Simple Solutions for the Second Decade of Wireless Sensor Networking

Michael Breza, Pedro Martins , Julie A. McCann, Evangelos Spyrou, Poonam Yadav, Shusen Yang
Department of Computing
Imperial College London, UK
{*mjb04, pm1108, jamm, es309, pooyadav, sy709*}*@doc.ic.ac.uk*

At 2010, we are at the effective end of the first decade of Wireless Sensor Network (WSN) research and that the aspects such as energy efficient routing have pretty much had their day as the sensing, computing and energy technologies have now moved on. There now seems to be a consensus in the field as to what the next challenges are. This paper will describe some of those challenges for the next decade and discuss some of the key impediments hindering WSN research. From this we report a small experiment whereby we try to take on board some of the critiques we put forward to see what we come up with. To this end, we indeed show how a simple solution can help to solve many of the problems we list, and in doing so, highlight some of the difficulties in keeping to our own recommendations regarding the future direction of WSN research.

Wireless Sensor Network, Shared Networks, Bio-Inspired Algorithms

1. INTRODUCTION

Engineering design and hardware technology advances of the 1990's led to dramatic reductions in the size, power consumption and costs of digital circuitry, as well as improved wireless communications and Micro Electro Mechanical Systems (MEMS, i.e. sensor devices). From this soup came very compact computing nodes each containing one or more sensors, computation and communication capabilities, and a power supply. This, optimistically termed Smart Dust, is what we simply call Wireless Sensor Networks (WSN) today.

In one of the seminal WSN papers from 1999 WSN nodes were described as autonomous, potentially mobile and exhibiting emergent behaviour (Kahn et al. 1999). Sensys has emerged as one of the premier WSN conferences. Its 2002 call for papers succinctly summarises the initial vision of the first decade of Wireless Sensor Network research as being composed of "distributed systems of numerous smart sensors and actuators connecting computational capabilities to the physical world have the potential to revolutionise a wide array of application areas by providing an unprecedented density and fidelity of instrumentation".

We are now a decade down the line and it is interesting that the WSN hardware proposed and built ten years ago (bar transceivers) has not matched predictions given the scale of change in general computing over these years. What has changed is the sheer numbers of researchers that have joined the field, yet on the whole, actual and/or long term physical deployments are still few and far between.

Fast forward to Sensys 2009 panel discussion regarding the next ten years of WSN challenges. What was noted from this, and other more recent WSN gatherings, was that we have moved away from the MEMS focus and actuation seems to have been sidelined altogether. We still rely on the impractical assumptions that all nodes and their operation are homogeneous and that a single application is executed on the network at a time. Further, guaranteed real-time wireless communication is no closer to reality and networking aspects such as congestion control and Quality of Service differentiation are only beginning to come to the fore. Pister (one of the forefathers of smart dust) recently mentioned in a personal conversation that academics are generally allergic to leveraging fine-grained time syncronisation to enable TDMA and coordinated channel hopping, which can greatly improve reliability Furthermore, with a decade of focus on energy conservation and routing we still rely predominantly on battery power and cannot route reliably over a dense network. Other discussions have surrounded the need to account for field operator usability (i.e. it is not computer scientists deploying these nodes) which has HCI implications. Further, better data mining of the sensor data streams, heterogeneity and formal validation for safety were also being discussed. The next recommendation is closely related to this. Sensors are to be used by domain scientists and

Proceedings of ACM-BCS Visions of Computer Science 2010

they want to know about the physical world. Computer Scientists are used to operating in the relatively closed world of computing which models the physical world. But the physical world is difficult to interact with. Until now many of the WSN research experiments would report performance, packet loss, time to converge and reliability metrics. Yet, many of us have used solutions that were published as achieving 90% reliability only to find they do not work so well in the wild. Timing, missing data, sensor calibration are all very important aspects that are not really being tackled by the WSN community. Recall that one of the proposed future areas of WSN research included actuation. Consider the degree of reliability one would need to be able to effect the physical environment; it is no wonder that that actuation has been sidelined to some extent as we are not at the level of reliability required yet. Furthermore, once we solve many of the aforementioned problems then we need to look at scaling the solution and this is not only in terms of protocol performance but system maintenance and upgrade. Again this highlights that it is time we looked at being able to better understand behaviour and future inter-system interactions if we wish to scale and give a reasonable lifetime to these systems. So let us get real.

The preceding paragraph is a whistle stop tour of where the issues regarding WSN research lie today. The problem with all this is that even with those remaining challenges, external perception is that the subject has little else to say and that it has let down the non Computer Science community; the very people who were going to be the primary benefactors. In turn this perception is very dangerous to the WSN community as it triggers less funding initiatives and venture capitol opportunities. Perhaps this pessimism is the forbearer to a turning point in which new life can be breathed into the subject. To this end, we believe that there are three things WSN researchers need to remember to break free of last decade ideas: think out of the box, get real and keep it simple. The Berkeley Mote and its TinyOS was a great device for kicking off the subject but it has since become somewhat de facto and this has meant that there is a perception within parts of the community that one must use them if one is to get published. Certainly it has been our experience that it is much easier to compare our much more improved routing algorithm if the old routing algorithms are lying around in some TinyOS library somewhere. However, there remains a mismatch between the kit that is being deployed and the applications that motivate the research. Why bother squeezing an algorithm with associated management code into a tiny node capable of 8MIPS and with 4KB for it to be packaged in a 50 cm^2 weatherproof box that could fit a larger equally priced tiny Linux box. So lets think out of the box; drop what has become WSN research shackles and move on. Finally, it seems that the most successful actual deployments in the wild are actually much simpler in terms of hardware and software than your typical academic proposed system. Given the close coupling with the physical world and the extra complexity it brings, surely we should keep the system simple. Yet again the Computer Science community while aiming to simplify the problem, sometimes end up complicating it (just look at operating systems with all their layers and layers of layers). The WSN community needs to keep it simple.

So with this in mind, we set about looking at some of the WSN challenges and as an experiment see what solutions we come up with. To this end, we begin by describing our problem domain in Section 2 followed by providing an overview of the solution(s) we came up with in Section 3. Section 4 provides some background to the details of our solution while listing some related work. Section 5 describes the initial analysis of the solution with section 6 discussing the results. Section 7 concludes both in terms of our results and our approach to solving the problem.

2. DENSE WSN FOR ENVIRONMENT MONITORING AND EVENT DETECTION

An example application area that presents us with many interesting challenges is that of dual purpose dense networks of sensors. In this application, we must tackle the network density, heterogeneity, multi-purposing of a single network, reliability (for actuation), scale challenges mentioned in the previous section.

