2008

SN-MAC: a cross layer MAC/Routing algorithm for high-throughput and low-latency WSNs

Mohamed Watfa
University of Wollongong, mwatfa@uow.edu.au

Samir Selman
American University of Beirut

Hovig Denkilkian
American University of Beirut

Publication Details

Watfa, M., Selman, S. & Denkilkian, H. 2008, 'SN-MAC: a cross layer MAC/Routing algorithm for high-throughput and low-latency WSNs', International Conference on wireless networks (ICWN'08),

SN-MAC: A Cross Layer MAC/Routing Algorithm for High-Throughput and Low-Latency WSNs

Mohamed Watfa[1], **Samir Selman**[2], **Hovig Denkilkian**[2]
[1]Computer Science, American University of Beirut, Beirut, Lebanon
[2]Electrical and Computer Engineering, American University of Beirut, Beirut, Lebanon

Abstract - *Energy savings have always been the primary concern in sensor network protocols; however there are applications where latency and throughput are prioritized over energy efficiency and are so significant that the application would not satisfy its requirements without them. Although existing duty-cycle MAC protocols such as S-MAC, T-MAC and Z-MAC are power efficient, they introduce significant end-to-end delivery latency and provide poor throughput. In this paper, we propose SN-MAC, a CDMA-based power controlled medium access protocol that uses both transmitter based and receiver based CDMA inside a formed cluster, and a TDMA schedule to make the cluster heads communicate with the base station. Our MAC/Routing algorithm targets latency and throughput needs in addition to its ability to increase the overall network lifetime. We discuss the design of SN-MAC, and provide a head-to-head comparison with S-MAC focusing on the performance in terms of latency, throughput and energy consumption.*

Keywords: MAC, Sensor Networks, CDMA, Latency, Throughput, Energy Efficiency**.**

1 Introduction

The communication unit and the antenna operation consume most of the battery powered energy of a sensor node. This means that the access to the medium must be controlled in a very strict manner in order to avoid collisions which result in lost transmissions and have a dramatic impact on the lifetime of the network**.** CDMA systems allow for concurrent transmissions at the same frequency to occur through separating the signals by their corresponding spreading codes**.**. Each terminal after joining the network receives a code through the code assigning protocol which it uses to expand the bandwidth of its signals that need to be transmitted, thus allowing for multiple transmissions from different users to occur at the same time and in the same frequency. These spreading codes can be transmitter based, receiver based, or a hybrid of both as described in [1].

One advantage of CDMA systems is that they allow users to send at any time without being confined by a certain allocated time slot or frequency channel. This leads to significant improvement in system performance in both latency and throughput measures. But these systems require sophisticated correlation filters that increase the complexity and the cost of the receiver node. In our paper, we solve the problem by having only specific designated nodes having this complex circuitry "Super Nodes" while the rest are relatively simple.

Previously proposed MAC protocols for wireless sensor networks aim at minimizing the energy consumption of the nodes and this is done at the expense of degraded throughput and latency performance. There are many applications in wireless sensor networks that have stringent latency and high throughput requirements such as medical monitoring, intruder detection and battlefield surveillance. In the last for example, the data gathered by the sensors need to be transmitted effectively and under no delay conditions since it contains timed information about movement of explosives and car bombs that will signal the soldiers to act upon detection of enemy presence. Thus our protocol came to balance the considerations of energy efficiency, latency, throughput, and fault-tolerance in sensor networks.

SN-MAC uses a combination of DS-CDMA and TDMA on the MAC layer and reduces channel interference by using a power control mechanism and a separate code for control packets. The network is divided into clusters (*formed initially after deployment*), where each node could be any hop away from the clusterhead, that are kept intact for the whole lifetime of the network because our goal behind clustering is to construct a logical hierarchy in the network rather than assuring that each node is part of a cluster and that the clusterhead role is dynamically rotated to distribute energy fairly among nodes. Our algorithm can run on top of previously proposed clustering algorithms; yet we develop our own simple clustering algorithm (*SCA*) to show that our protocol does not need complex clustering and works fine even if only the basic requirements are met. The algorithm targets the MAC layer and provides through a cross layer design an optimum routing strategy that gives a best effort design to deliver data from the sensors towards the base station. The information flow traverses several nodes within a cluster reaching the clusterhead which in turn delivers the data to the base station. The clusters are divided into levels where each node chooses its best neighbor which is one level away from it, based on considerations of the battery state of the node, and packet transmission information which are represented in the form of a priority function. Since robustness is one of the desired characteristics of sensor networks, our algorithm reacts favorably upon the addition or failure of nodes and which could severely affect the performance of the network.

