Swarm Intelligence based optimization

Of

MANET cluster formation

By

Anandamoy Sen

A Thesis Submitted to the Faculty of the

ELECTRICAL AND COMPUTER ENGINEERING

In Partial Fulfillment of the Requirements
For the Degree of

MASTER OF SCIENCE

In the Graduate College

THE UNIVERSITY OF ARIZONA

2 0 0 6

STATEMENT BY AUTHOR

This thesis has been submitted in partial fulfillment of requirements for an advanced degree at The University of Arizona and is deposited in the University Library to be made available to borrowers under rules of the Library.

Brief quotations from this thesis are allowable without special permission, provided that accurate acknowledgment of source is made. Requests for permission for extended quotation from or reproduction of this manuscript in whole or in part may be granted by the head of the major department or the Dean of the Graduate College when in his or her judgment the proposed use of the material is in the interests of scholarship. In all other instances, however, permission must be obtained from the author.

SIGNED: ____________________________

APPROVAL BY THESIS DIRECTOR

This thesis has been approved on the date shown below:

______________________________________ __________________

Dr. Bernard P. Zeigler Date

ACKNOWLEDGEMENTS

I wish to express my sincere gratitude to my thesis advisor Dr. Bernard P. Zeigler for guiding me through every step of the thesis and providing me direction and insight on numerous occasions during the course of this work. I thank Dr. Salim Hariri, Dr. Young Jun Son and Dr. Moon-Ho Hwang for taking out time to review this document and participate in the thesis defense committee. I would also like to thank my wife, Nabanita, for her constant support and appreciation. I am indebted to her and our parents for all the encouragement. May I take this opportunity to thank my friends Harish Krishnamoorthy, Hari Venkataramani and Vamsi Sata for supporting me in my effort to complete this thesis.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

The study of Mobile Ad-hoc Network remains attractive due to the desire to achieve better performance and scalability. This thesis describes a Swarm Intelligence inspired method of ad-hoc clustering to give a hierarchical structure to flat MANET. The proposed clustering algorithm derives it's method of operation from ant behavior in their colonies. The algorithm originates from the findings of entomologists who, on observing the ant societies, have remarked that larvae and food are not scattered randomly about the nest, but in fact are sorted into homogeneous piles. The emergence of higher form and behavior from the collaborative operation of numerous entities of trivial intelligence makes this algorithm distributed, adaptive and scalable. The algorithm is devised to be independent of the MANET routing algorithm. Depending upon the context, the clustering algorithm may be implemented in the routing or in higher layers.

In most applications of MANET, the node capabilities are constrained and the node function is heterogeneous during operation. This thesis identifies the various *Roles* played by nodes and uses that information to build a *Role* based routing model which takes the form of clusters. The dynamic formation of clusters helps reduce data packet overhead, node complexity, power consumption, and create multi-path routing. After cluster formation, specific nodes are elected as cluster-heads satisfying certain *Roles* and performance criteria. This thesis looks at the performance of the proposed clustering algorithm, when applied to random and pseudo-random mobility models. Studies on mobility models have indicated that temporally and spatially correlated mobility models with geographic restrictions are nearer to real life scenarios when compared to fully-random models. The proposed algorithm is proven to improve performance in pseudo-random mobility models.

Chapter 1

INTRODUCTION

1.1 Challenges in Mobile Ad-hoc Networking

1.1.1 About MANET

MANET is a collection of wireless mobile nodes, which dynamically form a temporary network, without using any existing network infrastructure or centralized administration. These are often called infrastructure-less networking since the mobile nodes in the network dynamically establish routing paths between themselves. Current typical applications of a MANET include battlefield coordination and onsite disaster relief management.

1.1.2 End-to-End service over Hop-by-Hop infrastructure

Current trend in networking shows a paradigm shift towards end-to-end service. Given the plethora of network services either implemented or in the stage of development, it is nearly impossible to design a networking infra-structure keeping all of them in mind. Such an attempt will result in huge information being stored per hop to understand and route traffic and this will result in an explosion of state information per hop. The right way to approach the problem is to concentrate on designing a networking infrastructure which will satisfy a few basic requirements and constraints. *Then* various end-to-end services can then be built over that infrastructure.

