Deakin Research Online

Address Reuse in Wireless Sensor Networks

R. Chellappa Doss, D. Chandra, L. Pan, W.Zhou, M. Chowdhury
School of Engineering and Information Technology
Deakin University, 221 Burwood Hwy, Victoria 3125, Australia.
{rchell, dchandra, ln, wanlei, muc}@deakin.edu.au

Abstract—**Sensor Networks have applications in diverse fields. While unique addressing is not a requirement of many data collecting applications of wireless sensor networks, it is vital for the success of applications such as emergency response. Data that cannot be associated with a specific node becomes useless in such situations. In this work we propose a dynamic addressing mechanism for wireless sensor networks. The scheme enables successful reuse of addresses in event-driven wireless sensor networks. It also eliminates the need for network-wide Duplicate Address Detection (DAD) to ensure uniqueness of network level addresses.**

Keywords – Wireless Sensors, System Design.

I. INTRODUCTION

Wireless sensor networks (WSN) represent the next step in the evolution of wireless communication. They are self-organizing networks that do not depend on a fixed communication infrastructure [1].

An application that is becoming increasingly attractive is the use of WSNs for emergency (first) response in mass casualty incidents. It is envisioned that these networks will play a pivotal role in disaster response and recovery [1]. WSN applications can be categorized as either data-centric or node-centric applications. While both categories of applications are concerned with data monitoring/collecting, data-centric applications do not require the node of data generation to be uniquely identified. Unlike in data-centric applications, node-centric applications require the identity of the node of data generation. In such applications data that is collected becomes useless if the source (sensor node) cannot be uniquely identified. An example is vital-sign monitoring applications such as in emergency response [1] [2].

Two levels of information can be identified in sensor nodes – events and data. Events are defined as critical data that is generated by a node [5] (e.g., a patient's vital sign measurement falls below a critical threshold or enemy movement has been detected). In event-driven sensor networks only events are of interest and need to be communicated to the sink.

In this work, we propose an On-demand Location Aided Addressing mechanism that can enable address reuse by exploiting the random nature of event occurrence in large scale WSNs. In section II we present related work followed by our proposed scheme in Section III. Analysis of the scheme is presented in Section IV. We conclude with final comments in Section V.

II. RELATED WORK

The need for efficient addressing schemes is well articulated with various addressing schemes proposed in literature [3] [5] [6] [7] [8] [9]. In [3] an energy-efficient node addressing scheme using spatial reuse of locally unique addresses is presented. Nodes are organized in a hierarchy of logical layers and used to satisfy the uniqueness condition. TreeCast [5] is a stateless addressing scheme proposed for efficient addressing. It requires the construction of multiple disjoint trees. A similar scheme [8] is based on the concept of hierarchical levels and repeated patterns and supports self-organization in sensor networks. In [4] a distributed on-demand addressing mechanism is proposed for assignment of MAC addresses. It exploits spatial reuse of addresses and uses Huffman coding to reduce the address length in the packer header. Event-driven addressing has been proposed in [7]. Local uniqueness between immediate neighbors is aimed for with link level addressing while an on-demand mechanism for network level addressing is proposed. The addressing protocol is coupled with the routing protocol and employs Duplicate Address Detection (DAD). In [6], data aggregation and dilution by modulus addressing is proposed while an addressing mechanism based on a hierarchical architecture using de Bruijn graphs is proposed in [9]. All of the above addressing schemes place the complexity of the addressing process on the sensor nodes by insisting on strict organization or by DAD through flooding. In our work the complexity is removed from the sensor nodes to the network control centre (sink).

III. DYNAMIC ADDRESSING FOR WSNS

The proposed addressing mechanism is an on-demand addressing protocol that employs a lease-based approach for address assignment. It exploits the random nature of event occurrence in event-driven sensor networks. Since events occur at random, addresses can be assigned and released in a dynamic manner enabling the reuse of addresses. In large scale sensor networks such an approach will reduce the overhead of addressing quite significantly. Location awareness is a requirement of many WSN applications [6]. The proposed mechanism incorporates location awareness and works with both absolute and relative levels of awareness. The proposed

addressing scheme has five main phases of operation. We shall discuss the operations of each of these phases in detail below.

