Quality-of-Context Driven Autonomicity

Michael Breza, Department of Computing, Imperial College, London email: mjb04@doc.ic.ac.uk
Richard Anthony, Department of Computer Science, University of Greenwich, London
email: R.J.Anthony@gre.ac.uk

Julie McCann, Department of Computing, Imperial College, London email: jamm@doc.ic.ac.uk

Abstract

Optimisation in wireless sensor networks is necessary due to the resource constraints of individual devices, bandwidth limits of the communication channel, relatively high probably of sensor failure, and the requirement constraints of the deployed applications in potently highly volatile environments.

This paper presents BioANS, a protocol designed to optimise a wireless sensor network for resource efficiency as well as to meet a requirement common to a whole class of WSN applications - namely that the sensor nodes are dynamically selected on some qualitative basis, for example the quality by which they can provide the required context information.

The design of BioANS has been inspired by the communication mechanisms that have evolved in natural systems. The protocol tolerates randomness in its environment, including random message loss, and incorporates a non-deterministic 'delayed-bids' mechanism.

A simulation model is used to explore the protocol's performance in a wide range of WSN configurations. Characteristics evaluated include tolerance to sensor node density and message loss, communication efficiency, and negotiation latency.

1. Introduction

Wireless Sensor Networks (WSN) are becoming increasingly popular as a means of remotely monitoring an environment and collecting information that provides context to applications. The increasing popularity stems from advances in the underlying technology, in terms of physical size, on-board processing and storage capabilities, wireless communication range and battery technology. Simultaneously, the per-unit costs are falling, so that it has become economically feasible to deploy systems that have some redundancy.

Applications of WSN tend to concern monitoring a physical environment in which various characteristics of the environment itself, or of the actors that occupy the environment, are collected and provide a means of run-time contextualisation. Typical and diverse examples of monitored environments include care homes for sick or elderly patients, in which information concerning the location and behaviour of the occupants is used to identify 'alarm' scenarios [1]. The deployed systems need to fully self-manage as the patient is unable to carry out technical support. Another application, operating on a much larger scale (in terms of the number of sensors and their geographical spread) is the monitoring of changes to glaciers, in which location (more specifically movement over time) and temperature is used to measure the effects of global warming [2]. Although sensor node technology is improving, there remains the fact that resources are finite. So even if (for example) you double the battery capacity it will still run out. The lack of resources of sensor nodes are the biggest single limiter that restricts the functionality of deployed applications. Communication is one of the costliest aspects of sensor node behaviour in terms of power consumption. The communication protocol employed in the glacier example [2], mentioned above, was designed specifically to optimise for coverage as the sensor nodes eventually, inevitably, exhaust their power supplies. By dynamically self-configuring, nodes optimise multi-hop routing such that transmission power is kept low (by transmitting to physically close neighbours), yet at the same time it is necessary to avoid depleting the power of those nodes who forward messages. Ant Colony Optimisation is a technique that uses simulated pheromone trails, inspired by stigmergic communication in insect colonies. This has been used to dynamically find 'good' routes through sensor networks, see for example [3] and [4].

In addition to the constraints of the technology, the typical applications also introduce several constraints. These include: robustness, scalable deployment platforms; stability despite configuration change; and low communication latency because the applications can have a real-time aspect. Applications select sensor nodes dynamically on some qualitative basis, for example the quality by which they can provide the required context information. This is defined as Quality of Context (QoC) [5]. Here it is not sufficient to locate any sensor, instead it is required that sensors are selected based on the QoC that they can deliver; and there can be many different attributes of this quality (such as up-to-dateness, granularity - of position information for example, and accuracy).

This paper describes a highly efficient sensor network protocol, BioANS (Bio-inspired Autonomic Networked Services), which facilitates QoC-based dynamic selection of sensor nodes, whilst simultaneously meeting the requirements of efficient resource usage (especially in terms of communication), robustness (through its ability to choose among redundant sensors and its resilience to message loss), scalability and low latency.

Section two describes sensor networks and their associated challenges. The third section discusses engineering emergence, and its appropriateness to solving some of the problems of WSN. BioANS's approach to autonomicity is borrowed from its predecessor ANS [1], and this is explained in section four. Section five follows with a description of the BioANS protocol. The next two sections (six and seven) introduce the simulation with its assumptions that were run to test the performance of BioANS. Sections eight, nine, and ten give the experimental results. This is followed by a discussion of related work in section eleven, and then a conclusion.

2. Sensor Network Concepts

WSN comprise varying numbers of sensor nodes. These are typically homogeneous devices that each are tasked with monitoring some environmental aspect (even if the nodes are equipped with different types of sensors, the basic processing platform is likely to be the same). The nodes also form a communication network through which the individual nodes receive configuration instructions and through which the monitored information is delivered back to some designated external collection point - often referred to as a sink (this node might differ from the other nodes in several important ways: its location might be fixed, and it might have a wired power supply and network connection). Various techniques are used to achieve efficiency in such environments. Primarily message routing has to be optimised to keep the transmission power as low as possible and to minimise the effect of communication on third-parties (both in the sense of forwarding costs, and also in terms of wasteful interruptions when non-interesting packets are 'overheard'; listening costs almost as much a transmitting). To reduce power costs, low-bandwidth wireless links tend to be used. This places a premium on bandwidth, reinforcing the need to minimise both the number of, and size of, messages. Therefore another optimisation is to dynamically aggregate data at sensor nodes before sending. The characteristics of WSN thus place significant constraints on the design of the communication and application protocols deployed.