Applications of such an architecture are very much in demand today, especially in in the combined areas of environment monitoring and event detection (which are usually treated individually). There are many examples of such applications in the field; here we present two: gas detection in mining and avalanche detection.

Gas explosion detection in Coal Mining: In coal mines, the advance prediction of explosions, identification of excess gas concentrations, and prediction of structural failures (such as wall collapse), along with reporting the location and timing of such events constitute highly important tasks (Wang et al. 2007). As is the identification of the event in order to prevent false alarms. Actuation here is in the form of both audio and visual alarms; evacuation path display systems, as well as location and time indicators. Therefore, a robust reliable sensor-actuator network of nodes is required that not only monitors the environment, but can detect events and relay those event messages rapidly to effect change (actuation).

Avalanche prediction: Traditionally, an Avalanche Predictor manually checks snowpack conditions by digging snow pits and keeping track of layers as they develop and metamorphose over time. Again, this would require that dense networks of sensor nodes be

deployed to monitor conditions and report to the base. When a potential avalanche is detected, event messages again need reliable routing to the base to effect a rescue or some other affirmative action.

In summary, all these monitoring and event detection applications require that we deploy a dense network of sensors; dense so that the sensors provide adequate coverage of the environment in question. However the communications radius will typically be much larger than the sensing radius and this causes issues with congestion leading to serious packet loss.

Here we also see that a general telemetry environmental monitoring application is run in parallel with the event detection system. The former's data being periodically sent to whomever is interested (probably via a base station to a server) and the latter messages need much more reliable and speedy transfer to effect change. However, the behaviours of each application are different.

In routine sensing applications, each sensing node generates a relatively known traffic or packet rate as it is monitoring phenomena periodically. For simplicity, we illustrate this for a single hop wireless sensor network. Suppose, there are n nodes in the network with a single sink with an achievable capacity of C [2]. If we consider that all nodes can detect and send routine messages at same time to the sink, then the maximum achievable capacity assigned to each node is shared over the network and is C/n. However, when we introduce the event detection scenario, nodes are likely to detect events that occur less predictably. Furthermore, these events may originate from a specific region in the network. Let us assume 10 % of total nodes n generate five times the *normal* packets/messages per second. In this case the minimum required capacity for these event detecting nodes is increased five times compared to the previous case. Therefore, the new required capacity for each is $\frac{5*C}{n}$. This means that the total increased traffic in the network is 50% more than that observed in the basic example.

Two conditions result in network congestion for single-hop scenarios. Firstly, if the number of nodes detecting events are more than expected, this obviously leads to extra traffic and secondly the event nodes generate packets at higher rates than the predefined rate. Generally, the majority of sensing application solutions are designed to provide a guaranteed QoS for predictable traffic (routine traffic) but in the case of events, further unpredictable data is generated that can bring the message rates beyond the expected capacity of the network leading to congestion. Even when the total network traffic has not exceeded the network capacity we can experience congestion due to other reasons such as channel contention caused by concurrent transmissions, buffer overflows and varying wireless channel conditions. Therefore, there is a need to handle the dual behaviours of the applications; allowing the network to cope with sudden increased requirements without drastically reducing the ability of the day-to-day telemetry application.

Using this brief specification we gave ourselves a fortnight to come up with a solution and produce some results. We took on board our recommendations that we require a simple solution and that we cannot be too loose with our assumptions. The next section describes our simple approach.

3. VIRTUAL NETWORK OVERVIEW

Dense networks mean that the network has more nodes in a space and therefore, is required to handle many more messages increasing the probability of collisions (causing retransmits) and contention for the medium. Initially our approach was to look at de-synchronization schemes that ensured that the nodes were not transmitting at the same time, but then came up with a more novel and yet simpler scheme. Here we essentially dynamically create a number of virtual subnetworks. Network energy and application sensing resolution requirements determine which nodes belong to which virtual network (VN). Each VN self-configures to different wireless radio channels to avoid contention and interference; the aim being to dramatically increase the capacity of the network.

We now need to see how the components operate; this is in terms of the sink node, the sensing nodes and rate control mechanisms. The sink node is the one that receives all the data from the network. To do this it must be able to receive messages from all the VNs. An alternative is to have many sinks, each on different channels, with a meta-sink but this restricts the agility of the network to move between channels and is more costly as we then need more sink nodes. To make things more interesting let us assume a single static sink. The challenge here lies in the design of the sink's channel switching algorithm such that when the sink is tuned to a given channel we wish to minimise the packet loss that can result from missing a message being sent from a different channel. Therefore, the sink must monitor all networks while reflecting the priorities specified by the system. That is, if a given VN is used for priority packets then it gets more attention from the sink.

For a sensing node to join the network it must traverse down the channels and then chose one to join. A simple, yet communications hungry approach would be to have the sink dictate the topology and the sensing node would follow. However, this means that all admin traffic must be directed to and from a centralised node

[2]Maximum receive rate (pkts/sec) at the sink

regularly and such traffic requires the radio to be listening (draining batteries) which generally means that the network is not flexible to reconfigure to overcome failure. Therefore, a more agile solution is to self-configure the network in a decentralised fashion, such that the potential for packet collisions are as low as possible, and to do so in a lightweight way lowering the number of messages. We introduce two approaches. Firstly, the nodes periodically decide on which VN to join based on weighted probabilities. This can be based on the number of neighbour channels a node hears; the less a node hears of a given neighbour channel the higher the probability of choosing it, therefore, reducing collisions. Another scheme is to base the probabilities on the amount of energy available at each node. The intuition is that a new node in the network can be used to replace a dying node thus aiding the network upgrade/maintenance. Specifically, we allocate the VN with the least energy a higher probability of being chosen.

Finally, even with this, there will be congestion issues within a given network and this is especially important to the nodes closest to the sink for each of the VNs; as all nodes in the respective VNs reach the sink via them. Therefore, for each network we need mechanisms to prevent these nodes becoming a barrier to communicating to the sink. Our solution was to use rate restriction mechanisms.

4. BACKGROUND AND RELATED WORK

To make our evaluation more realistic we need to examine similar work researching tracking, channel switching and congestion control respectively. This allows us to populate our simulations with more realistic parameters.