The rest of the paper is organized as follows. In Section 2, we present the related work in this area. Section 3 describes in detail the SN-MAC protocol. Section 4 presents the simulation results. We conclude this paper in Section 5.

2 Related work

S-MAC [2] and T-MAC [3] were designed for tackling the idle listening problem, which is a dominant source of energy waste in sensor networks, through the adoption of periodic sleep and wakeup schedules. In [4], a hybrid MAC scheme called Z-MAC was developed which combines the strengths of both TDMA and CSMA like protocols. In [5], CDMA Sensor MAC "CSMAC" was developed to minimize latency in addition to reducing energy consumption through the use of a DSSS CDMA system combined with frequency division to reduce multiple-access interference. Along the same lines, [6] introduced a cross layer analysis for CDMA based wireless sensor networks, that examined analytically the Multi-access Interference (MAI) problem and shed light on the trade-off between interference and connectivity. In [7], a CDMA-based MAC protocol was designed for wireless ad hoc networks where an out-of-band RTS/CTS handshake was used to dynamically determine the transmission power of a node so that it will not result in collisions at neighboring receivers.

All of the above mentioned protocols did not take into consideration the battery behavior when minimizing the energy consumption in the network. The authors in [8] proposed a novel battery-aware MAC protocol which schedules transmissions of different nodes in a round-robin fashion, based on the battery state of the contending nodes. However, this protocol does not take into consideration the energy consumed due to idle listening and uses a simple round-robin scheduler that is ineffective since it does not adapt to the needs of the transmitting nodes. More recently, a novel protocol named TP-MAC was described in [13] that achieved synchronized low power listening with rapid fast path establishment by the propagation of short wake-up tones. The results of their paper show that TP-MAC can achieve very low duty cycles for the same target latency when compared with pure SCP-MAC [11]. On the other hand, L-MAC [14] is a contention-based MAC-Protocol that targets low-latency, energy-constrained applications. L-MAC assumes that the network is divided into levels where nodes execute an adaptive sleeping schedule allowing those with lower traffic to have longer time to sleep in order to save more energy. Furthermore, Level based scheduling was used in DMAC [12] which presents an adaptive duty-cycle protocol that is designed for data gathering trees in sensor networks. DMAC uses topology knowledge in order to stagger nodes' schedules according to their position and depth in the routing tree, so that packets flow continuously from source nodes to the sink, minimizing end-to-end delay significantly.

Our paper presents SN-MAC, a comprehensive framework on the MAC and Routing layers to be adopted by sensor nodes. SNMAC further provides functionalities that can ease the design of upper layer protocols, especially clustering. Also, since CDMA code assigning protocols is essential in all CDMA systems, SN-MAC is able to integrate any code assignment protocol to the presented algorithm. SN-MAC tries to make make neighboring nodes at the same level adopt different schedules and try to make nodes at level 'n' adopt the same schedule as their neighboring nodes on levels 'n-1' and 'n+1'. The intuition behind these goals is that we try to make nodes adopt the same schedule as their upper and lower level neighbors since they will be responsible for relaying the packets from the upper level neighbors and through the lower level ones towards the cluster-head. SN-MAC presents a battery aware CDMA-based MAC protocol that will serve a low latency and high throughput demanding application. SN-MAC also provides the upper layers a routing strategy through a proper cross layer design.

3 Design and analysis of SN-MAC

After initial deployment, each node will be in the set-up phase in which it will run a simple clustering algorithm that achieves leveling, neighbor discovery, choice of schedule and PN-code exchange in addition to the formation of the clusters**.** Notice that all these functions are done in one step and only at startup and hence the overhead incurred will last for the whole lifetime of the network. After finishing the setup phase, nodes will use CDMA as their basic MAC protocol to communicate with other nodes in a cluster. Our algorithm implements an adaptive TDMA schedule between cluster heads to allow them to communicate with the base station.