1.1.3 Quality of Service guarantees in MANET

The current MANET protocols don't address this topic directly. To better understand the Quality of Service in MANET, it is imperative that we understand how Quality of Service is currently provided in wire-line networks. In the field of packet-switched networks and computer networking, the traffic engineering term **Quality of Service** (**QoS**) refers to the probability of the telecommunication network meeting a given traffic contract, or in many cases is used informally to refer to the probability of a packet succeeding in passing between two points in the network. A traffic contract (**SLA**, **Service Level Agreement**) specifies the assurances for the ability of a network/protocol to give guaranteed performance/throughput/latency bounds based on mutually agreed measures, usually by prioritizing traffic. Broadly speaking there are two types of service which require QoS: (1) **Elastic** and (2) **Inelastic**. Inelastic service, meaning that they require a certain level of bandwidth to function - any more than required is unused, and any less will render the service non-functioning. By contrast, *elastic* applications can take advantage of however much or little bandwidth is available. MANET is expected to provide QoS for Elastic services. In general QoS is the result of "**coordinated effort**" between various network entities. Such an effort initiative may be formally defined in the form of **SLA** between network entities in the case of wire-line networks. For MANET the concept of QoS would result from "**coordinated effort**" between various nodes.

1.2 Role based Routing in clustered MANET

Ad-hoc mobile nodes, though they appear homogeneous, play specific roles. The number of possible roles is finite and is a relatively small set {**OBS**, **SRC**/**SINK** and **FWD**}. Mathematically,

Let R be set of roles.

R = {OBS, SRC/SINK, FWD}

OBS = acts as an OBSERVER, uses the broadcast nature of nodes to sniff into neighbor transmissions.

SRC/SINK = acts as a packet SOURCE or SINK.

FWD = acts as a FORWARDING ENGINE.

Let the number of nodes in the experimental frame be N.

Let the nodes be numbered as $T_1 ... T_N$

At any point of time the nodes can be in any 1 or more roles as defined in the following Table 1.1. The total number of available Cumulative Roles is given by:

$$C_1^3 + C_2^3 + C_3^3 = 7$$

Out of these 7 Cumulative Roles, not all are valid.

CASE	Cumulative Role(s) at any given time t	VALID
1	OBS + SRC/SINK + FWD	NO
2	OBS + SRC/SINK	NO
3	OBS + FWD	NO
4	**SRC/SINK + FWD**	**YES**

5	**SRC/SINK**	**YES**
6	**FWD**	**YES**
7	**OBS**	**YES**

Table 1.1: Table of Roles

The **Cumulative Role(s)** may or may not vary over time. Identification of time-invariant (or mostly time-invariant) Cumulative Roles will pave the way to Role based Routing in MANET. **Role based Routing** *is defined as the method by which the MANET entities change their addressing and/or forwarding schemes to better fit the role that has been dynamically acquired or assigned.* Clustering of ad-hoc nodes on the basis of roles is proposed in this thesis.

1.3 Contributions of the thesis

It has been observed that Ad-hoc mobile nodes, though they appear homogeneous, play specific *Roles* in making data transfer related decision e.g. routing, forwarding etc. The number of possible *Roles* is finite and currently is a relatively small set. These small set of *Roles* may or may not vary over time. Proper identification of time-invariant (or mostly time-invariant) *Roles* will pave the way for a better MANET because specific nodes can now be assigned specific tasks.

Clustering of Ad-hoc nodes on the basis of *Roles* is discussed in this thesis. Since the size of an Ad-hoc network is boundless, partitioning the network into clusters makes it more manageable. Intra and inter-cluster interaction holds the key to hierarchical network service architectures like QoS and Multicast. Swarm Intelligence based Ant Clustering, is a probabilistic optimization technique for solving computational problems which can be reduced to relative organization of node positions in a graph.

This thesis exploits the above mentioned behavior of Ant optimization to form clusters. This work proposes a Swarm Optimized Algorithm for Clustering (SOAC) to identify MANET node *Roles* and use it for emergence of hierarchical structure in flat MANET.

Chapter 2

PROBLEM FORMULATION

2.1 Problem Description

Current MANET routing protocols have a flat approach to routing needs. Though MANET nodes play various roles (as seen later), current routing protocols have little or no consideration for that. The problem statement can be formalized as:

Can *identification of Roles* played by MANET nodes and *dynamic allocation of routing responsibilities* in tune with the *identified Role* improve the traffic characteristics as compared to flat routing?

2.2 Visualization using NS2

To prove the presence of Roles in MANET, a simulation environment in NS2 has been created. The environment consists of 5+1 nodes, Random Waypoint mobility model and AODV routing protocol. Even in a simple scenario like this, the nodes converge towards specific roles. Node 5 takes on the Role of a forwarding engine (FWD), nodes {0, 1, 2, 3, 4} take on the Roles of SRC/SINK. There are no observers (OBS) in this scenario. Figure 2.1 shows the traffic characteristics of the experimental framework. The next figure, 2.2, makes it possible to easily identify the *Roles* of the individual nodes as explained.