4.1. Tracking using WSN

In Border Surveillance (Dudek et al. 2009), a PIR sensor was used to monitor a small border area for trespassers. To cope with node failure every node sends and listens for heartbeat messages. In order to minimise the power consumption the nodes duty cycle all behaviour except for the sensing; the PIR sensor remains active. However, this means that when a node wakes up it must syncronise with the networks so that they have a global idea of time etc. Similar behavior was found in (Mitra et al. 2009) and (Tennina et al. 2009). What is common between these systems is that the identification of the object to be tracked is processed on the nodes that are doing the sensing. This means that when an event is detected typically the size of the message that is being sent only consists of the type of alarm (usually an integer) and the identification of the node raising the alarm (again an integer). We assume this size for the messages in our evaluation.

4.2. Channel Switching

There has been a number of studies related to interference patterns for sensor devices. In (Y. Wu and Lin 2008), several experiments have been performed on MicaZ nodes using the CC2420 radio chip in order to measure the interference caused by 802.11 as well as 802.15.4 networks operating at different channels. This identified that adjacent channels impact the CC2420 radio reception to a great extent and that packet reception ratio with adjacent channel wireless interference drops to 50% when the transmission power is lower than -15dbm. They also show that links with an RSSI between -77dB and -87dB become unreliable with the presence of adjacent channel interference. Likewise, in (Boano et al. 2009), precise and repeatable patterns of interference are generated by using the carrier-only mode of the Chipcon CC2420 radio chip. Therefore, we believe that the use of multi-channel hopping to implement the virtual network shows great potential. We use the figures from (Y. Wu and Lin 2008) to populate our simulations to illustrate our solution. Specifically, we assume the delay consumed by the switching task, is that for the CC2420 radio chip (as used by the MicaZ and other nodes) and is set as the value measured by (So et al. 2006) to be 300μs. We also assume that only 8 out of the 16 channels are available to our schemes. This would be dependent on the environment in which our system was deployed.

4.3. Congestion Control

Wireless sensor networks were initially proposed for low data rates. The multi-hop nature of WSN communication that provided the agility to overcome radio issues and node failure, has effectively reduced the network capacity (Song and He 2007)(Gupta and P.R.Kumar 2000)(Moscibroda 2007). The main capacity constraints can be attributed to: limited bandwidth, the half duplex capacity of the radios, the interference and contention on wireless medium due to its shared and broadcast nature, and the topology of the network (Incel 2009). Therefore, maximum reliable data rates achieved in different size networks and efficient rate allocation and control schemes have been proposed (Bian et al. 2007) (Fan et al. 2008) (Lin and Wang 2009) (Sridharan and Krishnamachari 2009) to overcome these problems.

There are different congestion and contention avoidance algorithms studied especially for wireless sensor networks (Ee and Bajcsy 2004) (Wan et al. 2003) (Kumar et al. 2008). When Congestion occurs in a particular region in the network, the data traffic is dispersed from the congested region to the sink via non-congested areas of the network (Tan et al. 2006) (Wang et al. 2006) or load-partition schemes can be used (Maimour et al. 2008). When the whole network is congested traffic reduction techniques have also been used. Network topologies and traffic aware congestion detection and

control schemes have also been proposed in (Kang et al. July 2007) (He et al. 2008). Buffer management and priority queue scheduling mechanisms are also used to enhance data rates and mitigate the congestion effects.

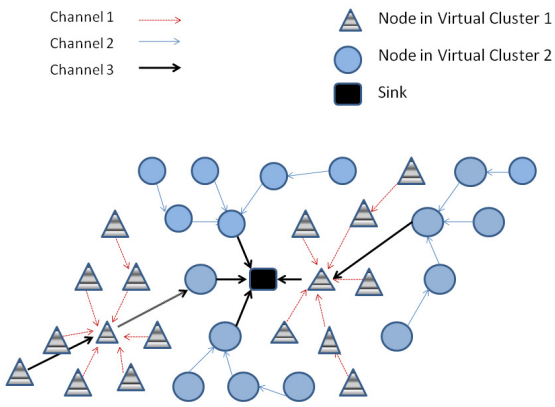

Figure 1: *Two virtual networks*

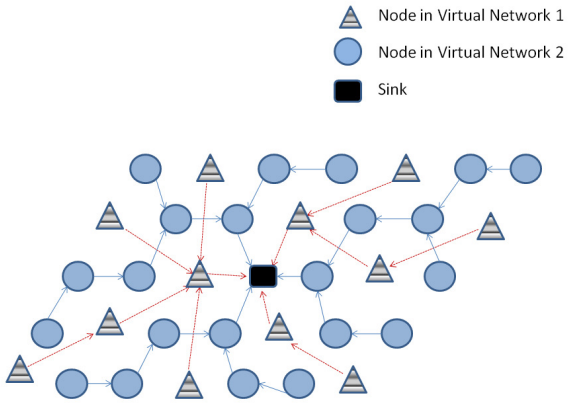

Figure 2: *Virtual clusters using two different channels*

Nevertheless, contention in wireless networks still plays major role in packet losses when data rates increase beyond the network capacity. Therefore, one way to overcome this is to utilise multi-channel radio control mechanisms. Others have began to explore this, (Gupta et al. 2006) propose a similar multi channel approach to ours where the communication channel is logically divided into sub-channels. However, this is on the basis of the locality of the data, see figure 1. Here channel assignment is per cluster of nodes which uses non-linear data-channel mapping. This approach assumes that the contention is reduced between the clusters, as each nearby cluster works on a different channel. However the contention within the cluster itself is not discussed. Our approach aims to evenly spread the channels across the network, further reducing contention as shown in figure 2.

5. ESTABLISHING THE VIRTUAL NETWORK

We conducted experiments to better understand how to best establish the different virtual networks. Recall that we assume a single sink and therefore, this must be able to receive data from all channels. Correspondingly, when a node joins the network, it must decide which channel to join. The following sections aim to evaluate our initial schemes to establish this.

5.1. Analysing Sink operation

The aim of designing the sink's channel switching algorithm is to make the packet loss probability as low as possible. In this section we analyze the sources of packet loss and give three simple but representative channel switching schemes to highlight the design principle.

5.1.1. Assumptions and definitions

Denote R as the data rate of each sensor node (packets per second). Denote $C = \{C_1, C_2, ...C_N\}$ as the set of reliable channels in the network. Denote n_i^j as the jth one-hop neighbour in channel C_i. The traffic of the each one-hop neighbour of the sink is assumed to be a Poisson Process[3]

$$P[N(t) = k] = \frac{(\lambda t)^k}{k!} e^{-\lambda t}$$

Denote λ_i as the total traffic intensity of channel C_i and λ_i^j as the traffic intensity of the jth one-hop neighbour in channel C_i.