3. Engineered Emergence in WSNs

Natural systems such as insect colonies have evolved to solve problems similar to those experienced in sensor networks: in the sense that they have large numbers of individuals who each have only local knowledge of their environment and of the context in which their colony is operating at any moment. Communication has to be highly effective and messages must be passed through the colony using the individual actors not only as 'receivers' but also as 'routers'. This is one of a great many examples of how order and structure 'emerge' from the interaction of many individuals in natural systems, and over a large number of iterations such 'protocols' have become highly optimised.

The term 'engineered emergence' describes the purposeful design of interaction protocols so that a predictable, desired outcome is achieved at a higher level (i.e. Emerges) even though the individuals work with only local knowledge. Effectively the intention is to mimic behavioural aspects of the natural systems. See for example [6] and [7].

Our use of the term emergence describes higher-level states, patterns or other behaviours that arise in systems of numerous lower-level components that have local autonomy to interact with their neighbours. The individual components are typically quite simple and operate with only a local view of the system. Higher-level behaviour cannot be predicted by examining the individual components or their behaviour in isolation. There are many examples in nature where highly optimised global behaviour emerges from these kinds of systems [8]. The science of emergence is described in [9] [10] [11]. One additional important characteristic of natural emergent systems is that they embrace random influences in their environments; actually incorporating the randomness into behavioural protocols - a specific example is symmetry breaking.

In contrast to the lightweight approaches that have evolved in the natural systems, traditional design of distributed computer applications focuses on strict protocols, message acknowledgements and event ordering. Each message and event is considered important and randomness is generally undesirable imposing sequenced or synchronised behaviour which is generally deterministic. Such a design paradigm can lead to inefficiency, for example through large numbers of transmitted messages (a significant fraction of which might be for handshaking and error recovery purposes) and additional communication latency. In addition, typically some of the low level messages do not directly contribute to correct application behaviour at higher levels [12].

Natural biological systems however are fundamentally non-deterministic and there are many examples of large-scale systems that are stable and robust at a global level; the most commonly cited examples being drawn from cellular systems and insect colonies. BioANS requires that a small number of appropriate quality bids are elicited from sensor nodes (service providers) in potentially very large systems. In this application domain it is important to minimise the total amount of communication, the latency of service negotiation, and also to preserve the battery power at each sensor node.

The BioANS protocol has been designed to purposely mimic some aspects of interactions in natural systems, to yield sensor net communication that is simultaneously scalable, self-configuring, reliable and efficient. The protocol employs emergence concepts to achieve scalable and robust negotiation (i.e. Dynamic selection of sensors based on their QoC characteristics). The negotiation protocol needs to be stable and predictable in terms of its higher-level behaviour (i.e. a suitable context provider needs to be located within a reasonable time-frame, and whilst making efficient use of messages), although the low-level behaviour (such as the actual interactions with and between sensor nodes, and the ordering of events such as message transmission) has elements of randomness and can thus not be precisely predicted.

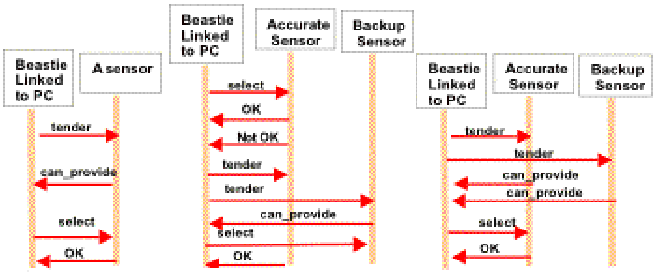
The delayed-bid mechanism, described in detail in [12], is employed in BioANS to introduce randomness into the timing of responses to QoC queries. By using this mechanism, the high number of bids sent as responses to a QoC request in large systems are spread out in time (each node waiting a short random time before responding to a request). As sensor bids are evaluated in the order they are received at the requester, the randomness makes the system non-deterministic in terms of which sensor will be the first to fulfil the QoC requirement and thus be selected. As soon as an acceptable sensor has been found, the requester sends a ’Stop-Bids’ message which has the effect of cancelling any pending bids that have not at that point been transmitted. See [12]for empirical analysis results.

The original ANS protocol [13] was reliable and simple, but had scale-related high communication overheads. The randomness and non-determinism that have been purposely introduced into the successor protocol, BioANS, enable it to be used in high-density networks whilst retaining protocol simplicity. This is shown by the experimental results presented in this paper.

4. Approach to Autonomicity using the original ANS

A perceived notion of context and quality of context (QoC) drives the autonomic behaviour of ANS. Autonomicity in systems is defined as the ability to be self managing, often referred to as self* [14]. Context refers to the circumstances, situations, or environment in which a computing task takes place [15]. A common example of quality of context is the quality of the location of the user or objects of interest. Location can be tracked in smart homes using different sensor types such as ultrasonic badges, or RFID tags [16]. There will be a difference in the quality of location information depending on the type or state (battery power, damaged or, obscured sensing apparatus) of the sensor. As an example, ultrasonic badges can determine location with a precision of up to 3 cm, while RF lateration is limited to 1-3 m precision. The quality needed and its degree will be application specific.

