share their time between sensing and traveling to meeting points. Low values of t , the time-between-visits, means more interaction with other agents, which benefits distributed algorithms in general. However, this also implies that agents need to interrupt their sensing task more often to attend meetings, and this results in a smaller proportion of agent time dedicated to sampling and sensing activities, thus in a lower overall sensing rate and the need for agent to rely on inaccurate (out-of-date) data during meetings. For most missions and associated algorithms, we expect that there will be an optimal time-between-meetings t_{opt} . If known, this optimum can be fed to the tour implementation optimization problem in order to minimize the time to sample the entire tour area.

Fig. 3 illustrates the tradeoff that is involved when deciding on a suitable time-between-visits. As this parameter decreases, agents have to spend more time traveling to meeting points and hence have less time for sensing. There is a minimum time-between-visits t_{min} for which agents spend all their time attending meetings and no sensing takes place. As t increases, agents can fit more sensing in each tour period and as a result will bring data that is more up-to-date when they attend meetings. Transfer of information, however, becomes slower as agents attend fewer meetings, which is certain to have a negative impact on most distributed algorithms. In many situations, there will be an optimal time-between-visits t_{opt} that results in maximum efficiency. Efficiency will decrease when $t > t_{\text{opt}}$ because of lack of interaction among agents; efficiency will also decrease when $t < t_{\text{opt}}$ because of agents using stale data from lack of sensing.

4 Application

4.1 Tour-based AUV missions

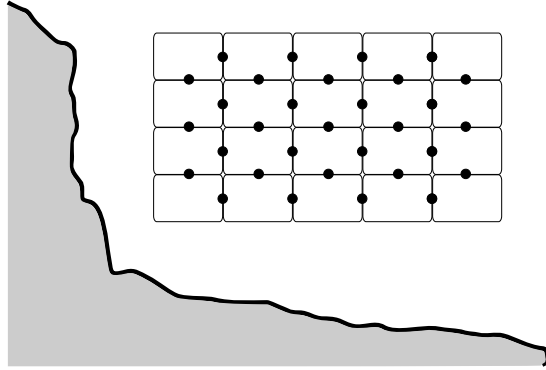


Fig. 4. Typical environmental monitoring mission: cooperating AUVs continuously sample an area of interest looking for an event of interest.

Many of the typical missions are amenable to the concept of tours. A broad category of AUV deployments aim at providing continuous environmental sensing of an area. In the cases of larger areas or higher required temporal resolution, the total sampled area is subdivided and assigned to vehicles as shown in Fig. 4. Sub-areas are typically sensed by a single AUV performing some variation of the “lawn mower” pattern. A recent field experiment performed at an AUV event in Monterey Bay, where AUVs were tasked to detect transient “thin layers” of biological activity [13], is an example of such a mission. Persistent barrier patrol (fig. 5) is an example of a security or law-enforcement AUV mission where a line of AUVs monitor a passageway for unusual activity [15]. Another example of a mission is inspired by the RiverNet project [16] where AUVs are used to monitor pollutants in the Hudson River. Fig. 6 shows several groups of AUVs performing vertical profiling of the river and facilitating cooperative adaptive sampling by intra- and inter-group communication.

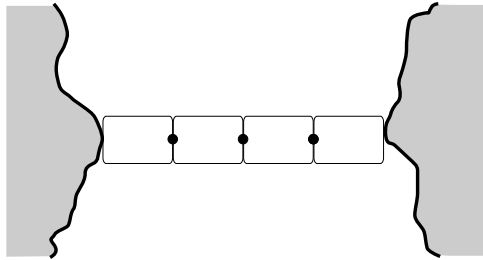


Fig. 5. Surveillance mission: a narrow waterway is monitored for unusual activity.

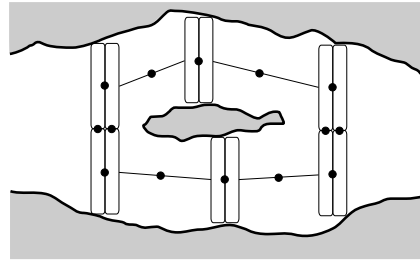


Fig. 6. An example of a pollutant monitoring mission in a river channel.

4.2 Event-detection mission on a regular 3D pattern

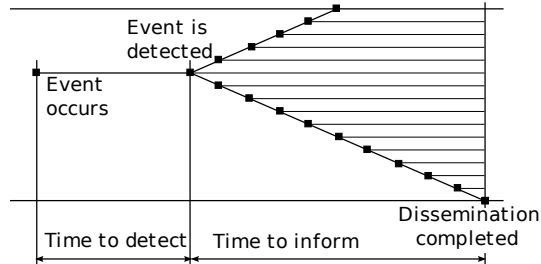


Fig. 7. Time to detect and time to inform.