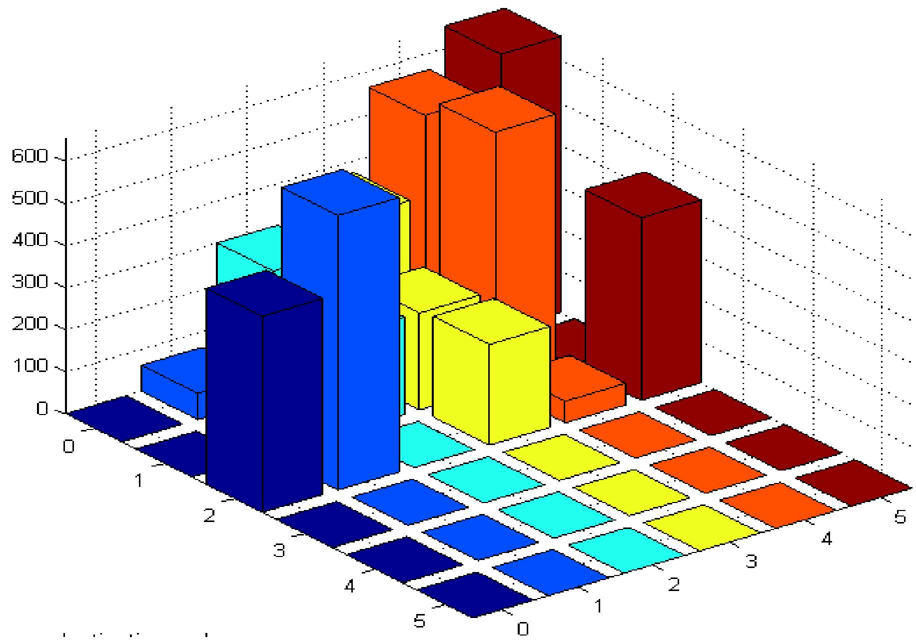

Figure 2.1: Traffic generation characteristics in a 5+1 node MANET over AODV

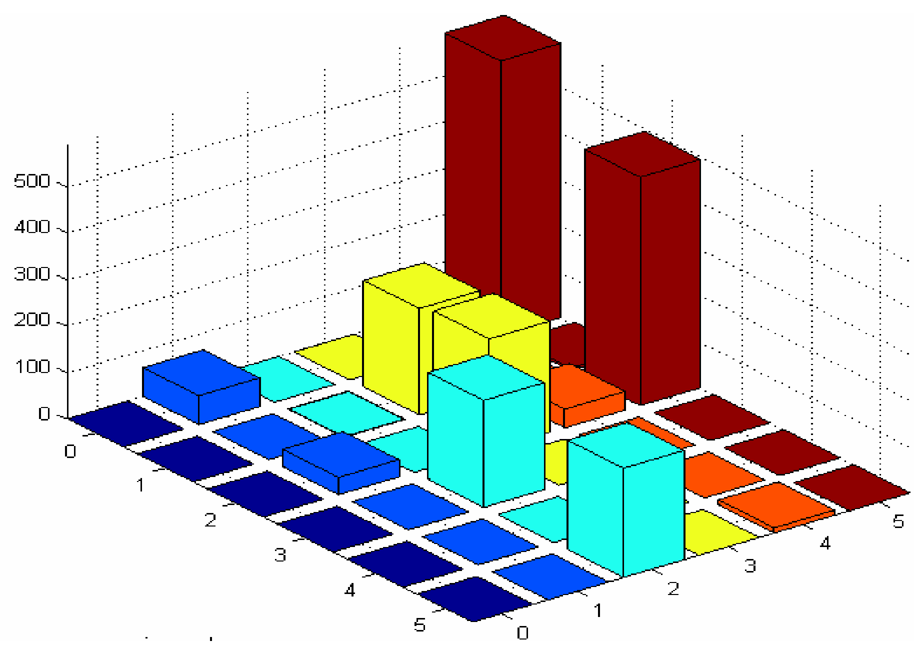

Figure 2.2: Traffic forwarding characteristics in a 5+1 node MANET over AODV

Chapter 3

BACKGROUND INFORMATION

3.1 About Swarm Intelligence

3.1.1 Swarms and Emergence

Swarm Intelligence (SI) [5] is an artificial intelligence technique based *around* on the study of collective behavior in decentralized, self-organized systems. The expression "swarm intelligence" was introduced by Beni & Wang in 1989, in the context of cellular robotic systems.