QoC is used by ANS to choose a suitable service among those available when delivering requested information to an application. While different types of information will have QoC attributes specific to them, certain attributes will be common to most types of information such as: Precision, Probability of correctness, Resolution, Up-to-datedness and Refresh rate [17]. In ANS sensors need to specify QoC attributes for the information they deliver. The QoC from a given sensor is susceptible to change over time and therefore needs to be autonomously managed.. ANS describes the data/function provided by a sensor as services. Services are represented as a named list of ’commands’ and ’events’. An application then calls on the service by naming it and its required QoC. Figure 1 shows the numbers and types of messages the protocol uses in a message diagram. Services are selected through a process called ’tendering’. When an application needs a service it broadcasts a ’request’ command containing the name of the service (such as temperature) and its preferences for the QoC attributes. Devices within range and able to offer the service use a ’utility function’ to calculate their ability to deliver the information at the requested QoC. The utility function returns a single signed integer called ’closeness’. If the value meets a certain threshold of the



(a) Self-configuring (b) Self-healing (c) Self-optimisation

Figure 1. A message diagram of the ANS protocol.

requested QoC, it is sent back to the requesting device to allow it to decide which device to use. Regular 'retendering' allows requesters to autonomously adapt to the best sensor available, take advantage of any new devices joining the network, and recover from sensor failures. All QoC attributes are scaled and translated when a service is defined so that they are all of the same order of magnitude [13].

The utility function works by treating QoC as a point in an n-dimensional space (n being the number of QoC attributes). Available and requested QoC attributes are plotted in n-dimensional space. The utility function then returns the the distance from the requested point to the available point also referred to as 'closeness'. The requester will choose the first device with a positive 'closeness' since that device will be at least as good as the application wants. If nothing sufficient is returned in a given time period then a list of all of the sensors who have responded will be checked. The sensor with the least negative closeness will be chosen since, while not sufficient, it is the least bad and for many applications tolerable.

5. The BioANS Protocol with engineered emergence

BioANS enables adaptivity through the selection of sensor(s) based on the quality of information that they can offer. Utility functions express the information requesters QoC requirements such as accuracy and up-to-datedness. Theses requirements encapsulate the externally deterministic behaviour needed. However, it is not important that the lower-level behaviour be deterministic. This gives great freedom in which sensor provides the information, or how the sensor(s) are selected. BioANS takes advantage of cheaper (less communication intensive, less synchronous and self-regulating) non-deterministic communication strategies inspired by biological systems such as insect colonies.

Certain techniques are found in natural systems that use non-deterministic strategies. Common are: randomness (such as in timing mechanisms) and attributing low-value to individual events, actors and messages. These systems are unaffected by message loss, or events going unobserved or unordered. BioANS uses the delayed-bid mechanism [18] to add a random timing component spreading out in time the responses to QoC requests.

When a sensor receives a request, it computes its QoC value based on the requested utility function. Once the node has determined the QoC it can deliver, it transmits a reply to the requester. Broadcasting the request is efficient with respect to the simplicity of the protocol and the total number of messages, but introduces a synchronisation point. Receipt of a request implicitly invokes a certain response at each node. As system size increases, near simultaneous reply messages present a problem by congestion of the communication channel. This can increase application latency and deny communication to another applications. Often only one or a small number of sensors are required to provide information to a particular application. If all the sensors respond this wastes communication bandwidth, battery power at the sensor nodes, and the processing of replies at the requester.

BioANS solves this problem by the injection of a random delay locally determined at each sensor. This spreads out the replies and directly reduces the network congestion problem. It also provides an opportunity for significant reduction of messages. This is because the response messages are dispersed in time and the requester node can process some messages before others have been sent. Once a sufficient response with the appropriate QoC parameters has been received, a Stop-Bids message is sent. This cancels all of the unsent responses at sensor nodes. Some unwanted messages may already be in transmission, but the large majority of unnecessary messages can be stopped.

All messages in BioANS have low-value, and the protocol tolerates the loss of any individual message. If a request receives no replies (within the maximum random time delay for replies) the request message is deemed lost and is repeated; if a request message is not received by an individual sensor node it simply does not participate in the bidding. Likewise individual response messages are of low value. If a Stop-Bids message is lost the protocol still functions completely correctly, it just looses efficiency as the reply-quenching savings are lost. Low value messages make BioANS extremely robust to network failures and dynamic network conditions, and allow BioANS to scale well.

6. Simulation of BioANS

To further understand BioANS we model it as a discrete event simulation, using observed performance from BioANS's predecessor ANS implemented on the Beastie sensor network [1]. The ANS protocol has been implemented for two applications; patient monitoring in the home, and large scale building usage monitoring. We want to examine the trade off between protocol overhead and performance. By overhead we refer to all of the communication not concerned with performing work. Performance we define as the ability of a requester to receive its desired QoC, and for what percentage of overall run time.