Denote S as the time consumed in one channel switch operation and $f(CSS)$ as the average switching frequency, under a certain Channel Switching Scheme(CSS). Consequently, the average available bandwidth of the sink is $R[1 - f(CSS)S]$. We assume that the channel switching time of the CC2420 radio is S=300 μs (So et al. 2006). Note that during this time the transceiver cannot send or receive messages. If $f(CCS) = 1000$, then the available bandwidth of the sink is 175kbps(R=250bps), without considering wireless contention.

5.1.2. Packet loss due to uniform channel scheduler.

We consider a very simple channel scheduler scheme (Algorithm 1), which uniformly divides the sink's bandwidth to each channel.

In this case, the long term average number of packets lost per channels per second is

[3]If the data of each node is Poisson, the integration of these flows (each sensor node generates a flow) can therefore, also be considered as a Poisson process. In addition, if the flow generated by each sensor is independent, when the network becomes large enough, the integration of these flows can further be considered as Poisson processes regardless which stochastic process each flow has.

$$\sum_{\lambda_i > R(1-f(CSS)S)/|C|} \lambda_i - R(1-f(CSS)S)/|C|$$

Note algorithm 1 is naive because it does not consider the traffic distribution of each channel.

Algorithm 1 Uniform Switching

Require: T_P, t
1: **while** (1) **do**
2: time t elapses
3: **if** $t = T_P$ **then**
4: switch to next channel
5: $t \leftarrow 0$
6: **end if**
7: **end while**

5.1.3. Packet loss due to data buffer constraints
This section demonstrates that even if the traffic intensity of each channel is identical when using Algorithm 1 or there is a traffic-aware channel scheduler scheme (Algorithm 2), packet loss can still occur because the arrival rate of incoming data means that the nodes closest to the sink will run out of buffer storage as they are serving the data to the sink from all channels. Therefore, we examine an alterative approach thus.

Algorithm 2 Prediction-based channel scheduler with single one-hop neighbour in each channel

Require: RD_i, λ_i, T^i_{remain}, T^{min}_{remain}
1: scan all channels once
2: get init λ_i, RD_i, T^i_{remain}, T^{min}_{remain}
3: update λ_i, RD_i, T^i_{remain}, T^{min}_{remain} based on each received packet and current time
4: **if** $T^{min}_{remain} = S$ **then**
5: switch channel to $C_j(T^j_{remain} = T^{min}remain)$
6: **end if**
7: goto 3

The aim of this algorithm is to ensure the node nearest the sink with the fullest buffer is served by the sink first so it does not fill. We assume the sink retrieves this information from its nearest neighbours. Assuming the sink knows the intensity of incoming data (through the neighbours packet inter-arrival time) λ_i, and the remaining data buffer RD_i at time t_0. It starts the channel switching operation for channel C_i before T^i_{remain} thus

$$T^i_{remain} = RD_i/\lambda_i + S + t_0 \quad (5)$$

T^i_{remain} is the remaining time for a given node before its buffer is full. Denote T^{min}_{remain} chosen is the smallest T^i_{remain} of all channels. The neighbour node in channel C_i can estimate the parameter λ_i using a prediction method (e.g. EWMA filter (Cox 1961), AR model (M. and Shien-Ming 1983), etc). Take EWMA filter for example: Assume $\overline{D^k_i}$ is the number of packets received by the sink in channel C_i in time slot k, then λ_i can be estimated as

$$\lambda_i = \alpha\overline{D^{k-1}_i}/TS + (1-\alpha)\overline{D^k_i}/TS \quad (6)$$

Where TS is the time duration of a time slot and α is the weighting factor. The parameter RD_i and λ_i can be obtained by adding an integer value to the data packet header, or obtained directly by the sink by periodically sending a request beacon. We believe that the first scheme is better because the broadcast beacon adding more traffic could potentially cause more collisions (defeating the purpose of our work).

Recall that packet loss can be due to buffer sizes. Assume the duration of transmission of a packet follows an exponential distribution and denote T_n as the transmission time of the nth packet thus:

$$P(T_n \leq t) = \begin{cases} 1 - e^{-\mu t}(t \geq 0) \\ 0(t < 0) \end{cases}$$

Assume the data buffer size is B in terms of packets. Then the transmission between each neighbour and the sink can be considered as an $M/M/1/N$ queue model ($N = B$) as shown in figure 3.

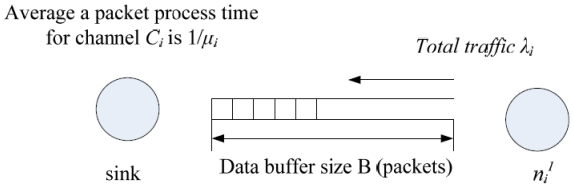

Figure 3: *The $M/M/1/N$ queue model ($N = B$) for each channel.*

In figure 3, the average transmission time $1/\mu_i$ of a packet in channel C_i is

$$\frac{1}{\mu_i} = \frac{1}{\eta_i R(1-f(CSS)S)}$$

Where η_i is the percentage of time that the sink allocates to channel C_i. Denote P_{loss} as the probability of packet loss and let $\rho_i = \lambda_i/\mu_i$. According to queuing theory (Kleinrock 1975), when $\rho_i < 1$, we get

$$P_{loss} = \frac{(1-\rho_i)\rho_i^B}{1-\rho_i^{B+1}}$$

Based on equations (3) and (4) we get figure 4 which shows an example of how data buffer sizes affect the packet loss probability B and consequently the channel switching frequency $f(CSS)$. Clearly, the probability of packet loss decreases as B increases. Obviously

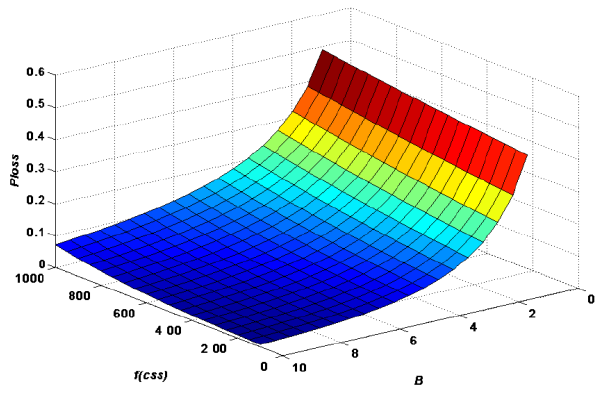

Figure 4: *probability of packet loss with different channel switching frequencies $f(CSS)$ when $\lambda_i = 1/3R$, $\eta_i = 1/2$, and S=300 μs.*