A. Boot Up Phase

The dynamic addressing mechanism is used only for the assignment of network level addresses. During the boot up phase each node self-assigns a link-level address that is locally unique. The assignment of the link-level address proceeds along similar lines as described in [7]. The negotiated link-level address is assigned permanently to a node and is only reassigned in the event of the original node dying or reconfiguration when new nodes join the network. During the boot up phase the Sink (S) broadcasts a configuration packet that contains the location of the sink (x, y co-ordinates). The purpose of this configuration packet is to allow each of the sensor nodes to calculate their distances from the sink. A sensor node is deemed to have successfully joined the network (booted) only after the reception of this configuration packet. The distance of the sensor node from the sink, $d_{i,SINK}$ is calculated according to (1). Since the sensor nodes are relatively stationary, recalculation of the distance is not required after boot up. Each sensor node is assumed to be connected to a location device such as a GPS receiver. It is to be noted that the use of GPS measurements is only needed during the boot up phase and hence the overhead of location awareness will not be significant (i.e., once $d_{i,SINK}$ has been calculated the GPS receiver is turned off to conserve energy). Alternate boot up procedures based on parent nodes and tree-based routing can also be used.

$$d_{i,SINK} = \sqrt{(x_{SINK} - x_i)^2 - (y_{SINK} - y_i)^2} \qquad (1)$$

Since the topology of wireless sensor networks is dynamic, it is possible that new nodes will join the network either to replace nodes that have failed (died) or to expand the network. To complete self-configuration a new sensor node joining the network sends a *join* request that is received by all nodes within its transmission range. The one-hop neighbours respond to the *join* request with a configuration packet (they have previously received from the sink) that contains the location information of the sink. On the reception of this configuration packet the bootup phase is completed. The purpose of the location information is to enable energy-efficient forwarding of messages.

B. Address Request Phase

Once a node has successfully completed the boot up phase it becomes a candidate for address request. A node performs its data monitoring function with limited levels of local processing to generate an event. In order for the event to be communicated reliably to the sink a network level address is required to identify the source of the event at the sink (link-level addresses are only locally unique). The node generates an *address_request* packet of the form {*Source, Type, event_ID, moteDist_i, moteDist_Source*}. *Source* is the link level address of the requesting node, *Type* denotes the type of packet and

doubles as an identifier for the destination of the address request packet (since all address requests are destined for the sink), *event_ID* is an identifier for the specific event generating the request and is used to map an address allocation to the corresponding *address_request*, *moteDist_Source* is the distance of the requesting sensor node from the sink calculated using (1) and *moteDist_i* is the distance of the forwarding node from the sink. *moteDist_i* is used by forwarding nodes in their decision making process (i.e., to decide if a node is closer to the sink than the forwarding node to it). The distance of the requesting sensor is included in the address request packet as the Sink is not aware of the location of the deployed sensors. The *event_ID* carried in the address request packet is generated using a random function that takes the link-level address of the sensor node as an input to uniquely identify the address request and to match the corresponding reply (address allocation) from the sink to the original address request. This is needed as the link-level addresses are only locally unique.

Forwarding of the address request from the sensor node to the sink is done making use of limited-scope flooding. This is achieved making use of the distance rule. We define the distance rule to be – a node *j* forwards an address request from node *i* only if it is closer to the sink than node *i* (i.e., the forwarding node to it). The distance rule has been previously used in location aided routing protocols [10] for mobile ad hoc networks and is shown to be an effective mechanism. The distance rule effectively creates a multicast group towards the sink and hence is different from greedy forwarding (that can suffer from local maxima). A forwarding node will change the value of *moteDist_i* in the address request packet to its own *moteDist_j* to enable its neighbours to apply the distance rule. The value of *moteDist_Source* remains unchanged.

C. Address Allocation Phase

The sink maintains an *address_allocation* table with a list of addresses and a corresponding status flag for each address. On the receipt of an *address_request* the sink allocates a free address to the requesting node based on availability or alternatively discards the request. The sink also has the option to queue the request until an address becomes available or for a pre-defined time interval. In our analysis and simulations for the sake of tractability address requests are not queued.