As stated above, we have implemented an initial version of ANS on a real sensor network of 5 nodes. From this we have gathered our main parameters to drive the BioANS simulations. Simulation is necessary to examine the effects of the protocols as systems scale up to thousands of nodes, because implementation and deployment of actual sensors at this scale in controlled realistic conditions is currently highly problematic and costly.

The first set of experiments includes location and reception radius limiting the number of sensors that can hear a given requester. These add a realistic set of constraints that the system will face in deployment. First we focus on the percentage of time a requester receives its desired QoC as the deployment area of the network increases, and the sensor density decreases. Then we choose one density, and see how BioANS scales at this density.

The second set of experiments looks at how BioANS performs as the failure rate increases, and as how performance degrades as the network fails, and resumes as the network recovers. Theses experiments model sensor networks in highly dynamic environments, like mobile nodes, or single deployment situations, where the sensors are deployed, and then left to run until their power is exhausted. An example of this is monitoring glaciers, where the nodes are deployed over a large area by being thrown out of a helicopter, and left to run until they fail. After failure, more nodes can be thrown out again, to resume network operation.

7. Assumptions

BioANS assumes that the sensors themselves choose if they are going to respond to a re-tender request. The criterion they use is to calculate if they can provide at least 60% of the QoC asked for. If not, the sensor remains silent. This heuristic was derived from previous experiments.

The duty-cycle between re-tenders is an important consideration in ANS; a large duty-cycle between re-tenders lowers protocol overhead at a cost of resilience to sensor failure. In these simulations the duty-cycle was 10 queries. Packets sent to the sensor for readings and packets with sensor data were counted as work packets. All others were considered protocol admin packets, i.e. overhead.

To faithfully simulate the dynamic nature of the sensor network, sensor failure and replacement is built into the experiments. The sensor failures are exponentially distributed with a mean of one failure every 5000 time units, and a failure triggers a replacement of one or more new sensors with a replacement time lag exponentially distributed with a mean of 10000 time units. When the sensors are replaced, the number of new sensors is geometrically distributed with a mean sensor node count of 1.6. When a sensor used by a requester fails, the requester immediately starts the re-tendering process. The QoC of the new sensor is completely random, and it has a ten percent chance of having a variable QoC that will decrease by 33% for each user. The advertised QoC of the sensors is assumed to be correct. All sensors serve data that is of interest to all of the requesters, but different requesters want different QoC. Each requester requires only one sensor at a time.

Location information in the model constrains the number of sensors that can respond to a (re)-tender request. This is due to communication range.

To facilitate exploration of the effect of sensor node population density on the protocol's performance, a simple means of diluting the sensor nodes is used.

A 2-dimensional grid of cells is used to simulate the area of deployment. One or more node can reside in a given cell. Consequently, given a constant sensor population, the size of the grid determines the population density of the nodes. For a given node population, a larger grid will have a lower density than a smaller grid. In each experiment (i.e. change in grid size) the nodes are distributed across the grid with a uniform random distribution.

The term 'density factor' is defined as the mean number of sensor nodes within a 100 cell area of the grid; i.e. a density factor of 1 implies that there is an average of one sensor node per 100 grid cells. Each type of node has a wireless range of 20 cells, with the assumption that there is no interference to limit range. Thus its communication range covers $400\pi \approx 1257$ cells. Figure 7 illustrates a typical sensor node distribution with density factor 0.4 (for clarity, the lines in the diagram are drawn at a distance of 10 cells apart). On average this density factor equates to 5 sensor nodes being in communication range of a requester node. The simulation is

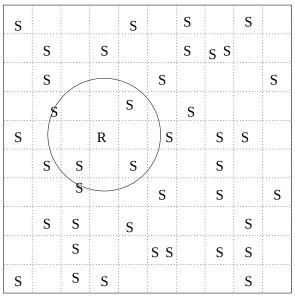


Figure 2. A random sensor node distribution with a density factor of 0.4. Sensor and requestor nodes are represented by the symbols 'S' and 'R' respectively.

initialised such that at the beginning each requester can hear at least one sensor. Failures and the constrained number of sensors available can leave a requester in a state with no sensors available.

The time between packet arrivals is affected by the random back-off algorithm used by the radio link layer BNET [19] that ANS was built on. The arrival of responses from requests (non-random arrivals) were normally distributed with the means and standard deviations taken from the packet traces in [1]. Packet loss, collision and traffic management problems were not modelled, because we assume this to be handled by BNET.

Three metrics were measured in these experiments: the average percentage of time in the simulation run that the requesters got their requested QoC or no sensor; the average ratio of work related packets sent and received by the requesters (the inverse of the protocol's overhead) and negotiation time, from request packet being sent out, to the sending of the select sensor packet. The simulations were run for 10,100,000 time units, measured at equilibrium with a restart at 100,000 time units. Results are generated as an average over all requesters, each over ten runs.

8. BioANS performance at various node densities

Given a range of 20 units, the density of the sensors in the deployment area will affect the performance of the network. The performance we choose to measure is QoC. Figure 3 shows us the average QoC received by 50 requesters in a network of 500 sensors. The x axis shows the density factor of the monitored region occupied by the nodes (sensors and requesters). As area in the monitored region occupied by the node decreases, so does the density.