Algorithm 3 Prediction-based channel scheduler with more than one one-hop neighbour in each channel

Require: RD_i^j: The remaining buffer size of n_i^j,
λ_i^j: Intensity of incoming data of n_i^j,
T_{remain}^{ij}: Time before sink start switching to C_i,
T_{remain}^{min}: The smallest T_{remain}^{ij} in the set of all one-hop neighbours

1: scan all channels once
2: get init λ_i^j, RD_i^j, T_{remain}^{ij}, T_{remain}^{min}
3: update λ_i^j, RD_i^j, T_{remain}^{ij}, T_{remain}^{min} based on each received packet and current time
4: **if** $T_{remain}^{min} = S$ **then**
5: switch channel to $C_i(T_{remain}^{ij} = T_{remain}^{min})$
6: **end if**
7: goto 3

increasing buffer size decreases the probability of packet loss correspondingly. Interestingly, figure 4, shows that $f(CSS)$ does not have a significant influence on the probability of packet loss (P_{loss} as it increases almost linearly as $f(CSS)$ increases) - compared with the data buffer size B. This is because ρ_i is always less than 1 over the $f(CSS)$'s value interval [100, 1000] in figure 4. According to queueing analysis, even when $\rho_i = 1$, the probability of packet loss is equal to $1/(B+1)$. This result does not mean that the $f(CSS)$ can be enlarged without limitation. If ρ_i is larger than 1 caused by increased $f(CSS)$, the average service time of a data packet is larger than the incoming packet interval time, therefore, the P_{loss} will increase significantly.

5.1.4. Packet Loss due to multiple neighbours in one Channel

Algorithm 3 is proposed for the channel switching of the sink in the more general cases that there is more than one neighbour in each channel. The sink will switch to the channel in which there exists a neighbour with the least remaining time among all the sink's neighbours in all channels. The method of estimating the remaining time is similar as that in algorithm 2.

This approach can be combined with rate control mechanisms, see section 5.3. Here the rate allocation algorithm reduces the probability of of buffers becoming full at the nodes near the sink as they restrict the lower nodes in the tree from sending packets if that node cannot handle them; i.e. they explicitly reduce traffic.

5.2. Analysing Sensing Node operation

We now look at the sensing nodes building the VNs in a distributed way. The objectives of this first experiment was to gage the reduction in packet collisions that would come with VNs. Given that we assume a dense network, the number of neighbours an individual node can communicate with is large (the connectivity is high) hence the chance of collisions is high. Note as WSNs are heavily constrained by energy (we assume battery powered) and radio communication is the largest user of energy, it is of utmost importance to conserve energy by reducing these collisions.

Multiple radio channels then partition the network into (geographically) overlapping virtual networks. We decided to approach this, for now, by using probabilistic heuristics to determine which channel a node joins and this is inversely proportional to the number of neighbours heard in a given channel.

5.2.1. Experimental Assumptions

These schemes were simulated for a dense tracking wireless sensor network with 100 sensing nodes for the application as described in section 2.

We also assume that the nodes know the set of 'good' channels that could be used, although they do not know which channels are actually being used at this time; they discover that upon joining the network.

5.2.2. Bio-inspired Channel Choice Algorithms

These algorithms are all extensions to an event synchronization algorithm based on pulse coupled oscillators (Mirollo and Strogatz 1990). Recall when nodes sleep as part of the duty cycle to conserve energy they need to be synchronized. Previously we used a bio-inspired approach to this based on a combination of RFA and Gossip (Levis et al. 2004) (Allen et al. 2005). To keep the protocol both decentralised and minimize communication, the process to choose a channel is based on overhearing these periodic time synchronization messages. That is, during a given wake period in the duty cycle, a node listens for its synchronization messages from its neighbours and, for each channel, listens and counts the number of other nodes that are active for that channel. We assume all of the sensors work at the same frequency. In the following algorithms we use the synchronization messages to infer

populations of a given channel. We use this information to decide which channel is best for a new node to join.

When the algorithms begin, they start the synchronization procedure. After a short time (about 2 seconds) the node performs a channel scan. The channel scan consists of listening to each channel for half a second, and then switching to the next channel. The length of the scan window affects the algorithm as we trade off giving the node a chance to overhear all its neighbours per channel while getting as many channel switches done in an awake period. For this experiment we assume this to be half an event period (smaller windows showed more contention in earlier simulations). The choice of channel to begin the channel scan needs to be randomised, in order to prevent synchronization of all of the nodes to the same channel. After the initial channel is selected at random, the scan proceeds one channel at a time until all are exhausted. Channels where no neighbours are overheard are ignored in order to not partition the network. Our first idea is to use inverse weighted probabilities based on the number of neighbours per channel (Algorithm 4). In this way, the node has a higher chance of joining a channel with less nodes. The algorithm which calculates the weighting for the weighted channel choice process is $p_i = 100/(n_i \sum(1/n_i))$.

Algorithm 4 Pseudo code showing the weighted channel choice process

1: At random time scan all channels in network.
2: **loop**
3: Listen for number of neighbours in each channel.
4: **end loop**
5: Assign probabilities to each channel based on the number of neighbours heard in each channel.
6: $p_i = 100/(n_i \sum(1/n_i))$
7: Choose channel randomly based on the weighted probabilities.

We also show a second scheme (Algorithm 5) that examines how we could save energy. In this scheme the new node records a rolling average of energy levels of all of the nodes it overhears. When it makes its decision of which channel to join, it takes energy into account in the same way the weighted algorithm takes node population into account. See (Algorithm 5)

5.2.3. Methodology

These experiments were done by implementing the algorithm in the nesC language, and running it on the TOSSIM simulator for TinyOS-2.1.0. TOSSIM is tightly coupled with the Berkeley mote suite of hardware and TinyOS software and is used as a standard tool in the WSN field (amongst many) for quickly testing and evaluating without having to design and implement an actual testbed. All experiments were run 100 times.

Algorithm 5 Pseudo code showing the weighted channel choice process, taking energy into account.

1: At random time scan all channels in network.
2: **loop**
3: Listen for number of neighbours in each channel, and their energy levels.
4: **end loop**
5: Assign probabilities to each channel based on the number of neighbours heard in each channel, and the average energy level of the network.
6: $p_i = 100/n_i \sum 1/n_i$
7: Choose channel randomly based on the weighted probabilities.

Some adjustments to the simulator were necessary to facilitate the experiments. The version of TOSSIM used did not support changing channels in run-time easily therefore, we added channel switching as a simulator function call. Thus, we added a map of nodes to channel numbers as state in the simulator, and added two functions to set and get the channel (within the implementation of a node). Because the node ID is extracted from the context of the calls, nodes can only set their own channel.