SI systems are typically made up of a population of simple agents interacting locally with one another and also with their environment. Usually there is no centralized control structure dictating how the individual agents should behave, but local interactions between such agents often lead to the emergence of a global behavior. Examples of systems like this can be found in nature, including ant colonies, bird flocking, bee swarming, animal herding, bacteria molding and fish schooling.

3.1.2 Swarm Intelligence as an optimization method

Two of the most successful swarm intelligence techniques [6] currently in existence are **Ant Colony Optimization** (ACO) and **Particle Swarm Optimization** (PSO). ACO is a meta-heuristic optimization algorithm that can be used to find approximate solutions to difficult combinatorial optimization problems. In ACO artificial ants build solutions by moving on the problem graph and they, mimicking real ants, deposit artificial **pheromone** on the graph in such a way that future artificial ants can build better solutions. ACO has been applied successfully to an impressive number of optimization problems. PSO is a global minimization technique for dealing with problems in which a best solution can be represented as a point or surface in an n-dimensional space.

3.1.3 Use of Ant Colony Optimization technique

ACO is the primary optimization algorithm being used here [7]. The reason behind choosing ACO is its distributed nature and inherent randomness. The algorithm used for optimization is a purely distributed algorithm. This holds serious implications for MANET. Since in a MANET the nodes are constrained by power, storage and processing power limitations, a purely distributed algorithm like ACO prevents per node computation load, reduces state information to be stored per node. The randomness built in the algorithm prevents convergence to local maxima/minima but has a tendency to search for global optimization of parameters.

3.2 Ad-hoc Mobility Models

3.2.1 Introduction

The mobility model [3] is designed to describe the movement pattern of mobile users, and how their location, velocity and acceleration change over time. Figure 3.1 shows the different kinds of mobility models. Since mobility patterns may play a significant role in determining the protocol performance, it is desirable for mobility models to emulate the movement pattern of targeted real life applications in a reasonable way. Otherwise, the observations made and the conclusions drawn from the simulation studies may be misleading. Thus, when evaluating MANET protocols, it is necessary to choose the proper underlying mobility model. Therefore, it is necessary to have a deeper understanding of mobility models and their impact on protocol performance.

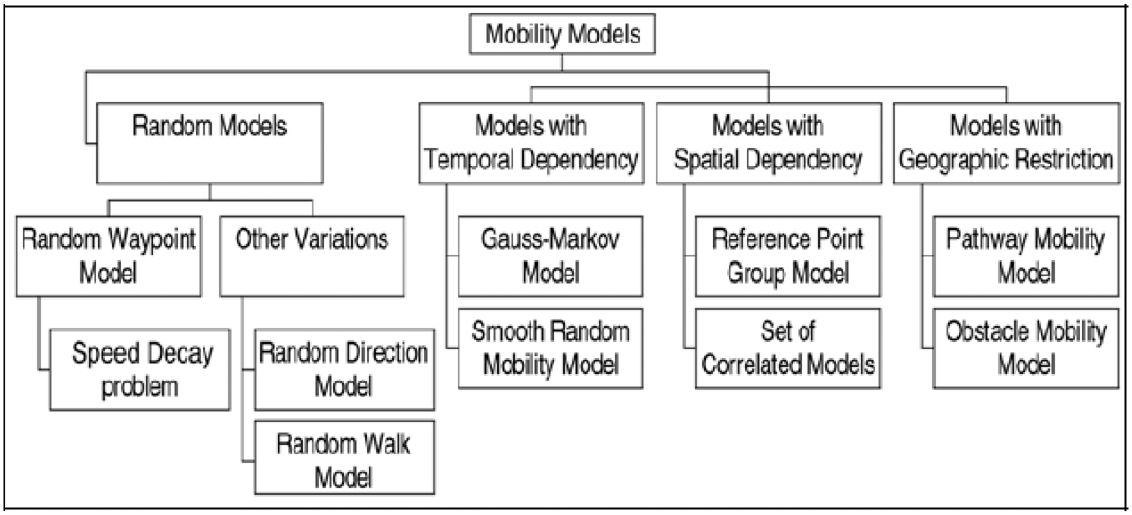

Figure 3.1: Categories of Mobility Models in mobile ad-hoc networks

3.2.2 Categories of Mobility Models in MANET

3.2.2.1 Trace based Mobility Models

A good method of creating mobility models is to construct **trace based models**. These models are constructed from traces of real-life ad-hoc network and its node movement. Since MANETs haven't been deployed on a wide-scale, obtaining real-life mobility traces becomes a major challenge. So, the current trend is to propose different mobility models in an effort to capture the unique characteristics of real-life ad-hoc node movement. These models are sometimes termed as **synthetic models**. As opposed to wireless cellular networks, ad-hoc nodes in MANET are examined at a deeper level to analyze the effect of location, velocity with respect to other nodes to determine when peer-peer links are formed and broken.