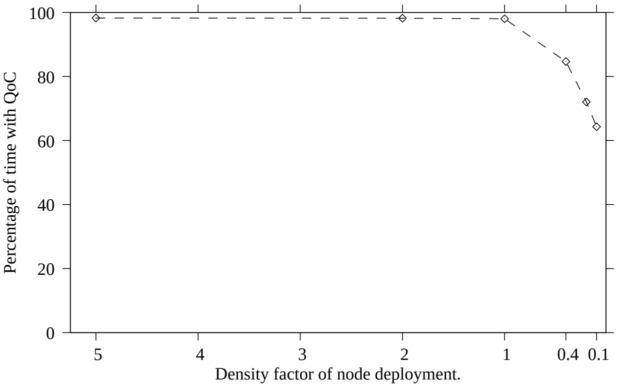


Figure 3. Relationship between average percentage of time requesters got QoC and density factor.

The average QoC obtained by the requester is almost 100% until the density factor fall below 1. At that point the average QoC begins to drop very quickly. At density factor 0.4 the requesters are only getting their QoC an average of 85% of the time. By density factor 0.2 that figure has dropped to 72%, and 64% by density factor 0.1. A further look at the raw data not graphically presented here shows that with a density factor of 1 a requester can hear an average of 8 sensors. A density factor of 0.4 reduced that to an average of 2 sensors. These figures are below the expected mathematical average because of sensor failure in the network.

9. BioANS performance at a fixed node density

Given the results above, we decided to test the performance of BioANS as it scales in network size. The density factor was set to 0.4 to examine the performance as it begins to deteriorate. The results are summarised in figures 4, 5 and 6.

These experiments show that BioANS scales well up to 1000 sensors and 100 requesters (the maximum population tested in this experiment). Figure 4 shows that the average QoC received is the same for all network sizes. The average time requesters had no senor fluctuated a bit, was was consistently below 1% of the time. The inverse of the overhead, represented here as percentage of work packet traffic to overall packet traffic, is constantly high (overheads consistently low), see figure 5. Latency, represented here as average time a re-tender took to complete, is also low, and consistent among all of the populations tested, see figure 6

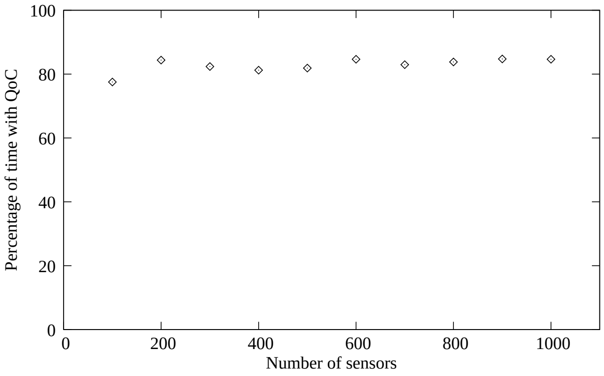


Figure 4. Average percentage of time requesters got QoC as network size increases with fixed density.

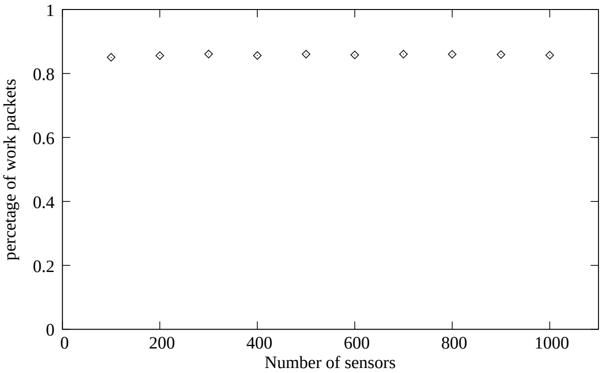


Figure 5. Ratio of work packets to overall packets.

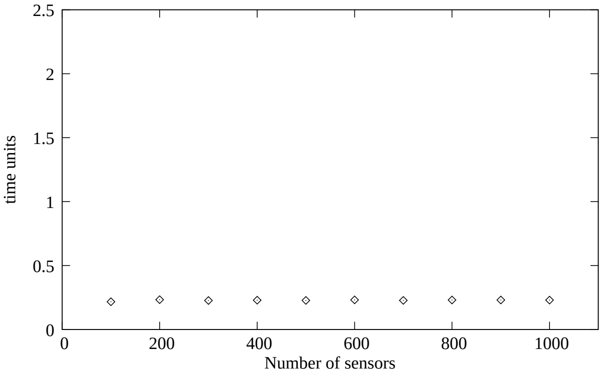


Figure 6. Average re-tender time as network size increases with fixed density.

10. BioANS performance with failures

To test the robustness of BioANS in the face of a dynamic network, we run experiments where we vary the failure and resumption rate of the network. We also watch the degradation of the network as all of the nodes fail without replacement, and as the network recovers from a state of no sensors, to a full population of sensors.