Further, we then changed the simulation code to incorporate the channel of both the source and receiver when the packet is received. We adjust the received packet signal power level depending on whether the channel numbers match. If both nodes are in the same channel the power level calculations are that of the original TOSSIM implementation, otherwise we assume no communication and the packet is dropped mimicking the node ignoring another channel. This technique could easily be extended to add interference by modelling the RF filters in the receivers and changing the received power level to the one calculated based on the frequency interval between the channels involved.

We used a similar approach to model the battery life. There is a global state that maps every node to their battery level and this battery level is decremented every time a node has radio activity. When the battery of a node reaches zero, all messages sent are dropped at the simulator level, so the node is effectively dead for the purpose of our simulation results. This model is clearly unrealistic but it effectively models the fact that nodes will die at some time due to their battery draining, which is all that is assumed in our results.

The topology was generated by using the tool provided with TOSSIM, setting the loss at the reference distance to the signal power threshold for loss set in the TOSSIM transmission model (PL_{loss}). The size of the terrain (d_X,d_Y) and the reference distance (d_0) were then set such that the distance between the nodes was below a

minimal PIR sensor range of 8m thus:

$$PL(d = d_0) = PL_{loss} \quad (1)$$

$$d_0 = \frac{d_X}{3} / * 3\ hops * / \quad (2)$$

$$d_X = d_Y \quad (3)$$

5.2.4. Initial Results

The results in figure 5 show a sharp decrease in collisions detected as the number of channels increases. Great improvements are seen up to four channels. From there the improvements continue, but at a slower rate. The time to network death is shown in figure 6 which shows that the energy-aware algorithm shows a correspondingly longer average network lifetime as we increase the channels.

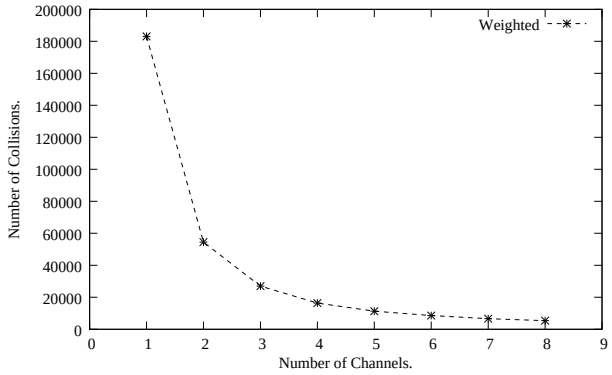

Figure 5: *Relationship between number of collisions detected by the nodes, and the number of channels used to partition the network.*

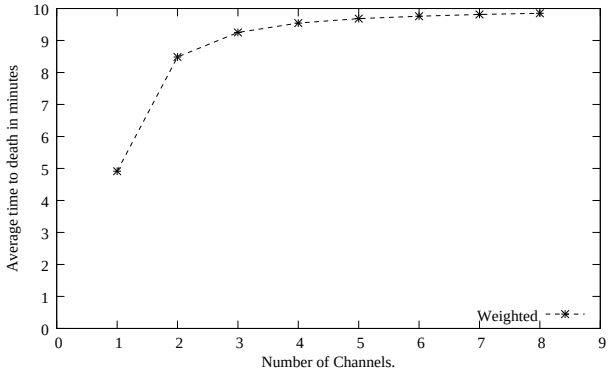

Figure 6: *Relationship between average time to death in minutes, and the number of channels used to partition the network.*

5.3. Congestion Control

The congestion control mechanism makes use of the capacity of each node to receive data. This is a product of both the ability of the communications being able to handle the rate of incoming data and the size of the data buffer on the node. To this end, we allocate to each node a total receiving capacity threshold and the nodes of the network that feed that node (i.e. the nodes lower in the tree) must not break that threshold. Initially, the receiving capacity of the sink is allocated dynamically to the different virtual networks depending on the traffic overload in each individual virtual network.

Algorithm 6 Rate Allocation and Control Schemes for individual Virtual Networks.

1: **loop**
2: Assign each Virtual Network the maximum available capacity, $C_i = C/Vn$; Here, 'C' is maximum capacity of the sink and 'Vn' are maximum number of virtual networks.
3: **end loop**
4: **loop**
5: For each virtual network, starting from sink: Allocate the capacity C_i to each connected children in the virtual network, and the maximum available capacity is further allocated to lower branches of the spanning tree.
6: **end loop**

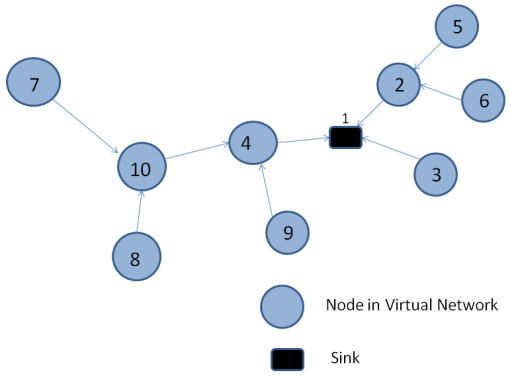

Figure 7: *Topology of Virtual Network considered in the experiment to show the efficient rate allocation schemes as a part of congestion control schemes*

5.3.1. Methodology and Assumptions

We simulated our algorithms for the MicaZ motes using a "closest-fit pattern matching" noise model and a SNR based interference packet error model with CSMA.

The subset example network simulated is illustrated in figure 6.

5.3.2. Network Congestion Control Evaluation

Algorithm 6 explains the initial maximum available capacity allocation for each virtual network as well as within the virtual network. In figure 8, we can see the data delivery ratio in two rate allocation schemes. First is a per-node based fair rate allocation scheme where all nodes in the network get an equal share of the sink's receiver's maximum capacity. In the second scheme