3.1 Code Assignment Protocol

Since our algorithm uses CDMA as the basic MAC protocol, a distributed code assignment protocol becomes a must. This code assignment protocol should offer spatial reuse and aim at assigning nodes with PN codes such as guaranteeing that no logically neighboring nodes use the same PN-code. Several code assignment protocols have been designed as in [10] and all tackle the above goal. In this paper, we assume that a code assignment protocol is present at a higher layer, yet SN-MAC design provides great opportunities of code reuse through its scheduling algorithm that tends to have neighboring equi-level nodes adopt different schedules. Thus, these nodes are now able to use the same spreading code without interfering with each other since their wakeup schedules are different.

3.2 Simple clustering algorithm & network formation (SCA)

Our algorithm can run on top of previously proposed clustering algorithms; yet we develop our own simple clustering algorithm (*SCA*) to show that our protocol does not need complex clustering and works fine even if only the basic requirements are met. We also use SCA to leverage the overhead inherently present in any clustering algorithm to perform neighbor discovery, leveling, schedule selection and exchange of PN-codes at the same time. This is run only at startup and hence this minimal overhead is incurred only once in the network's lifetime. We assume there are two kinds of nodes in our network. The first we shall call "*super node*" and is supposed to have higher capabilities than the rest of the nodes in that they are more energy abundant and have high communication ranges (i.e. can directly communicate to the base station). They also have relatively more complex circuitry in order to receive packets from their one hop neighbors that will be sent using transmitter based CDMA. The second type is the "*normal node*" and which constitute most of the nodes in the network.

In SCA, we aim at forming multi-hop clusters with super nodes as cluster-heads. After deployment, each normal node waits for a predefined period T which depends on the total number of nodes in the network 'N'. In this time T, super nodes are allowed to form clusters. Each super node forms an invitation packet and includes in it a cluster ID, a level field and a PN-code field. The level field in the packet is set to 0 whereas the levels initially stored in the nodes have a value of INFINITY until they get updated by the reception of an invitation packet. This invitation packet is first sent to the super node's one-hop neighbors with a power equal to a normal node's maximum power. The one hop neighbors in turn are supposed to store the cluster ID which is now their cluster, increment the level field in the packet and store it as their level to replace the INFINITY value. They will also include the PN-code they listen at in the PN-code field and rebroadcast the packet to their one hop neighbors..

Next the two-hop neighbors of the super node now receive the invitation packet from their lower level neighbors. Upon receiving the packet, a node looks at its level and if it is greater than the level in the packet plus one, it updates its level by setting it equal to the packet level plus one then joins the advertised cluster. Also, it stores the address of the lower level neighbor that sent the packet along with its PN-code. On the other hand if the node's level is equal to the level in the packet plus one, it only updates its table given that the advertised cluster ID in the received packet is the same as its current cluster ID. Hence, it forms a table of its lower level one-hop neighbors and their corresponding PN-codes. These nodes in turn will then replace the PN-code of the received packet by their own PN-code, increment the level field and rebroadcast the packet again after waiting for the sufficient time 'T2' to allow their lower level nodes to finish broadcasting.

A node which doesn't belong to any cluster yet will try to contact the base station. If it succeeds, it elects itself as a cluster-head and floods an invitation packet as before however this time indicating that it is not a super node and hence do not have the complex circuitry needed to receive messages using transmitter based CDMA. Its one hop neighbors will thus resort to a centralized TDMA schedule managed by the advertised cluster head. We suspect such a case to be rare if enough super nodes are present, yet we present this patch to our algorithm to solve such unpredictable cases and decrease the chance of having partitioned sections in the network. Fig. 1 is an illustration of the scenario under consideration.