All of the previous experiments are run with a failure rate of one failure every 5000 time units distributed exponentially. The resumption rate is half that, with new nodes being added every 10000 time units, but added in batches (one or more) with a geometric distribution with a mean of 1.6. The previous experiment shows that the network is stable as the size increases. In this experiment, we increase the failure and resumption rate from one every 5000 time units to one every 10 time units. The resumption rate is half the failure rate, and the new node batch mean is always 1.6. The population of the network is always 1000 sensors and 100 requesters, and the density factor is fixed at 0.4 The results are summarised in figures 7, 8, and 9. Figure 7 shows the average received QoC remaining consistent as the failure rate increases. Figure 8 gives a close up of the graph at the more frequent failure rates of one every 100, 50 and 10 time units. Even with the high frequency, the received QoC remains consistent. Similar behaviour is observed for percentage of time a requester has no sensor, the ratio of work packets to overall packets (inverse of the overhead) in figure 9, and the average time for a re-tender to complete. These results are not surprising since BioANS uses a frequent re-tender method. At the same time BioANS manages to keep low overheads (see figure 9).

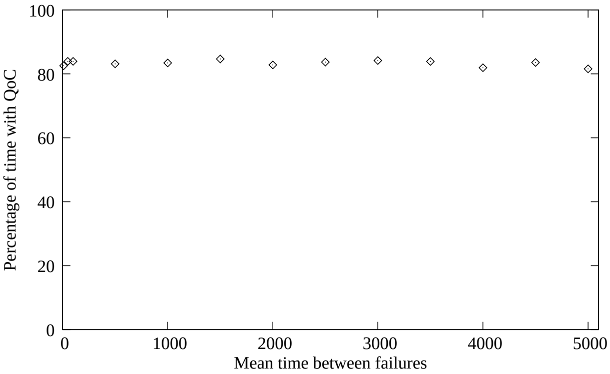


Figure 7. Average percentage of time requesters had QoC in relation to mean time between failures.

Our final experiment looks at how average QoC degrades as the sensors fail without replacement, and re-

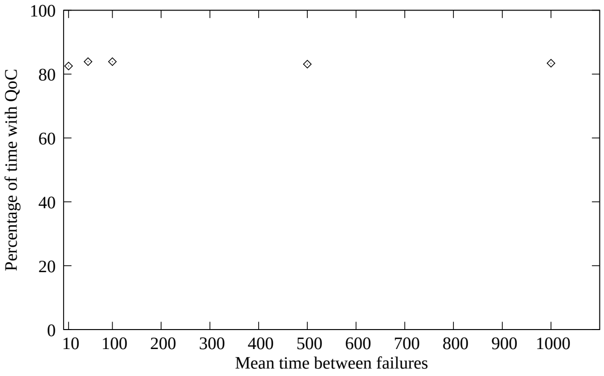


Figure 8. Subset of x axis from 0 to 1000 of average percentage of time requesters had QoC in relation to mean time between failures.

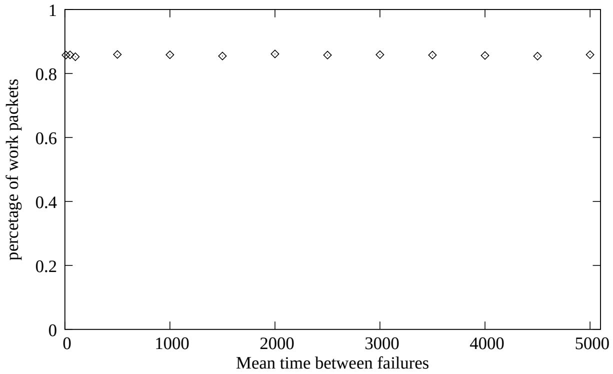


Figure 9. Ratio of work packets to overall packets in relation to mean time between failures.

sumes as new sensors are are added. The experiments were run with a density factor of 0.4, varying the sensor node population from 0 to 1000. The failure and recovery rates are exponentially distributed with a rate of one every 5000 time units. Each experiment was run ten times, and all ten runs are shown to show the deviation of results in figure 10. The results show that BioANS degrades and recovers gracefully. The collective results of the above experiments show that the bio-inspired approach of low cost messages works well to make a stable, robust protocol with low overheads. The protocol is very resilient to highly dynamic network conditions.

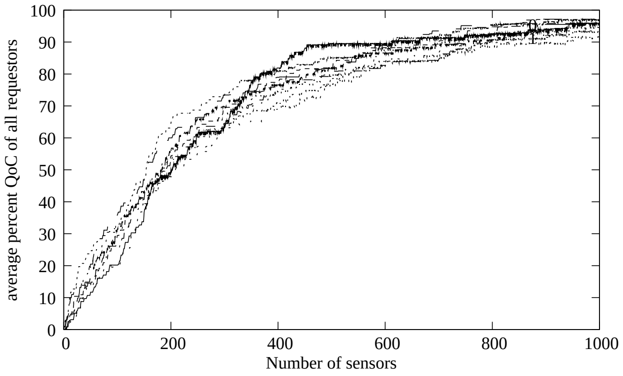


Figure 10. Average percentage QoC as network resumes.

11. Related Work

Utility based service selection is gaining interest in the Self-Adaptive Computing community. Much of this work prescribes a given utility of service per application without providing a single generic framework that is shared between applications. The applications request a service using bulky middleware that buffers utility information and drives the self-management of the system therein. This middleware solution is not suitable for resource scarce sensor networks and this central component limits he scalability required for larger scale 'smart dust-like' applications as found in environmental monitoring.