3.2.2.2 Random Models

We briefly discuss the characteristics of the most widely used Random Models; **Random Waypoint Model** and two of its variants; **Random Walk** and **Random Direction**.

The NS2 implementation of this mobility model is as follows: as the simulation starts, each mobile node randomly selects one location in the simulation field as the destination. It then travels towards this destination with constant velocity chosen uniformly and randomly from [0, V_{max}], where the parameter V_{max} is the maximum allowable velocity for every mobile node. The velocity and direction of a node are chosen independently of other nodes. Upon reaching the destination, the node stops for a duration defined by the 'pause time' parameter. If T_{pause}=0, it leads to continuous mobility. After this duration, it again chooses another random destination in the simulation field and moves towards it. The whole process is repeated again and again until the simulation ends.

The key parameters are: V_{max} and T_{pause}

Definition of Mobility Metric: Measure of relative speed between node *i* and *j* at time *t*.

$$RS(i, j, t) = | V_i(t) - V_j(t) |$$

The Mobility Metric ($\overline{M}$) is formalized as:

$$\overline{M} = \frac{1}{|i, j|} \sum_{i=1}^{N} \sum_{j=i+1}^{N} \frac{1}{T} \int_0^T RS(i, j, t) dt$$

Where |i,j| are the number of distinct node pairs (i , j)

The Random Walk model was originally proposed to emulate the unpredictable movement of particles in physics. It is also referred to as Brownian motion. This model can be thought of as a Random Waypoint model with zero pause time. Random Walk model differs from the Random Waypoint model in a way that the nodes change their speed and direction at each time interval. Therefore at each time interval *t* the node randomly and uniformly chooses its new direction *θ(t)* from (0, 2Π]. Similarly the new speed *v(t)* follows a uniform distribution or a Gaussian Distribution from [0, V_{max}]. At any discrete time interval *t* the velocity vector looks like:

$$\{v(t).\cos\theta(t), v(t).\sin\theta(t)\}$$

On reaching the boundary the node is reflected back into the simulation field with an angle *θ(t)* or *Π - θ(t)*. This is called the **border effect**. The border effect leads to an unbalanced spatial node distribution. At steady state, the node density is highest at the central region whereas it is almost zero at the boundary of the simulation area. The phenomenon is called non-uniform spatial distribution. It has also been observed that the average number of neighbors for a particular node periodically fluctuates with time. It has been proved that the spatial distribution of nodes is not a function of node velocity. This implies that no matter how fast the nodes move, the spatial distribution is only determined by its Cartesian location.

Key features: memory-less, border-effect, not very responsive to clustering algorithms (may be SOAC)

3.2.2.3 Models with Temporal Dependency

Mobility of a node may be constrained and limited by the physical laws of acceleration, velocity and rate of change of direction. Hence, the current velocity of a mobile node may depend on its previous velocity. Thus the velocities of single node at different time slots are ‘correlated'. We call this mobility characteristic the *Temporal Dependency of velocity*.

Random Mobility models fail to capture the mobility characteristics of some real-life scenarios. The primary reason behind the use of Random Models is their simplicity in implementation. The most common limitations are temporal and spatial dependency of velocity and geographic restrictions of movement. The two most common mobility models with temporal dependency are *Gauss-Markov Mobility Model* and *Smooth Random Mobility Model*. Both these models capture the temporal dependency of velocity over time.

3.2.2.4 Models with Spatial Dependency

This type of mobility model captures the team collaboration among various nodes. Therefore the mobility of mobile nodes could be influenced by the neighboring nodes. Thus the velocities of different nodes are correlated spatially. Reference Point Group Mobility Model is a good example of spatially correlated model and is based upon node role division, namely, group leader and group member. The various types of RPGM models are column mobility model, pursue mobility model and nomadic community mobility model.

3.2.2.5 Models with Geographic Restriction

In most real-life scenarios, the movement of nodes is subject to geographic restrictions. The constraints in place for node movement vary according to the environment like freeways, pathways, campus, tunnels etc. The nodes are assumed to move in a pseudo-random way on predefined pathways in the simulation field. Two such mobility models are Pathway mobility model and Obstacle mobility model.