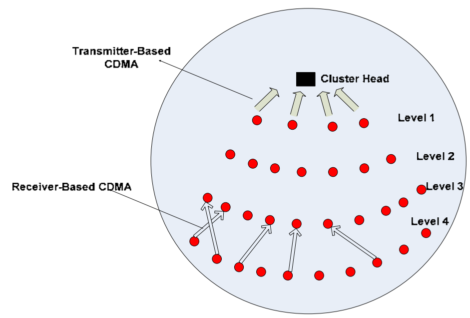

Fig 1. An illustration of the scenario within a cluster

3.3 Power control mechanism

In SN-MAC, we use a combination of transmitter based and receiver based CDMA. All level 1 nodes of a cluster (*i.e. nodes one-hop away from the cluster-head*) use transmitter based CDMA to send packets to the cluster-head whereas nodes with a level of 2 or above use a receiver based CDMA to communicate with upper level nodes. CDMA suffers from Multi Access Interference (MAI) and one of the ways of reducing the effect of MAI is through power control. Moreover, power control significantly reduces energy consumption which is highest during transmissions; however it also decreases the signal to interference plus noise ratio (SINR) at the targeted receiver.

This results in a tradeoff between reducing interference at non-targeted nodes and battery consumption on one hand, and reducing the SINR at the targeted node on the other hand. Since the applications we are considering (Intrusion detection, medical monitoring, animal tracking, etc.) possess the requirement of having all sensors that detect an event transmit their data with the lowest possible latency, there would be instances in the network lifetime where there is a burst of data that needs to be transmitted. The sensors would be carrying different types of information as well as different data within every type, and all need to reach the base station with minimum delay. Therefore, our power control mechanism will prioritize reducing MAI at neighboring nodes which we expect will be concurrently receiving transmissions as well. This will not totally prevent collisions from occurring; however, it will reduce their occurrence drastically and doesn't require a node to keep listening on any channel for the whole time.

In our scheme, a node *i* wishing to send to another node *j* will first send it an RTS on the control channel at maximum power $P_{\max}$. Node *j* will then calculate the minimum power that node *i* can use to send data to it. Hence node *j* would reply back with a CTS which includes the minimum power that *i* can use to send to *j*. Consequently, node *i* will send at the power which it received the CTS. We assume that all nodes initially agree and know the value of $P_{\max}$. Node *j* on the other hand receives the RTS with a certain power P_r and can thus compute the channel gain G $= \frac{P_r}{P_{\max}}$. We assume that the packet duration (Δt_p) is small compared to the coherence time of the channel (τ_c). Therefore, the channel is slowly fading and G can be assumed constant for the duration of the packet transmission. Let $\Omega_j = \frac{E_b}{N_{0eff}}$ be the effective bit energy-to-noise spectral density ratio at node j. Let Ω_j^* be the effective bit energy-to-noise spectral density ratio at node j that is needed to achieve the target bit error rate. Hence at node j we require that $\frac{P_{recv}}{P_{\text{Thermal}} + P_{\text{MAI}}} \geq \Omega_j^*$ where P_{recv} is the power received by terminal *j*, $P_{Thermal}$ is the thermal noise power and P_{MAI} is the total power due to multi-access interference at the receiver. Therefore, $P_{recv}^{\min} = \Omega_j^* \times (P_{Thermal} + P_{MAI})$, where $P_{recv}^{\min}$ is the minimum received power needed by terminal *j* in order to correctly decode the packet. Finally, $P_{\min} = \frac{P_{recv}^{\min}}{G}$ is the minimum power needed for the transmitter in order to allow the receiver to properly decode the packet. Node *j* places the minimum power calculated above as an additional field in the CTS packet and sends it back to node *i* for it to use in data transmissions.

3.4 Priority Assignment

A node needs to select which lower-level neighbor among the possible candidates it will choose as its intermediate node towards the cluster-head. The priority function aims at reducing the latency on the routing level and maximizing the network's lifetime through the proper choice of the next hop forwarders and doing load balancing. Three components determine the priority of a node to be chosen as the next hop forwarder:

1- *The Battery model:* Designing to reduce the average power consumption does not necessarily lead to optimum battery lifetime, since battery behavior highly depends on the discharge profile experienced by the battery [9]. Our algorithm forces the nodes to track the state of their batteries, assess their participation in the network, and react to changes in their battery states by changing their routing decisions. As a node's battery capacity increases, its priority will increase. This provides load balancing and prevents the formation of holes in the network. Also, as the time of the last transmission decreases, the computed priority will increase. Thus, we aim at choosing the neighbor whose last transmission was the farthest in time. This is to allow for nodes to recover and thus make efficient use of the capacity recovery effect in the battery model; hence also increasing the network lifetime.

$$\Pr_1 = \frac{BatteryCap}{Initial\ \ Battery\ \ Cap}\left(1 - e^{-\frac{CT\ -\ TOLT}{\tau}}\right) \quad (3)$$

Where *CT* = Current time, *TOLT* = Time of last transmission and τ = Average time to recover.