Consider an event-detection mission in which agents are assigned different areas, which they repeatedly sample to decide if the event of interest has occurred. In accordance with the tour pattern, agents also meet regularly to inform other agents (or be informed by them) whether the event has happened in their own individual areas. We assume that the event of interest is such that it can be observed by a single agent independently, and that the goal of the agents as a group (the mission) is that they all know that the event has happened as quickly as possible after any occurrence of the event. Accordingly, performance is measured in terms of the amount of time between an occurrence of the event and the moment awareness of this occurrence reaches the last agent.

Fig. 7 describes the two stages involved in solving such a mission: event detection *per se*, by some agent, followed by a propagation of information. This example illustrates the tradeoff between individual work and collaboration. With more interaction, agents spend less time monitoring their area and the time to detect increases. However, once the event is detected, the propagation of information benefits from this high-level of interaction and the time needed for this information to reach all agents decreases. Conversely, agents can detect the event more quickly if they spend less time attending meetings, but the propagation stage will then take longer. We are interested in finding the optimum amount of interaction that results in the best overall performance.

Case study: underwater event detection in straight lines. In order to keep this illustrative example simple, we use the following model of costs for sensing and interaction. N agents navigate in circles, looking for the event of interest along the line of their circle (Fig. 8). Each circle has length (circumference) L and we assume that agents sense and travel horizontally with speed λ .

Each circle is explored by a single agent as a tour (the “area” monitored in each tour is the circle itself). Tours are connected via meeting points in a linear graph (inner tours have two meeting points and the top and bottom tours only have one). The distance between two adjacent circles is $2d$ and agents meet at distance d from their circle, which means that agents travel a roundtrip vertical distance of $2d$ for each meeting they attend. We assume that agents travel vertically with speed μ .

Let t be the time-between-visits as before. Each agent will attend two meetings per t period (1 up and 1 down), except for the top and bottom agents, which only do one meeting (and could either sense more slowly or wait at meeting points

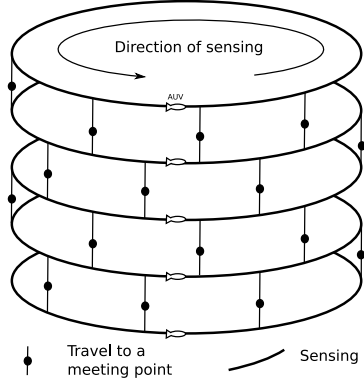


Fig. 8. Event detection example

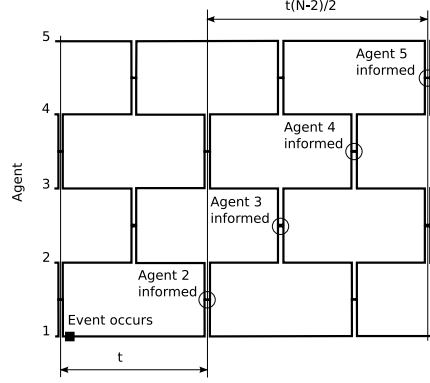


Fig. 9. Information propagation (worst case scenario).

to stay in sync with the rest of the group, depending on other considerations like energy consumption). We are interested in finding the optimal time-between-visits t that will result in the fastest completion time for the mission.

In this section, we analyze the network via a worst-case calculation of how much time elapses between one occurrence of the event and the earliest moment all agents are aware that the event has occurred. We carry out this calculation in terms of the parameters t , L , d , λ and μ . It is straightforward to see that the worst case scenario is one where there is a single occurrence of the event and the following conditions hold:

- the event happens on the top or bottom line, which causes the information to travel from one end of the network to the other;
- it is located right after a meeting point, which means with the longest possible time before the next opportunity to communicate with other agents;
- it happens when the agent has just traveled past this point, forcing the agent to travel one full circle back to this point before the event is detected.

During each period t , agents on the inner lines attend two meetings and need to travel a vertical distance of $2d$ at speed μ for each one of these meetings. Overall, agents spend $2 \times \frac{2d}{\mu}$ in each period t traveling to meeting points. There must be enough time in each period for agents to travel to their meeting points, which defines t_{\min} , the smallest possible t :

$$t_{\min} = \frac{4d}{\mu}$$

We assume $t > t_{\min}$, which leaves $t - \frac{4d}{\mu}$ out of each period t for agents to sense while they cover a horizontal distance of $\lambda(t - \frac{4d}{\mu})$. Therefore, it takes $\frac{Lt}{\lambda(t - \frac{4d}{\mu})}$ for an agent to complete one full circle. As discussed before, this is also the time

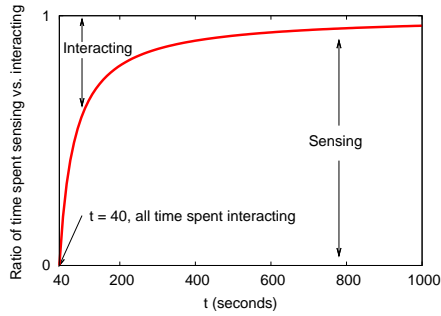


Fig. 10. Proportion of the total tour time spent sensing and interacting ($T=1000$ s, $L=10$ km, $d=10$ m, $\lambda=5$ m/s, $\mu=1$ m/s).