On the receipt of the *address_request* packet the sink responds with an *address_allocation* packet of the form {*Source Type, event_ID, moteDist_Source, Address, moteDist_i*}. *Source* denotes the link level address of the destination sensor, *Type* denotes the packet type and also doubles to identify the sender as the Sink, *event_ID* is copied to the *address_allocation* packet from the corresponding *address_request* packet and is used to match the *address_allocation* message to the requesting sensor, *moteDist_Source* denotes the distance of the requesting sensor from the sink and is copied from the

address request packet. *Address* denotes the allocated address; $moteDist_i$ denotes the distance of the node that forwarded the original *address_request* message to the sink. Limited scope flooding using the distance rule is again employed to forward the reply to the requesting node. However, some modification is required to make it efficient. At the first instance the address allocation message is unicast using $moteDist_i$ and the link level address to the node that forwarded the address request message. On receiving this message the node then employs the distance rule with respect to the destination sensor. All calculations are with respect to the destination sensor and are done using the value of $moteDist_{Source}$. This requires that the location information ($moteDist_{Source}$) of the final destination sensor is included in the reply message generated by the sink. On sending of the *address_allocation* the sink stores the location information of the requesting sensor node and the *event_ID* and associates this pair with a network level address.
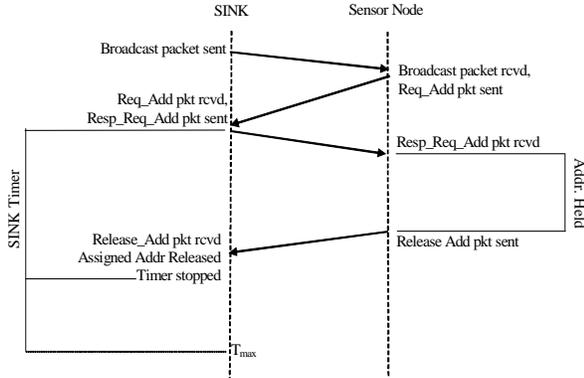

Figure 1. A Typical Address Request, Allocation and Release

When the sink sends out an address allocation response it starts a timer (*lease_timer*) that is associated with the allocated address. The value of this timer is set to T_{max} which is the longest amount of time that a node is allowed to hold on to an address. We refer to this time as a lease. T_{max} can be a fixed value. However there is the scope of extending this to differentiate between different classes of events. On the expiry of the lease *i.e., lease_timer* the address becomes available for allocation to a new node requesting an address. Along similar lines priority queuing can also be used to service address requests that have been queued

D. Address Release Phase

To enable reuse of addresses and optimization of the size of the address space, addresses are not allocated on a permanent basis to each sensor node. Instead, we adopt a lease based approach. Each address is held by a sensor for a period of time (less than T_{max}) until it completes communication related to an event. When communication is completed the sensor node explicitly releases the address enabling the sink to reuse the address (prior to expiry of T_{max}) for another node. The success of lease based schemes depends on the

effectiveness of the lease management mechanism. We adopt a distributed lease management mechanism for our scheme by combining the sink based *lease_timer* with a node-based address release mechanism. The *lease_timer* specifies the maximum time that a node can hold on to an address while the node based address release mechanism enables release of an address prior to the expiry of the *lease_timer*. It can be argued that a purely sink-based scheme is preferable to reduce the computational load on individual sensors. However, in event-driven networks a distributed approach is more beneficial as it enables efficient reuse of addresses.

When a node decides to release an address it sends an *address_release* message of the form {*Source, Type, moteDist_i, Release_Address*} to the sink. *Source* indicates the link level address of the node, *Type* identifies the packet as being destined to the sink, $moteDist_i$ is the distance of the forwarding node from the sink and *Release_Address* is the network level address being released. The sink on receipt of an *address_release* message deallocates the specific address which allows for the address to be reused by other nodes in the network. The release of the address is controlled entirely by the sensor nodes while the address allocation is controlled entirely by the sink nodes. The advantage of centralised address allocation controlled at the sink is that DAD can now be done at a central point. In comparison, other schemes perform DAD by flooding the entire network to see if a duplicate address exists across the network. Figure 1 illustrates a typical address request, allocation and release with no packet loss.
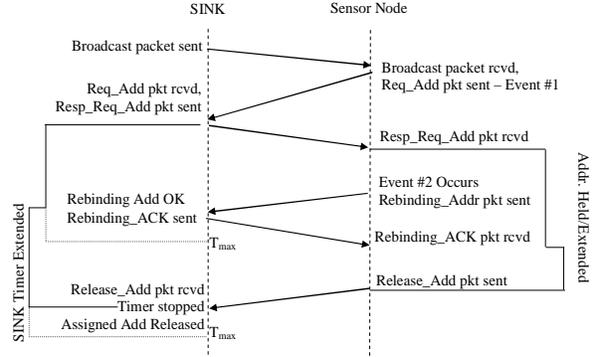

Figure 2. A Typical Address Rebind and Address Release

E. Address Rebind Phase

In order to further increase the reuse of addresses and reduce the overhead involved in address allocation we allow nodes to also rebind an address. The overhead of rebinding an address is much less in comparison to the overhead involved in address request/allocation. The condition for the extension of a lease is the occurrence of a new event prior to the release (i.e., expiry of the lease) of the current address. On the occurrence of a new event the sensor node sends out an *address_rebind* message of the form {*Source, Type, event_ID, Address, moteDist_i*}. The fields have the same meaning as in earlier phases.