2- *The distance of the candidate node from the sender node:* As this distance decreases, the power needed to transmit a packet to that node will also decrease. This will preserve energy and also reduce MAI. Thus a neighboring node closer to the sender will be given a higher $\Pr_2$ value than another neighboring node farther yet still in its communication range.

$$\Pr_2 = \frac{d_{cand}}{d_{\max}} \quad (4)$$

Where d_{cand} is the distance of the candidate node whose priority is being computed to the sender node. d_{max} is the maximum distance between the sender node and any of the candidate nodes. d_{cand} can be estimated by a node from the power of the received CTS packet using the Free Space Pathloss Channel Model.

$$\frac{P_{recv\_CTS}}{P_{\max\_CTS}} = \frac{G_t G_r \lambda^2}{16\Pi^2 d_{cand}^2} \quad (5)$$

Where G_t, G_r are the antenna gains of the transmitter and receiver respectively. λ is the wavelength of the transmitted signal. P_{max_CTS} is the power by which the CTS packet was transmitted. This power is the maximum power that the nodes initially agree on as described earlier. Finally P_{recv_CTS} is the power of the CTS packet when it was received.

3- *Estimated congestion at the candidate*: Since congestion maps directly to latency, nodes tend to pick the neighbor who is least likely to be congested. The sender node can estimate the amount of congestion to each candidate node through a weighted average of the proportion of failed attempts to forward a packet to this designated candidate node i.e. a weighted average of the ratio of RTS packets which did not result in CTS replies to the total attempts of sending RTS packets. Then the corresponding component of the priority function would be:

$$\Pr_3 = \sum_{i=1}^{m} \frac{n_{fail\_RTS\_i}}{n_{attempts\_RTS\_i} \times i} \qquad (6)$$

Where *'i'* designates the current frame number. Hence as the frame number increases, the information being used for estimation is older and thus is given a lower weight. $n_{fail\_RTS\_i}$ is the number of RTS packets sent and did not result in a CTS reply during frame *i*. $n_{attempts\_RTS\_i}$ is the total number of attempts to send RTS packets during frame *i*.

Finally, the total priority is:

$$\Pr = \frac{\Pr_1}{\Pr_2 \times \Pr_3} \qquad (7)$$

Each node keeps a table in its memory containing its lower level neighboring nodes and their corresponding priorities. Since these priorities depend on the current time, they should be continuously calculated.

3.5 SN-MAC

The topology is now divided into clusters. Each cluster contains *n* levels of nodes. The level of a node within a cluster is defined by the number of hops the node lies away from the cluster head. Each node has PN-Codes which it can use to spread any signal it needs to send. The goal of the nodes within a cluster is to sense and forward data towards the base station through cluster heads and intermediate nodes within the cluster. The first time a node has data to send, will broadcast an RTS on the control channel. All of its awake lower-level neighbors will wait for a random time then reply with a CTS. This random waiting is aimed to avoid packet collisions. After receiving the CTS packets, the node builds up a table of its lower level neighbors. Notice that the table has in addition to the priority field, a NAV field which indicates the duration for which the node in the corresponding entry will remain busy. This NAV field is updated after overhearing an RTS or a CTS of the corresponding node and hence deducing that it will be busy for the time indicated by the duration field in theses packets. This table will be regularly updated with every overheard RTS and CTS. Since SN-MAC aims at giving nodes the same schedule as their lower level intermediate neighbors, we can expect that most of the RTS and CTS packets sent by nodes will be overheard by their higher level neighbors who will then be able to update the corresponding priorities and NAV fields. Next, we give the steps taken by the nodes to successfully route data all the way towards the base station:

1- When a node "n_i" at *level n* has data to send, it must first choose an intermediate lower level neighbor to send its packet to and which in turn will further forward it to lower level nodes. The first choice would be the non-busy neighbor with highest priority (*Suppose that node is node B*).