3.2.2.6 Summarizing

	Temporal Dependency	Spatial Dependency	Geographic Restriction
Random Waypoint Model	No	No	No
Reference Point Group Model	No	Yes	No
Freeway Mobility Model	Yes	Yes	Yes
Manhattan Mobility Model	Yes	No	Yes

Table 3.1: The characteristics of mobility models

The mobility models have different properties and different mobility characteristics. As a result, these mobility models behave differently and influence MANET protocol performance in different ways.

3.2.3 Clustering and Logical Hierarchy

3.2.3.1 Benefits of Clustering and formation of Logical Hierarchy

The concept of hierarchical routing is widespread in wire-line routing. A collection of subnets forms a network. A collection of networks forms an autonomous domain. Routing within the autonomous domain is typically managed by an ISP and hidden from the rest of the world. This method has the following benefits:

- It reduces the size of routing table in each and every network entity

- Makes the network more manageable
- Makes networks metrics bounded and deterministic

3.2.3.2 Methods of formation in wire-line networks

Autonomous domains (AS) are formed to serve a particular geographical region. Then the AS is broken up into networks which are further broken down into subnets which consist of the smallest network entities (hosts). Internet backbone routers connect the Autonomous Domains and provide the final level of hierarchical structure to the Internet.

Chapter 4

SWARM OPTIMIZED ANT CLUSTERING

4.1 Concept

Swarm Optimized Ant Clustering (SOAC) is a Swarm Intelligence based concept used as optimization tool for optimal cluster formation in MANET. It works on the principle of collective intelligence and emergence. It's application in graphs and networks involves swarming agents (ants) hopping node to node, analyzing a set of local variables exposed by the nodes and changing the color of the node. A group of neighboring nodes with the same color collectively form a cluster. The SOAC algorithm is limited to node coloring only. A simple token based election algorithm can be used to elect cluster-head among nodes of similar color in radio distance to each other. The cluster-head formation results in an immediate elimination of state information being stored in nodes of same color. It also results in marked improvement in "good-put" (section 5.2.1) as IP & TCP/UDP headers are immediately replaced by labels used for switching. This also gives the node a chance to better utilize the resources using directed smart antennas and getting rid of local network state information.

4.2 Benefits of Swarm

SOAC utilizes local interaction without centralized control. Global representation of the parameters is not required. The algorithm is based on a simple set of rules making it a low computation over-head.

4.3 The Algorithm

The algorithm originates from the findings of entomologists who, on observing societies of ants, have remarked that larvae and food are not scattered randomly about the nest, but in fact are sorted into homogeneous piles. For an explanation of the algorithm please consider that:

- environment is a two-dimensional grid
- mobile nodes are scattered on the grid
- each ant is modeled as an automaton which is able to hop from node to node
- each ant can change the color of the mobile nodes according to probabilistic rules
- the probabilistic rules require only local environmental information as input

Let G represent the grid

Let A represent a finite set of automata

Let M represent a finite set of mobile nodes present in grid G

Let C represent a finite color set

Let N represent the number of neighbors

At each discrete time-step an automaton is generated and takes on the color of the generating node. The mobile nodes are colored, re-colored or made colorless by the automaton according to rules which take local data as input. The constraint on coloring is dependent on node and cluster capacity

When the cluster population reaches a threshold, the member mobile nodes elect a cluster-head.

The **probability P_d (m_i)** that an automaton will discolor (make colorless) a mobile node m increases the more m is isolated

$$P_d = \left(\frac{K_d}{K_d + f_m(u)} \right)^2$$

u = number of unique neighbors

K_d is a constant

$f_m(u)$ is a local density function relevant to node m

The **probability Pr (**m_i**)** that an automaton will recolor mobile node increases with the number of similar colored nodes in the immediate neighborhood

$$P_r = \left(\frac{f_m(u)}{K_r + f_m(u)} \right)^2$$

K_r is a constant

$f_m(u)$ is a local density function relevant to node m

f is a local density function ~[0,1]

Different colored nodes in the neighborhood will lead to overall reduction in value of *f*.

The local density function represents an estimation of the density of same colored nodes in the neighborhood of m.