Forwarding nodes employ the distance rule using *moteDist$_i$*. When the sink receives the *address_rebind* message it checks to see that the address has not already been released on the Sink (i.e., the *lease_timer* has expired). If the address is still bound on the sink it responds with an acknowledgement (*rebind_ack*) message of the form {*Source*, *Type*, *Address*, *moteDist$_i$*, *moteDist$_{Source}$*}. *Address* represents the address that has been rebound and *mote_Dist$_{Source}$*, *Source* and *event_ID* are used for forwarding and to identify the destination sensor node. When the sink receives an *address_rebind* it resets the *lease_timer* and the address is held for a period T$_{max}$ from the time of rebind. We reset the *lease_timer* as failure to do this can result in the address being released on the Sink prior to release on the sensor node. It is also imperative that the *lease_timer* is associated with the current event. A sensor node assumes a successful rebind only after the receipt of a *rebind_ack* message. The arrival of the *rebind_ack* message does not have to be prior to completion of communication associated with the previous event. However, if communication of the previous event is completed prior to the receipt of *rebind_ack* then an *address_release* message is not sent. Instead on completion the node sends an *address_Request* message. The rationale behind this design is that in the event of a *rebind_ack* arriving after the completion of the communication associated with the previous event the address is still usable as it has been bound (for a period T$_{max}$) on the Sink. There is also not a need to explicitly release the address in the event of the *address_rebind* message being lost as this is addressed by the use of the *lease_timer* (worst case scenario). Further, the duplicate *address_request* will be ignored by the sink. A typical address rebind scenario with no packet loss is presented in Figure 2.

IV. PROBABILITY OF ADDRESS ALLOCATION

Our dynamic addressing protocol issues and renews a network level address with respect to a sensor's request; and, every allocated address will be recycled by the sink on a given timeout value *T$_{max}$*. We assume a maximal message loss rate is observable across the network.
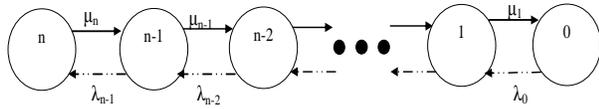

Figure 3. State Machine of A Sink with *n* Address

Suppose a sink has n addresses, we then construct a state machine consisting of n+1 state - states "n" to "1" guarantee the address allocation/renewal and state "0" means denial of request. For each address $1 \le i \le n$, two transitive actions λ_{i-1} and μ_i are associated with address release/timeout events and address allocation/renewal events respectively.

According to the definition of Markov Chain [12], the above state machine corresponds to the following equation of a probability function set P(t) and a transition matrix Q(t):

$$P'(t) = P(t) \times Q(t) \tag{2}$$

where $P(t) = [P_n(t), P_{n-1}(t), \dots, P_0(t)]$ and

$$Q(t) = \begin{bmatrix} -\lambda_0 & \lambda_0 & 0 & 0 & 0 & 0 & \dots \\ \mu_1 & -(\lambda_1+\mu_1) & \lambda_1 & 0 & 0 & 0 & \dots \\ 0 & \mu_2 & -(\lambda_2+\mu_2) & \lambda_2 & 0 & 0 & \dots \\ 0 & 0 & \mu_3 & -(\lambda_3+\mu_3) & \lambda_3 & 0 & \dots \\ 0 & 0 & 0 & \mu_4 & -(\lambda_4+\mu_4) & \lambda_4 & \dots \\ \dots & \dots & \dots & \dots & \dots & 0 & \dots \end{bmatrix}$$

$$\tag{3}$$

For $1 \le i \le n$, we get these two equations after matrix multiplication:

$$\begin{cases} P_0'(t) = -\lambda_0 \cdot P_0(t) + \mu_1 \cdot P_1(t) \\ P_i'(t) = \lambda_{i-1} \cdot P_{i-1}(t) - (\lambda_i + \mu_i) \cdot P_i(t) + \lambda_{i+1} \cdot P_{i+1}(t) \end{cases} \tag{4}$$