2- n_i will send an RTS to node B at maximum power. Node B in turn will reply with a CTS packet except in the following cases: - i) Node B is busy (*In this case n_i have missed overhearing the RTS or CTS that should have told it that node B will be involved in a transmission so that n_i won't send the RTS in the first place*); ii) Node B is asleep; iii) Node B is both awake and non-busy however when it received the RTS and computed the minimum power at which n_i needs to send at (*so that the packet can be successfully recovered*), this power came out to be larger than the maximum power a node can use to send due to the very large MAI around the receiver. Therefore B will refrain from replying with a CTS packet.

3- If node B doesn't reply for the above reasons, n_i will choose its non-busy neighbor with second highest priority as its intermediate neighbor and will hence send it an RTS. On the other hand, it will update its table after node B didn't reply by changing the priority of node B to a value equal to the minimum priority in the table i.e. n_i will place node B at the bottom of its table. This is to avoid high energy neighbors from being constantly requested and hence stay busy all the time.

4- If for the second time no CTS was received then n_i will broadcast an RTS. Several CTS packets will be received and the one with highest priority is to be chosen. The broadcasting after two failures is done because of the latency requirement imposed by the application running our algorithm.

5- If even after broadcasting the RTS, still no CTS packets arrive, then this means that all lower level neighbors of n_i are currently unavailable. n_i will then try to forward its packet through a neighbor with the same level by sending a *help* message. There are three cases in which a node might ask for help from another node on its level: i) the node after broadcasting an RTS still did not get a reply; ii) all the node's neighbors have a NAV> 0; iii) if there are only two or less available neighbors and have not replied on the RTS unicast. The *help* message is a modified RTS with a help bit set to 1. The receiver of a help message checks the corresponding level against the node's level. If they match the node will reply with a help-ack message (*modified CTS*) in case it had lower level non-busy neighbors which it can route through. Further failure to receive replies will cause the node to delay its transmission to a later time.

6- Upon receiving a CTS, a level n node (n≠1) switches its transceiver to the data channel, spreads the packet it wishes to send using the PN code of the desired receiver and then transmits it using a receiver based CDMA. The receiver in turn uses the same PN code to de-spread the packet sent.

7- The packet will continue to be forwarded upstream (to lower levels) using a receiver based CDMA until it reaches a node with level one. Level one nodes transmit to the cluster head using transmitter based CDMA.

8- The packet has successfully reached the cluster-head. Cluster heads in turn communicate to the base station using a centralized TDMA schedule. This is done to provide load balancing in addition to fairness between all regions of a network.

4 Simulation results

For comparative analysis, we simulated SN-MAC and SMAC using ns2. The number of the nodes was 200 and 16 clusters were formed as a result. Each point in any curve is the average of five different simulation runs. We use the following simulation parameters: Packet rate:4 packets/sec; Initial Energy=10J ; Transmission Power= 2 mW ; Receiver Power=1 mW; Sleep Power= 0.001 mW.

Fig. 3 shows the results of our *latency* evaluation for scenarios using SN-MAC and S-MAC. Delivery latency in both S-MAC and SN-MAC increases as the hop count of the path increases. However, delivery latency in S-MAC increases at a much faster rate showing the benefit of SN-MAC's capability of multi-hop delivery within a single cycle and multiple transmissions in the vicinity of the receiver are possible using transmitter based-CDMA. We also simulated the *latency* as a function of the packet arrival rate of both SN-MAC and S-MAC shown in Figure 4. The difference between the latency incurred by SMAC and our protocol is much higher as the packet arrival rate increases and this is due to the use of DSSS for the transmission of data packets adopted by our algorithm.