The dissimilarity function *d* is defined as:

$d(c_X, c_Y) = 0$: When x = y, where x, y belongs to set C

$d(c_X, c_Y) = 1$: When x ≠ y, where x, y belongs to set C

The maximum value of f is reached if and only if for all values of <x,y>, d(cx,cy) = 0, in which case $f(m_i) = 1$

The local density function f is estimated as below:

$$f_m(u) = 1 - \sum_{x,y \in N} \left(\frac{d(c_x, c_y)}{C_2^{|N|}} \right)$$

Parameters:

K_d, K_r are constants which determine the extent to which the probabilities $P_r(m_i)$ and $P_d(m_i)$ depend on the local density function f

Decided on $K_r = 0.10$ by setting up a constraint that when all the neighbors are of same color, $P_r(m) > 0.8$

Similarly K_d was set to 0.5 by constraint that $P_d(mi) > 0.5$ when at least (N-2) neighbors are different

Under these constraints, it can be easily shown that:

$$(P_d + P_r) < 1$$

So, we can deduce another probability P_U that an automaton will leave a node untouched

$$P_U = 1 - (P_d + P_r)$$

This will later be related to mobile node power expenditure estimates for traffic overhead. An isolated state is reached when the number of similar colored vertices in the neighborhood is small. Mathematically,

N_{Sx} = number of neighbor nodes of same color as x

N_N = total number of neighbors

K_{ISO} = constant

$$\max\left(N_{S_i} \big/ N_N \right) \leq K_{ISO} \; \forall i \in C$$

On successful formation of cluster-head, the following happens:

- Member nodes can start registering themselves with one or more cluster-heads
- Member nodes may reduce the routing table to include neighbors and cluster-head only
- Member nodes can forward unknown route packets to cluster-head
- IP header based switching is replaced by label switching

The immediate benefits to the MANET are in the form of reduction in IP header overhead. As member nodes lower transmission power to reach cluster-head only, substantial power saving is possible in MANET. Member nodes registered with more that one cluster-head may act as bridges and provide multi-path fault tolerant routing capabilities.

4.4 Implementation of SOAC Algorithm

4.4.1 Automaton

The automaton is a piece of computer code that is executed on every node. An automaton mimics the movement of ants from node to node. On reaching a node the automaton is presented with an environment which identifies the state of the node. The environment consists of a set of environment variables which hold state and satellite information associated with it.

4.5 Simulation Framework

The framework is in NS2. The automaton code is implemented within the guidelines of the framework. The figure below gives the overall architecture of NS2.

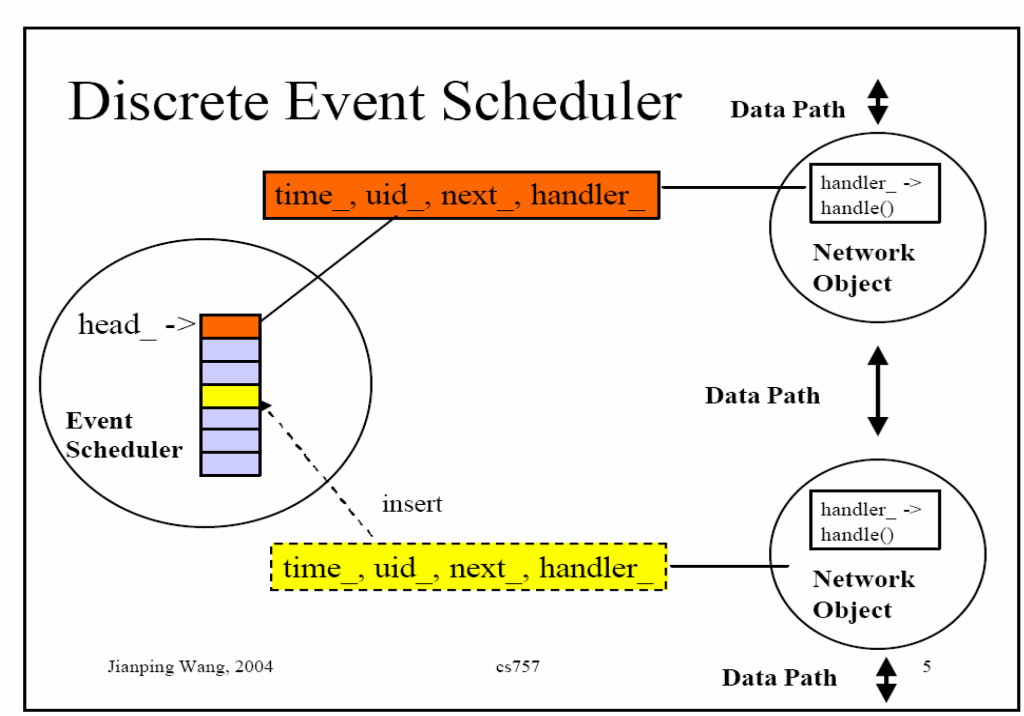

Figure 5.1: Discrete Event Scheduler implementation in NS2

4.5.1 NS2 Nodes, Agents and Packets

The environment is setup by creating a new SOAC agent derived from NS2::Agent. The new agents will generate automaton, allow automaton controlled access to node state variables/tables and forward automaton

4.5.2 Validation of Simulation Framework

This is a proposed algorithm (SOAC), and its implementation is intertwined with the NS2 framework. The framework thus needs to be validated. SOAC validation with NS2 consists of testing the algorithm against two boundary conditions to observe the algorithm behavior. The expected behavior is discussed here.