At any given time t, the sink either issues an address or denies the request, i.e., $P_n(t) + P_{n-1}(t) + \dots + P_0(t) = 1$. At the initial moment $t = 0$ after the Boot Up phase, the sink has n address, so $P_n(0) = 1$ and $P_i(0) = 0$. The sensor network reaches state "0" if most addresses are allocated and many request and/or rebinding messages arrive simultaneously. On any stabilized network, the derivative of the probabilities of each state tends to be zero. Therefore, we have

$$\begin{cases} 0 = -\lambda_0 \cdot P_0(t) + \mu_1 \cdot P_1(t) \\ 0 = \lambda_{i-1} \cdot P_{i-1}(t) - (\lambda_i + \mu_i) \cdot P_i(t) + \lambda_{i+1} \cdot P_{i+1}(t) \end{cases} \tag{5}$$

Solving these differential equations, we will get the solution for the possibility of successfully acquiring an address from the sink

$$P_1(t) = \frac{\lambda_0}{\mu_1} P_0(t)$$

$$P_2(t) = \frac{\lambda_1 \cdot \lambda_0}{\mu_2 \cdot \mu_1} P_0(t) \tag{6}$$

$$P_3(t) = \frac{\lambda_2 \cdot \lambda_1 \cdot \lambda_0}{\mu_3 \cdot \mu_2 \cdot \mu_1} P_0(t)$$

$$P_i(t) = \frac{\lambda_{i-1} \cdot \lambda_{i-2} \cdot \cdots \cdot \lambda_0}{\mu_i \cdot \mu_{i-1} \cdot \cdots \cdot \mu_1} P_0(t)$$

Hence, we derive the limit of $P_0(t)$ when $t \to \infty$ as:

$$\lim_{t \to \infty} P_0(t) = \left(1 + \sum_{i=1}^{n} \frac{\lambda_{i-1} \cdot \lambda_{i-2} \cdot \cdots \cdot \lambda_0}{\mu_i \cdot \mu_{i-1} \cdot \cdots \cdot \mu_1} \right)^{-1} \tag{7}$$

The successful possibility of address acquisition, i.e., the sum of possibilities of all states other than state "0", is equal to

$$1 - P_0(t) = \frac{\sum_{i=1}^{n} \frac{\prod \lambda_{i-1}}{\prod \mu_i}}{1 + \sum_{i=1}^{n} \frac{\prod \lambda_{i-1}}{\prod \mu_i}} = \frac{1}{\left(\sum_{i=1}^{n} \prod_{i=1}^{n} \frac{\lambda_{i-1}}{\mu_i} \right)^{-1} + 1} \qquad (8)$$

The value of success probability of address allocation increases provided a bigger ratio of the product of λ_{i-1} over the product of μ_i, when the success probability value approaches to 1; and vice versa, a smaller ratio of the product of λ_{i-1} over the product of μ_i gives a success probability close to 0. The value of λ_i depends on the frequencies of address release events and timeout events; and the value of μ_i depends on the frequencies of address request events and rebind events. The ratio of λ_{i-1}/μ_i is affected by message loss rate of the network - the loss of request and rebind messages decreases μ_i and the loss of release messages increases λ_{i-1}, and the increment of the ratio depends on which loss rate is dominant. Suppose N independent events occurs at a frequency of $1/t_{min}$ and each lasts t_{max}, together with n available addresses and the timeout value T_{max} at the sink, we derive the ratio of λ_{i-1}/μ_i in the following scenarios:

No loss transmission assumes no packet loss across the network. Thus, $N - i$ request events, $n-i+1$ rebind events, release events and timeout events occur at frequencies of $1/t_{min}$, $1/|T_{max} - t_{max}|$, $1/t_{max}$ and $1/T_{max}$ on the network respectively. The ratio of λ_{i-1}/μ_i becomes

$$\frac{\lambda_{i-1}}{\mu_i} = \frac{\frac{n-i+1}{t_{max}} + \frac{n-i+1}{T_{max}}}{\frac{N-i}{t_{min}} + \frac{n-i+1}{|T_{max} - t_{max}|}} \qquad (9)$$

Symmetric loss transmission assumes consistent packet loss across the network at the rate c. So, $(1-c)(N-i)$ request events, $(1-c)(n-i+1)$ rebind events and release events occur at frequencies of $1/t_{min}$, $1/|T_{max}-t_{max}|$ and $1/t_{max}$ respectively. The $n-i+1$ timeout events are not affected by the transmission loss and occur at the frequency of $1/T_{max}$. Therefore, the ratio of λ_{i-1}/μ_i becomes

$$\frac{\lambda_{i-1}}{\mu_i} = \frac{\frac{n-i+1}{t_{max}} + \frac{n-i+1}{(1-c) \times T_{max}}}{\frac{N-i}{t_{min}} + \frac{n-i+1}{|T_{max} - t_{max}|}} \qquad (10)$$