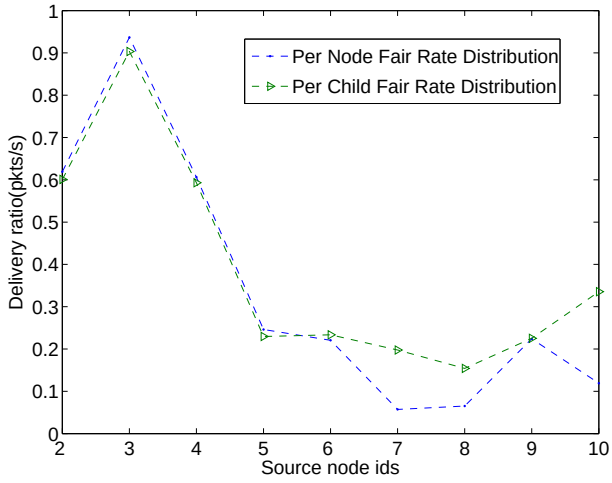

Figure 8: *Two rate allocation schemes in the one virtual network (Topology shown in figure 7)*

each node equally shares the available capacity of their parent nodes and this permeates down the network dividing by the number of children at each level. We call the latter scheme the per-child fair allocation scheme. In figure 8, we can see the comparative packet delivery ratios in two schemes for the virtual network that is shown in figure 7. Both schemes seem to behave in a similar way but we see that when node 7 is the event generating node (node 7 is farther away from the sink and the tree is not balanced here) we receive a better delivery ratio with the per-child scheme because the allocation is distributed for that particular branch and reflects the number of nodes and the node depth; providing better allocation for unbalanced trees. We use this allocation scheme in the subsequent experiments.

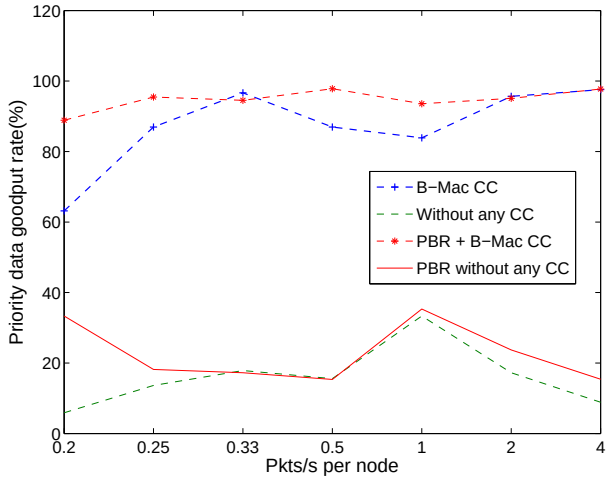

Figure 9: *Useful packet throughput(Goodput) rate at sink for event packets with Priority Based Routing*

We implement the rate control using our Priority Based Rate control (PBR) algorithm(Yadav and McCann 2009). Briefly, this method routes priority packets over the shortest path to the sink which diverts the non-priority data via broader slower routes (blocking off the fastest route to them). The per-child rate control mechanism described in Algorithm 7 is implemented in PBR.

We measure Goodput rates for increasing numbers of events generated per node. Firstly using the Congestion Control that the MAC layer provides (i.e. TinyOS Multihop-LQI which does not discriminate on packets) which is then compared with no congestion control mechanisms running. Also, the Priority Based Routing rate control mechanism is compared with the B-mac (Multihop-LQI) congestion control and this is compared with PBR with the MAC layer congestion control turned off.

It is very clear from figure 9 that an efficient Congestion Control scheme is very important for high delivery of data in dense networks whether or not we use our PBR scheme. Further, we notice that priority based routing improves Goodput by around 10% as compared to the Multihop- LQI Algorithm alone. For more detailed information on the PBR congestion control scheme see (Yadav and McCann 2009). This is a first stab experiment which could be improved with further optimisation.

6. RESULTS DISCUSSION

From section 5.1.2 to section 5.1.4, we have introduced three simple channel switching algorithms that are used by the sink, and analyzed their packet loss due to the channel switching process. Here we showed that the buffers of nodes close to the sink were important. We found that the time consumed for channel switching does not have a significant influence on packet loss if ρ_i is less than 1. In addition, given that algorithms 2 and 3 require a prediction method and control information from adjacent nodes to the sink, the performance of these significantly influenced by the ability of these nodes to communicate their control information. On the other hand, algorithm 1 does not require prediction and control information, however it does require that the traffic of each channel to be as uniform as possible.

Both the sink and sensing node experiments have scope for much more exploration. We could take into account comparisons with purely random choices or other schemes rather than the ranked probabilities we used. It would be interesting to see if we can observe a situation whereby weighted results were worse than the uniform ones. Note that when a channel is highly congested, the node will not hear all the packets. Intuitively, one can imagine a situation whereby all nodes join the most congested channel because they hear the least number of packets from it. This perhaps could be overcome by better understanding the presence of collisions in the channel. For example, the MAC layer might be able to communicate the number of collisions over a time window. Further, though none of

our experiments had a reachability issue, we have not proven that in all cases a VN can develop such that it will definitely reach the sink; something for further work.

7. SUMMARY AND CONCLUSION

This paper presents the argument that the WSN community must think out of the box, get real and keep it simple. With this ethos in mind, and a fortnight set aside, we believe we produced a novel but simple solution to the problems associated with dual purposed dense networks that uses a decentralised self-organising network that operates over different channels in the available radio band. The nodes in the network join a particular virtual network based on the energy levels and the channel information of its neighbours to reduce contention. Rate allocation mechanisms are deployed for each virtual network which is then dynamic, and data rates are adjusted to reflect the priority of the event traffic requiring transmission over that network to ensure timely and reliable delivery.

Is this a simple solution, though? We started out with a simple solution and then when we began to think about its nuances things got more tricky and complex. Further, as we had only a fortnight to whip up some results, we conformed to norm and used the standard Motelab, motes, TOSSIM tool kit - hardly thinking out of the box. We tried our best to keep it real, but not having a cave/mountain reserve to hand this was difficult. Let us return to some of the more robust examples of large scale WSNs that have been active in the wild for years. Two spring to mind, namely the Oklahoma City Micronet and the Victoria & Albert Museum Conservation system. What is common to both systems is the use of very simple hardware and software. What is also common with these examples, like that of the WWW, is that they were not designed by Computer Scientists. They were designed by meteorologists and conservational physicists respectively. Perhaps our Computer Science way of thinking which tries to cover all angles just complicates things. So to conclude, the WSN field is not dead by any means, we are at a turning point, and when we stop bit-twiddling perhaps we can show that it is a field that can really contribute to helping us preserve and protect the world around us.

8. REFERENCES

Allen, G., Tewari, G., Patel, A. and Nagpal, M. W. R. (2005), Fire-inspired sensor network synchronicity with realistic radio effects, *in* 'SenSys'05'.

Bian, F., Rangwala, S. and Govindan, R. (2007), Quasi-static centralized rate allocation for sensor networks, *in* 'SECON', pp. 361–370.