However, the significance of our algorithm is evident in the *overall network lifetime* simulation as depicted in Fig. 5. Our online battery model embedded with SN-MAC results in spreading the energy load on the whole network which results in increasing the overall lifetime of the network. The simulation was run for 1000 seconds and random events are generated at a rate of 5 events/second. Initially all the nodes were alive, but after 440 seconds, less than 90% of the nodes are alive using S-MAC however using SN-MAC it takes the network about 750 seconds for the number of active nodes to go below 90% and thus highlighting the effect of SN-MAC on the network lifetime extension.

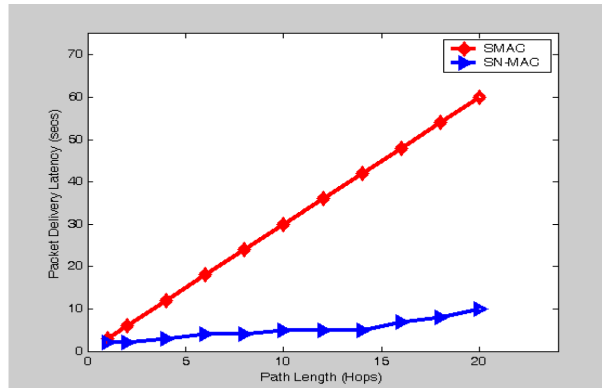

Fig 3. The delivery latency as the path length increases.

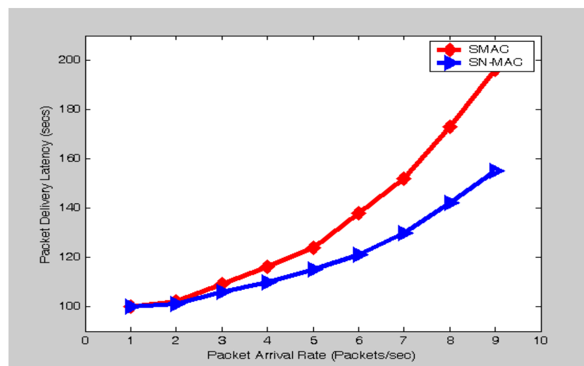

Fig 4. The delivery latency as the arrival rate increases.

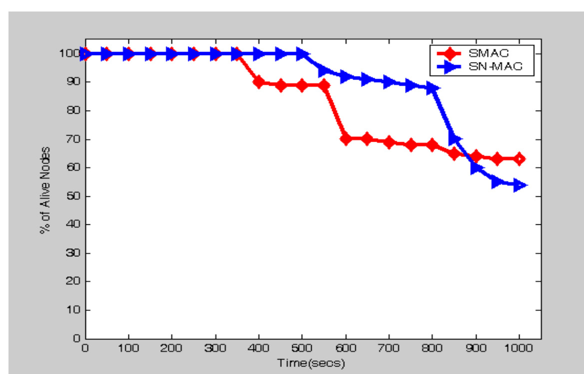

Fig 5. The network lifetime as time varies.

In Fig. 6, we evaluated the *network throughput* using both SN-MAC and SMAC. For both protocols, the output rate follows the input rate when the input rate is low and finally the output rate reaches its peak point. Using SMAC, if we continue injecting more packets into the system, after the output has peaked, the input creates more contention in the system and decreases the throughput slowly until the throughput reaches a steady state value. Although the medium is saturated when the load is high, SN-MAC packets can still be forwarded whenever it is possible and thus uses its medium access opportunity more efficiently than with RTS/CTS in S-MAC. That is because of SN-MAC's capability of multi-hop delivery within a single cycle and multiple transmissions in the vicinity of the receiver are possible using transmitter based-CDMA.

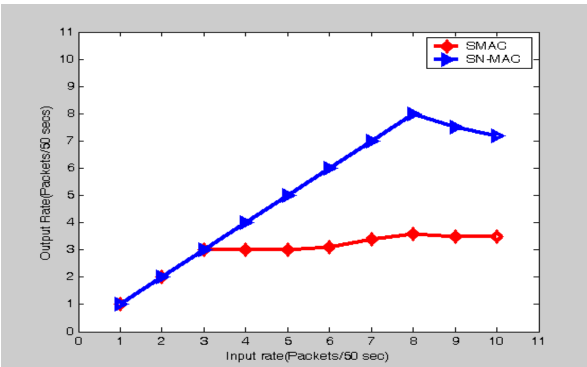

Fig 6. The analysis of the network throughput as we vary the input rate.