Rajkumar et al. [20] propose a resource allocation model for QoS management within a single system. Resources include CPU utilisation, memory consumption, network bandwidth and latency. Each application delivers to the system the minimum resource requirements it has, plus a utility function that returns the increase in performance given additional resources. The system then allocates resources to each application such that the total system utility is maximised.

The Context Toolkit [21] is a framework aimed at facilitating the development and deployment of context-aware applications. Similar to the work presented here, it abstracts context services, e.g. a location service, from the sensors that acquire the necessary data to deliver the service. Again, the Context Toolkit allows sharing of context data through middleware, but has the advantage that this is distributed over the base-stations or nodes that have higher resources. Again, unlike our work this middle-ware infrastructure is quite bulky thus not suitable for sensor applications where the infrastructure is deployed on the actual sensor nodes. Moreover, it does not provide any self-adaptation in terms of allowing applications that enter the distributed environment to discover available services: the location of context services (IP address and port number) has to be known in advance. Also, there is no mechanism that allows context services to adapt and react to failure or degradation of the underlying sensor infrastructure, e.g. by switching to an alternative means of acquiring the same type of context.

Cohen et al. have proposed iQueue [22], a data-composition framework for pervasive data. iQueue allows applications to create data composers, specify a composer's data sources using functional data specification, and specify a composer's computation. Similar to our Requester, the iQueue run-time system selects data sources satisfying the data specifications, dynamically re-selects data sources as appropriate. The goal is very similar to ours, although their middleware has not been designed in a lightweight fashion. They use a mechanism similar to our periodic retender request, in that a data source issues advertisements periodically, but also whenever properties of the data source, e.g. quality of information, change. It would appear that they use Boolean predicates over the values of the properties of the data source. Instead, we present a mathematical model based on application's wishes that evaluates each applications quantitative satisfaction with regard to any particular data source. These centralised solutions are not suitable for sensor networks as many of the nodes are too small to carry out this burden and it introduces a central point of failure to the system. Therefore we aimed to carry out the same functionality in a more lightweight and decentralised way, hence our bio-inspired approach.

BioANS is an enhancement of the wireless sensor network protocol ANS [13] with the addition of the stop-bid mechanism . ANS is a much simpler protocol, and proved very robust. Its problem is high communication overhead when the network grows to larger sizes. BioANS is proposed as a solution to this scaling problem.

The inspiration for BioANS is stylistically bio-inspired. In [23] an emergent leader election algorithm is given whose communication style is based on the mechanics of pheromone based communication. Pheromone communication is essentially a broadcast, with no guarantee of delivery. The emergent leader election algorithm uses the inherent non-determinism of unreliable communication to make a very efficient algorithm for large size distributed systems. In [12] a similar use of the non-determinism of un-reliable communication is used to recruit idle nodes for distributed computation. Because of the similarity between the style of communication used in these works, and the type of communication we are restricted to in sensor nets, the optimised ANS is heavily inspired by these algorithms.

As far as we are aware there is no protocol for sensor networks that we can use to compare our work directly. Recall that BioANS is essentially a service-oriented protocol designed to be lightweight while implementing self-optimisation and reliability through service redundancy. There is a large body of service oriented/service selection research in the large scale computing field; mainly focusing on semantics and performance at that scale. Conversely, BioANS was designed with a low footprint and operational overhead for tiny sensor node devices where the emphasis is on minimising overheads and achieving scalability in the thousands rather than tens of nodes. The main body of emergent-like algorithms that exist for sensor networks mainly focus on the reliable and timely delivery of messages from a given node to a sink or between nodes and not reliability through service selection which sits at a higher level of abstraction (e.g. LEACH and the large body of subsequent related work [24]). Incidentally, BioANS does not assume a reliable underlying network layer and assumes messages can be lost. As a result, there are no results, other than our own obtained from the initial protocol runs, to which we can directly compare the figures we present here.

12. Conclusion

BioANS employs emergence engineering concepts to satisfy the demanding requirements of large-scale applications deployed on sensor networks. Specifically these requirements are scale, robustness, low latency negotiation and efficient resource usage. BioANS describes services provided by the WSN as contexts and the quality (QoC) with which this context can be delivered. Applications are composed of sets of calls to retrieve context (sensor data) from those sensors that provide the appropriate quality (accuracy, up-to-dateness etc.) of such data.

This paper has evaluated the extent to which the introduction of these emergence mechanisms, such as delayed bid, contribute to the overall quality of the protocol.

To this end we took measurements obtained from a smaller scale WSN running the original ANS and applied them to a simulation model to observe how the protocol would operate under extreme conditions such as failure or large numbers of nodes, which further allowed us to carry out partial validation of results.

The experiments have evaluated the performance of the BioANS protocol in terms of: its resilience to node failure; the effects of changes in sensor node population and density; the efficiency of communication in terms of the ratio of work packets to overhead packets; and the various factors that impact on the QoC received by requesters.

The average received QoC was measured as the density of the nodes in their deployment was reduced. At the given range of 20 units, the nodes offered above 94% average QoC over time as long as the density factor was greater than 1.0. At density factors less than 1.0, the QoC begins to reduce. The performance of BioANS was measured at a density factor of 0.4 and was found to scale without change in performance up to 1000 nodes. BioANS was also tested with increasing failure rate of the senors, and found to offer the same performance even with very high failure rates, i.e. the performance was found to gracefully degrade as the the sensor pool shrinks due to failures. In conclusion the experiments demonstrate that the bio-inspired optimisations of BioANS provide a stable, highly scalable and robust protocol that has general applicability to a wide range of applications in sensor networks and similar resource constrained domains.