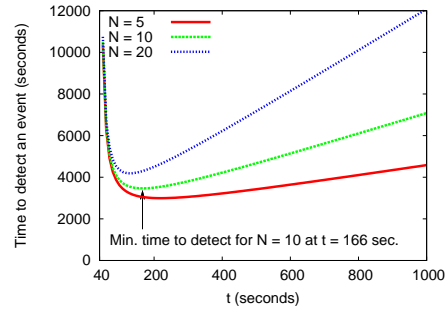


Fig. 11. Worst case time to detect an event ($L=10$ km, $d=10$ m, $\lambda=5$ m/s, $\mu=1$ m/s) for varying number of nodes.

for event detection in the worst case:

$$\text{Time to detect} = \frac{L}{\lambda} \frac{1}{1 - \frac{4d}{\mu t}}$$

Fig. 9 shows the propagation of information after the event is detected. It takes t units of time for the top (or bottom) agent to reach its meeting point and inform another agent. It then takes $\frac{t}{2}$ for this agent to inform the next one, and so on. Overall, it takes $t + (N - 2)\frac{t}{2} = \frac{Nt}{2}$ for the information to reach all N agents:

$$\text{Time to inform} = \frac{Nt}{2}$$

The overall duration between event occurrence and the end of information dissemination is the sum (Time to detect + Time to inform), minimized when:

$$t_{\text{opt}} = \frac{4d}{\mu} + \sqrt{\frac{8Ld}{\lambda\mu N}}$$

Note that $\frac{4d}{\mu}$ is t_{min} , the time spent traveling to (and back from) meeting points, which leaves $\sqrt{\frac{8Ld}{\lambda\mu N}}$ for scanning and sensing.

Illustration. As an illustration, consider a situation where $L=10$ km, $d=10$ m, $\lambda=5$ m/s, $\mu=1$ m/s and $N=10$. In this case, the optimum $t_{\text{opt}}=166$ seconds, of which 40 seconds are spent traveling to meeting points and 126 seconds are spent scanning. Fig. 10 shows the proportion of time spent sensing versus traveling to meeting points as a function of t . With an amount of interaction set to t_{opt} , it takes 44 minutes for an agent to complete a full circle (instead of about 33 minutes if agents didn't need to attend meetings to communicate) and the overall time to mission completion is 58 minutes. As a comparison, reducing

agent interaction by half ($t=333$ seconds) allows agents to cover the tour in 38 minutes but increases the time to completion to 66 minutes; doubling agent interaction ($t=83$ seconds) results in a tour time of 64 minutes and a time to completion of 71 minutes. At one extreme ($t=40$ seconds), all the agents' time is spent interacting and no sensing takes place; at the other extreme, the smallest (meaningful) amount of communication to complete the mission (the circle is entirely sampled before each meeting) corresponds to $t=4040$ seconds, in which case agents need only 33.7 minutes to sample the circle but leads to a time to completion of 6 hours and 10 minutes. Fig. 11 shows the time to mission completion as a function of t and for different values of N . Fig. 10 and 11 illustrate, for this case study, the general tradeoff discussion of sect. 3.4 (Fig. 3).

5 Current and Future Work

Mobile sensor networks, and other applications in which mobile agents carry out distributed computations in a challenging environment, present the networking task with new difficulties. In many situations, standard networking techniques can be modified, extended and adapted to address these new challenges. We feel, however, that there is also an opportunity for a fresh look at some of the assumptions that underlie the networks in use today.

In our broader effort, we are seeking primitives that combine communication with other tasks central to the new networks, like sensing, moving and computing. By moving away from communication as an address-based, always-on service more or less independent (in its interface, not necessarily its implementation) from other agent activities, we hope to explore novel designs and architectures better adapted to the inherent needs of mobile networks.