5 Conclusions

Sensor networks have always given energy efficiency much more importance than other requirements like latency and throughput. In this paper, we have achieved a low latency delivery of data from sensing nodes towards the base station taking into consideration sources of energy wastage and successfully minimizing them. SN-MAC decreases the delivery latency, increases the throughput while extending the overall network lifetime. We believe that simulating accurately the DSS could result in better energy efficient results since our protocol gains energy efficiency by adopting a sleep/wakeup schedule, using the battery capacity of the nodes and minimizing the number of data collisions through CDMA with a separate data channel.

6 References

[1] E. Sousa and J.A. Silvester."*Spreading code protocols for distributed spread-spectrum packet radio networks*". IEEE Trans. on Communications, vol 36, 98, pp. 272–281.

[2] W. Ye, J. Heidemann and D. Estrin, "*An Energy Efficient MAC Protocol for Wireless Sensor Networks*," INFOCOM 2002, USA, pp. 1567-1576.

[3] T. van Dam and K. Langendoen, *"An Adaptive Energy-Efficient Mac Protocol for Wireless Sensor Networks*," International conference on Embedded networked sensor systems**,** LA , 2003, pp. 171 – 180.

[4] Injong Rhee ,Ajit Warrier ,Mahesh Aia ,Jeongki Min, "*Z-MAC: a hybrid MAC for wireless sensor networks*,"In Proceedings of the international conference on Embedded networked sensor systems, San Diego, 2005, pp. 90 - 101.

[5] B. Liu, N. Bulusu, H. Pham, and S. Jha, "*CSMAC: A Novel DS-CDMA Based MAC Protocol for Wireless Sensor Networks*," Global Telecommunications Conference Workshops, Texas, 2004, pp. 33 – 38.

[6] S. De, Qiao, Pados, S.Philip, "*An Integrated Cross-Layer Study of Wireless CDMA Sensor Networks*", IEEE Journal, vol. 22, pp. 1271 – 1285, 2004.

[7] Alaa Muqattash and Marwan Krunz, "*CDMA-based MAC protocol for wireless ad hoc networks*", In 4th ACM international symposium on Mobile ad hoc networking & computing, 2003, pp. 153 – 164.

[8] S. Jayashree, B. Manoj, and C. Murthy, "*On Using Battery State for Medium Access Control in Ad Hoc Wireless Networks***,**" international conference on Mobile computing, Philadelphia, 2004, pp. 360-373.

[9] D. Rakhmatov and S. Vrudhula," *An analytical high-level battery model for use in energy management of portable electronic systems*", International Conference on Computer Aided Design, Atlanta, 2001, pp. 488-493.

[10] A. Bertossi and M. Bonuccelli." *Code assignment for hidden terminal interference avoidance in multihop packet radio networks",* IEEE Transactions on Communications, vol. 3, 1995, pp. 441–449.

[11] W. Ye, J. Heidemann, "Ultra-Low Duty Cycle MAC with Scheduled Channel Polling", Technical Report ISI-TR-2005-604, USC/Information Sciences Institute, July 2005.

[12] G. Lu, B. Krishnamachari, C. S. Raghavendra, "*An Adaptive Energy-Efficient and Low-Latency MAC for Data Gathering in Wireless Sensor Networks*", in Proceedings of the 18th International Parallel and Distributed Processing Symposium.

[13] A. Grilo, M. Macedo, M. S. Nunes, "*Tone-Propagated MAC (TP-MAC): A Low Duty-cycle Low Latency MAC Protocol for Wireless Sensor Networks*", Performance Control in Wireless Sensor Networks Workshop, held jointly with the IFIP Networking 2006 conference, Coimbra, Portugal, May 2006.

[14] S. Chatterjea, L. van Hoesel, and P. Havinga. *"AI-LMAC: An Adaptive, Information-centric and Lightweight MAC Protocol for Wireless Sensor Networks".* In Proceedings of the DEST International Workshop on Signal Processing for Sensor Networks. IEEE Computer Society Press, 2004.