References

[1] J. McCann and A. Hoskins, “Proof of concept adaptivity and performance benchmark results,” tech. rep., Imperial College London, May 2006.

[2] K. Martinez, J. Hart, and R. Ong, “Environmental sensor networks,” *IEEE Comput.*, vol. 37, pp. 50–56, August 2004.

[3] S. Okdem and D. Karaboga, “Routing in wireless sensor networks using ant colony optimization,” in *First NASA/ESA Conference on Adaptive Hardware and Systems*, IEEE, 2006.

[4] W. Cai, X. Jin, Y. Zhang, K. Chen, and R. Wang, “Aco based qos routing algorithm for wireless sensor networks,” in *Lecture Notes in Computer Science*, Springer Berlin/ Heidelberg, August 2006.

[5] T. Buchholz, A. Kupper, and M. Schiffers, “Quality of contex: What it is and why we need it,” *10th Workshop of the HP OpenView University Association (HPOVUA03)*.

[6] R. Anthony, “Emergent graph colouring,” *Engineering Emergence for Autonomic Systems (EEAS), First Annual International Workshop, at the third International Conference on Autonomic Computing (ICAC), Dublin, Ireland*, pp. 4–13, June 2006.

[7] O. Babaoglu, G. Canright, A. Deutsch, G. D. Caro, F. Ducatelle, L. Gambardella, M. J. N. Ganguly, R. Montemanni, A. Montresor, and T. Urnes, “Design patterns from biology for distributed computing,” in *ACM Transactions on Autonomous and Adaptive Systems*, vol. 1, pp. 26–66, September 2006.

[8] M. Gell-Mann, *The Quark and the Jaguar: Adventures in the Simple and the Complex*. Abacus, London, 1994.

[9] S. Johnson, *Emergence: The connected lives of Ants, Brains, Cities and Software*. Penguin Press, London, 2001.

[10] J. Casti, *Complexification: Explaining a Paradoxical World Through the Science of Surprise*. Abacus, London, 1994.

[11] R. Genet, *The Chimpanzees who would be Ants: A Unified Scientific Story of Humanity*. Nova Science Publishers Inc, New York, 1997.

[12] R. Anthony, “Engineering emergence for cluster configuration,” *Journal of Systemics, Cybernetics and Informatics*, vol. 4, no. 3, 2006.

[13] J. McCann, M. Huebscher, and A. Hoskins, “Context as autonomic intelligence in a ubiquitous computing environment,” *International Journal of Internet Protocol Technology (IJIPT) special edition on Autonomic Computing*, 2006.

[14] J. Kephart and D. Chess, “The vision of autonomic computing,” *Computer -IEEE Computer Society*, vol. 36, no. 1, pp. 41–51, 2003.

[15] M. Roman, C. Hess, R. Cerqueira, A. Ranganathan, R. Campbell, and K. Nahrstedt, “A middleware infrastructure for active spaces,” *IEEE Pervasive Computing*, vol. 1, no. 4, pp. 74–83, 2002.

[16] J. Hightower and G. Borriello, “Location systems for ubiquitous computing,” *IEEE Computer*, vol. 34, pp. 57–66, August 2001.

[17] A. K. Dey, G. D. Abowd, and D. Salber, “A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications,” *Human-Computer Interaction*, vol. 16, pp. 97–166, 2001.

[18] R. Anthony, “Emergence: A paradigm for robust and scalable distributed applications,” *1st Intl. Conf. Autonomic Computing (ICAC), IEEE, New York*, pp. 132–139, 2004.

[19] “Ans over bnet.” http://www.doc.ic.ac.uk/ asher/ubi/ans/bnet.html.

[20] R. Rajkumar, J. L. C. Lee, and D. Siewiorek, “A resource allocation model for qos management,” in *Proceedings of the 18th IEEE Real-Time Systems Symposium (RTSS ’97)*, p. 298, IEEE Computer Society, 1997. ISBN 0-8186-8268-X.

[21] D. Salber, A. K. Dey, and G. D. Abowd, “The context toolkit: aiding the development of context-enabled applications,” in *Proceedings of the SIGCHI conference on Human factors in computing systems*, pp. 434–441, ACM Press, 1999. ISBN 0-201-48559-1.

[22] N. H. Cohen, A. Purakayastha, L. Wong, and D. L. Yeh, “iqueue: A pervasive data composition framework,” in *Proceedings of the Third International Conference on Mobile Data Management (MDM)*, pp. 146–153, January 2002. URL http://www.research.ibm.com/syncmsg/MDM2002.pdf.

[23] R. Anthony, “An autonomic election algorithm based on emergence in natural systems,” *Integrated computer-aided engineering*, vol. 13, no. 1, pp. 3 –22, 2006.

[24] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan, “Energy-efficient communication protocol for wireless microsensor networks,” *System Sciences, 2000. Proceedings of the 33rd Annual Hawaii International Conference on*, p. 10, 2000.