The notion of tours presented in this paper is an example of a mechanism that combines motion and communication so they can be jointly optimized based on other network parameters, such as sensing range, energy consumption and the definition of the mission to be implemented. We are currently exploring other mechanisms, and we plan to continue to study tours based networks. In particular, we would like to have the steps of the distributed computation play a more explicit role in the model. For instance, the event detection mission used as an illustration in the paper relies on extremely simple agent states (boolean) and computational steps at meetings (logical *OR*). More involved functions and/or states can have a substantial impact on the design and optimization of tours. We have started to explore classes of functions that will give network designers more freedom in the way they can arrange tours [17] and we are also looking at tradeoffs between the size of agents' states and the amount of interaction needed to solve a mission with the desired performance. We have also completed preliminary work on techniques that can be used to design tour-implementing trajectories in such a way that the cost of high meeting rates is minimized.

As presented in this paper, the notion of tours does not deal with agent failures, such as missed meetings because of delays or temporary failures. These are important in the context of mobile sensor networks and we have started to

investigate techniques that will let agents use meetings to adjust at runtime their tour trajectories and future meeting schedules.

References

1. Bartoš, R., Chappell, S.G., Komerska, R.J., Haag, M.M., Mupparapu, S., Agu, E., Katz, I.: Development of routing protocols for the solar-powered autonomous underwater vehicle (SAUV) platform. *Wireless Communications and Mobile Computing* **8**(8) (2008) 1075–1088
2. Fall, K.: A delay tolerant networking architecture for challenged internets. In: *SIGCOMM 2003*. (2003)
3. Ho, M., Fall, K.: Poster: Delay tolerant networking for sensor networks. In: *First IEEE Conference on Sensor and Ad Hoc Communications and Networks (SECON 2004)*, Santa Clara, CA (2004)
4. Zhao, W., Ammar, M., Zegura, E.: A message ferrying approach for data delivery in sparse mobile ad hoc networks. In: *ACM MobiHoc*. (2004)
5. Zhao, W., Ammar, M., Zegura, E.: Controlling the mobility of multiple data transport ferries in a delay-tolerant network. In: *IEEE INFOCOM*. (2005)
6. Burns, B., Brock, O., Levine, B.N.: MV routing and capacity building in disruption tolerant networks. In: *IEEE INFOCOM*. (2005)
7. Burns, B., Brock, O., Levine, B.N.: MORA routing and capacity building in disruption-tolerant networks. *Ad Hoc Netw.* **6**(4) (2008) 600–620
8. Balasubramanian, A., Levine, B.N., Venkataramani, A.: DTN routing as a resource allocation problem. In: *Proc. ACM SIGCOMM*. (2007)
9. Sozer, E.M., Stojanovic, M., Proakis, J.G.: Underwater acoustic networks. *IEEE Journal of Oceanic Engineering* **25**(1) (2000) 72–83
10. Akyildiz, I.F., Pompili, D., Melodia, T.: Underwater acoustic sensor networks: research challenges. *Ad Hoc Networks* **3**(3) (2005) 257–279
11. Cui, J.H., Kong, J., Gerla, M., Zhou, S.: The challenges of building mobile underwater wireless networks for aquatic applications. *IEEE Network* **20**(3) (2006) 12–18
12. Partan, J., Kurose, J., Levine, B.N.: A survey of practical issues in underwater networks. In: *WUWNet '06: Proceedings of the 1st ACM international workshop on Underwater networks*, Los Angeles, CA, ACM Press (2006) 17–24
13. Chappell, S.G., Komerska, R.J., Blidberg, D.R., Duarte, C.N., Martel, G.R., Crimmins, D.M., Beliard, M.A., Nitzel, R., Jalbert, J.C., Bartoš, R.: Recent field experience with multiple cooperating solar-powered AUVs. In: *15th Intl. Symposium on Unmanned Untethered Submersible Technology (UUST'07)*, Durham, NH (2007)
14. Haag, M.M., Agu, E., Komerska, R., Chappell, S.G., Bartoš, R.: Status packet deprecation and store-forward routing in AUSNET. In: *First ACM International Workshop on UnderWater Networks (WUWNet)*, Los Angeles, CA (2006)
15. U.S. Department of the Navy: The Navy Unmanned Undersea Vehicle (UUV) Master Plan (2000) <http://www.npt.nuwc.navy.mil/UUV/UUVMP.pdf>.
16. Popa, D., Sanderson, A., Komerska, R., Mupparapu, S., Blidberg, D., Chappel, S.: Adaptive sampling algorithms for multiple autonomous underwater vehicles. In: *Proc. of IEEE/OES AUV2004: A Workshop on Multiple Autonomous Underwater Vehicle Operations*, Sebasco Estates, ME (2004)
17. Chandy, K.M., Charpentier, M.: Self-similar algorithms for dynamic distributed systems. In: *27th International Conference on Distributed Computing Systems (ICDCS'2007)*. (2007)