Asymmetric loss transmission assumes messages are lost at different rates - c_1 from the sink to nodes, c_2 from nodes to the sink. So, $(1-c_1-c_2)(N-i)$ request events, $(1-c_1-c_2)(n-i+1)$ rebind messages and $(1-c_2)(n-i+1)$ release messages occur at frequencies of $1/t_{min}$, $1/|T_{max}-t_{max}|$ and $1/t_{max}$ respectively. The $n - i + 1$ timeout events are not affected by the transmission loss and occur at the frequency of $1/T_{max}$. Therefore, the ratio of λ_{i-1}/μ_i becomes

$$\frac{\lambda_{i-1}}{\mu_i} = \frac{\frac{(1-c_2) \times (n-i+1)}{t_{max}} + \frac{n-i+1}{T_{max}}}{\frac{(1-c_1-c_2) \times (N-i)}{t_{min}} + \frac{(1-c_1-c_2) \times (n-i+1)}{|T_{max} - t_{max}|}} \qquad (11)$$

Formula (9) (10) (11) suggest that the success rate of address allocation, essentially $\sum_{i=1}^{n} \prod_{i=1}^{n} \frac{\lambda_{i-1}}{\mu_i}$, increases when $|T_{max}-t_{max}|$ or t_{min} increases, or t_{max} or T_{max} decreases. Because t_{min} and t_{max} are determined by the events, the only viable approach is to configure T_{max}, so that the address can be timed out immediately after events finish, which is the focus of our future work on adaptive lease mechanisms.

V. CONCLUSIONS AND FUTURE WORK

In this work we have presented an addressing mechanism that is capable of achieving address reuse. Initial simulation results (not presented due to space limitations) show the scheme to offer an address reuse factor of more than 2.5 with minimal address allocation delays (around 0.6secs). In our future work we hope to evaluate the performance of the scheme through more detailed simulations and investigate adaptive leases.

REFERENCES

[1]. K Lorincz et al., 'Sensor Networks for Emergency Response: Challenges and Opportunities', IEEE Pervasive computing, Volume 3 (4), Oct-Dec 2004, pp 16-22.
[2]. Vijay Kumar et al., 'Robot and Sensor Networks for First Responders', IEEE Pervasive computing, Volume 3 (4), Oct-Dec 2004, pp 23-33.
[3]. M. Ali and Z. A. Uzmi, "An Energy-Efficient Node Address Naming Scheme for Wireless Sensor Networks", Proceedings of the IEEE International Networking and Communications Conference (INCC 2004), June 2004.
[4]. C. Schurgers, G.Kulkarni, M.B. Srivastava, "Distributed On-Demand Address Allocation in Wireless Sensor Networks", IEEE Transactions on Parallel and Distributed Systems, Volume 13 (10), Oct 2002, pp 1056-1065.
[5]. S. PalChaudhuri et al., "TreeCast: A Stateless Addressing and Routing Architecture for Wireless Sensor Networks", Proceedings of the 18th International Parallel and Distributed Processing Symposium (IPDPS'04), 2004.
[6]. E. Cayrici, "Data Aggregation and Dilution by Modulus Addressing in Wireless Sensor Networks", IEEE Communication Letters, Volume 7(8), August 2003.
[7]. S. Motegi et al., "Implementation and Evaluation of On-Demand Address Allocation for Event-driven Sensor Network", Proceedings of the 2005 Symposium in Applications and the Internet (SAINT'05), January 2005.
[8]. J. Jobin et al., "A Scheme for the Assignment of Unique Addresses to Support Self-Organisation in Wireless Sensor Networks", Proceedings of the 60th Vehicular Technology Conference (VTC 2004), September 2004.
[9]. T. Huynh and C. Hong, "A Novel Addressing Architecture for Wireless Sensor Network", 24th IEEE Performance Computing and Communications Conference (PCCC 2005), April 2005.
[10].Y. B. Ko and N. H. Vaidya, "Location Aided Routing (LAR) in Mobile Ad Hoc networks," Proceedings of ACM/IEEE MOBICOM '98, Oct.1998, pp. 66-75.
[11].John G. Kemeny and J. Laurie Snell, Finite Markov Chains, Published Princeton, N.J.: Van Nostrand, 1960