Boano, C. A., Römer, K., He, Z., Voigt, T., Zú niga, M. A. and Willig, A. (2009), Generation of controllable radio interference for protocol testing in wireless sensor networks, *in* 'SenSys '09: Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems', ACM, New York, NY, USA, pp. 301–302.

Cox, D. R. (1961), *Prediction by exponentially weighted moving averages and related methods*, Royal Statistical Society.

Dudek, D., Haas, C., Kuntz, A., Zitterbart, M., Krüger, D., Rothenpieler, P., Pfisterer, D. and Fischer, S. (2009), A wireless sensor network for border surveillance, *in* 'SenSys '09: Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems', ACM, New York, NY, USA, pp. 303–304.

Ee, C. T. and Bajcsy, R. (2004), Congestion control and fairness for many-to-one routing in sensor networks, *in* 'SenSys '04: Proceedings of the 2nd international conference on Embedded networked sensor systems', ACM, New York, NY, USA, pp. 148–161.

Fan, K.-W., Zheng, Z. and Sinha, P. (2008), Steady and fair rate allocation for rechargeable sensors in perpetual sensor networks, *in* 'SenSys '08: Proceedings of the 6th ACM conference on Embedded network sensor systems', ACM, New York, NY, USA, pp. 239–252.

Gupta, A., Gui, C. and Mohapatra, P. (2006), Exploiting multi-channel clustering for power efficiency in sensor networks, *in* 'Communication System Software and Middleware, 2006. Comsware 2006. First International Conference on', pp. 1–10.

Gupta, P. and P.R.Kumar (2000), 'The capacity of wireless networks', *IEEE Transactions on Information Theory* **46**(2), 388–404.

He, T., Ren, F., Lin, C. and Das, S. (2008), Alleviating congestion using traffic-aware dynamic routing in wireless sensor networks, *in* 'Sensor, Mesh and Ad Hoc Communications and Networks, 2008. SECON '08. 5th Annual IEEE Communications Society Conference on', pp. 233–241.

Incel, O. D. (2009), Multi-channel Wireless Sensor Networks: Protocols, Design and Evaluation, PhD thesis, University of Twente, The Netherlands.

Kahn, J. M., Katz, R. H. and Pister, K. S. J. (1999), Next century challenges: Mobile networking for smart dust, *in* 'The Fifth Annual International Conference on Mobile Computing and Networks (MobiCOM '99)'.

Kang, J., Zhang, Y. and Nath, B. (July 2007), 'TARA: Topology-aware resource adaptation to alleviate congestion in sensor networks', *Trans. on Parallel and Distributed Systems* **18**(7), 919–931.

Kleinrock, L. (1975), *Queueing Systems. Volume I: Theory*, Wiley-Interscience.

Kumar, R., Crepaldi, R., Rowaihy, H., Harris, A., Cao, G., Zorzi, M. and La Porta, T. (2008), Mitigating performance degradation in congested sensor networks, *in* 'IEEE Transactions on Mobile Computing', Vol. 7, pp. 682–697.

Levis, P., Patel, N., Culler, D. and Shenker, S. (2004), Trickle: a self-regulating algorithm for code propagation and maintenance in wireless sensor networks, *in* 'NSDI'04: Proceedings of the 1st conference on Symposium on Networked Systems Design and Implementation', USENIX Association, Berkeley, CA, USA, pp. 2–2.

Lin, C. L. and Wang, J. S. (2009), *Distributed Computing in Sensor Systems*, Springer, chapter Optimal Rate Allocation of Compressed Data Streams in Multihop Sensor Networks, pp. 58–71.

M., P. S. and Shien-Ming, W. (1983), *Time Series and System Analysis with Applications*, John Wiley and Sons, Inc.

Maimour, M., Pham, C. and Amelot, J. (2008), 'Load repartition for congestion control in multimedia wireless sensor networks with multipath routing', *CoRR* **abs/0810.1139**.

Mirollo, R. and Strogatz, S. (1990), 'Synchronization of pulse-coupled biological oscillators', *SIAM Journal on Applied Mathematics* **50**, 1645–1662.

Mitra, S., Zheng, Z., Guha, S., Ghosh, A., Dutta, P., Krishna, B., Plarre, K., Kumar, S. and Sinha, P. (2009), An affordable, long-lasting, and autonomous theft detection and tracking system, *in* 'Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems'.

Moscibroda, T. (2007), 'The worst-case capacity of wireless sensor networks', *IPSN'07: The 6th ACM International Symposium on Information Processing in Sensor Networks* pp. 1–10.

So, H. W., Nguyen, G. and Walrand, J. (2006), Practical synchronization techniques for multi-channel mac, *in* 'ACM MobiCom'.

Song, M. and He, B. (2007), 'Capacity analysis for flat and clustered wireless sensor networks', *Proceeding of the International Conference on Wireless Algorithms, Systems and Applications (WASA 2007)* pp. 249–253.

Sridharan, A. and Krishnamachari, B. (2009), Explicit and precise rate control for wireless sensor networks, *in* 'SenSys '09: Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems', ACM, New York, NY, USA, pp. 29–42.

Tan, W. L., Yue, O. and Lau, W. C. (2006), Performance evaluation of differentiated services mechanisms over wireless sensor networks, *in* 'VTC'06: Proceedings of 64th IEEE Vehicular Technology Conference', pp. 1–5.

Tennina, S., Pomante, L., Graziosi, F., Di Renzo, M., Alesii, R. and Santucci, F. (2009), Integrated gps-denied localization, tracking and automatic personal identification, *in* 'SenSys '09: Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems', ACM, New York, NY, USA, pp. 355–356.

Wan, C.-Y., Eisenman, S. B. and Campbell, A. T. (2003), CODA: congestion detection and avoidance in sensor networks, *in* 'Proc. of ACM SenSys', Los Angeles, California, U.S., pp. 266–279.

Wang, C., Sohraby, K., Lawrence, V. B., Li, B. and Hu, Y. (2006), Priority-based congestion control in wireless sensor networks, *in* 'SUTC (1)', pp. 22–31.

Wang, X., Zhao, X., Liang, Z. and Tan, M. (2007), Deploying a wireless sensor network on the coal mines, *in* 'Proceedings of the 2007 IEEE International Conference on Networking, Sensing and Control', pp. 324–328.

Y. Wu, J. Stankovic, T. H. and Lin, S. (2008), Realistic and efficient multi-channel communications in dense sensor networks., *in* 'INFOCOM '08, 2008.'.

Yadav, P. and McCann, J. A. (2009), Qos based event delivery for disaster monitoring applications, *in* 'Proceeding of the 5th International Conference on Wireless Communication and Sensor Networks'.