When all nodes are dynamic (section 5.5.4.1), the nodes move using the Random Walk Mobility Model. Since the movement is random, the expected behavior is that the nodes will lose their color and there will be no cluster formation.

When all nodes are static (section 5.5.4.2), the nodes are not moving. Since there is no movement, the expected behavior is that the nodes within the range to each other will get colored. The color will be decided by the number of neighbors with the same color. Since the each node gets a color probabilistically decided by an uniform distribution, the expected result is that there will be scattered coloring with all colors present.

When SOAC with NS2 is tuned for stability (section 5.5.4.3), the framework has been validated for stable operation. By tuning K_d and K_r, it is possible to have a stable oscillation of the number of colored and uncolored nodes even when the nodes are static, thus maintaining a constant exploration by ants.

4.5.3 Experiments

4.5.3.1 MANET nodes are moving randomly, Random Walk Mobility Model

Experiment: Exp-1-3_run2

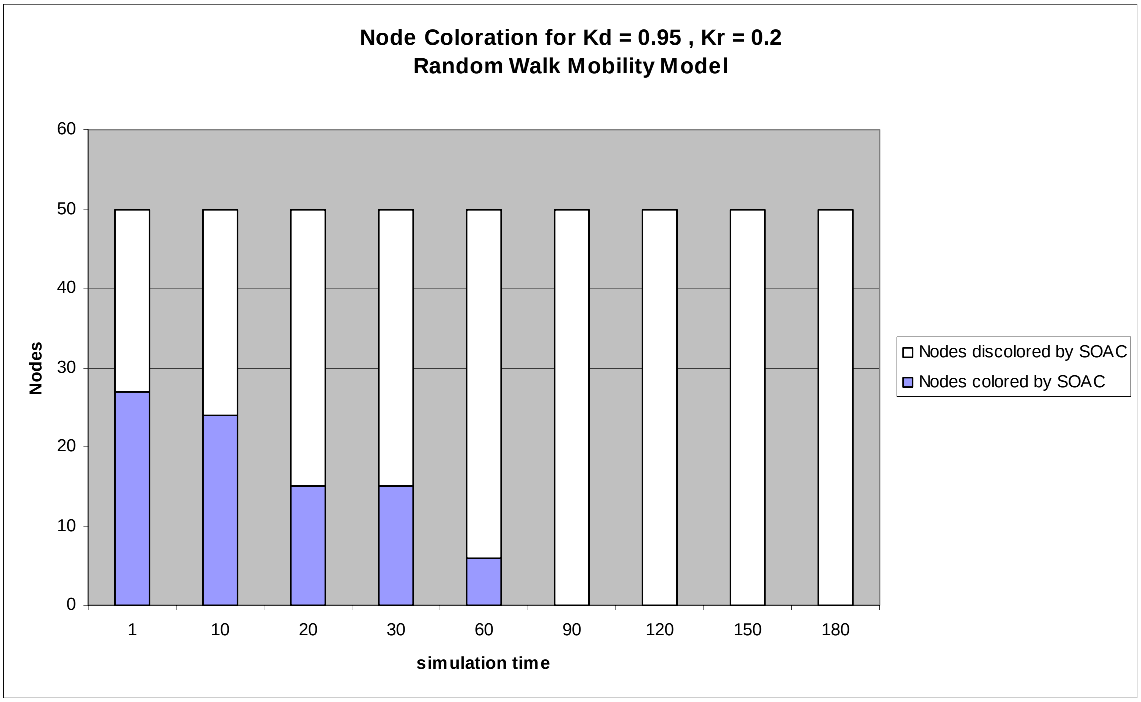

Figure 5.2: MANET nodes are moving randomly and cluster formation is not possible

The nodes are moving randomly and the SOAC implicitly detects the same and the nodes slowly become colorless to prevent SOAC overhead from trying to form clusters when MANET is too random for that. The probability of ant generation is low while a node is colorless. But there is a non-zero probability to generate ants so that, when the colorless nodes are no longer moving randomly, coloring and cluster formation can commence.

Experiment: Exp-1-3_run2

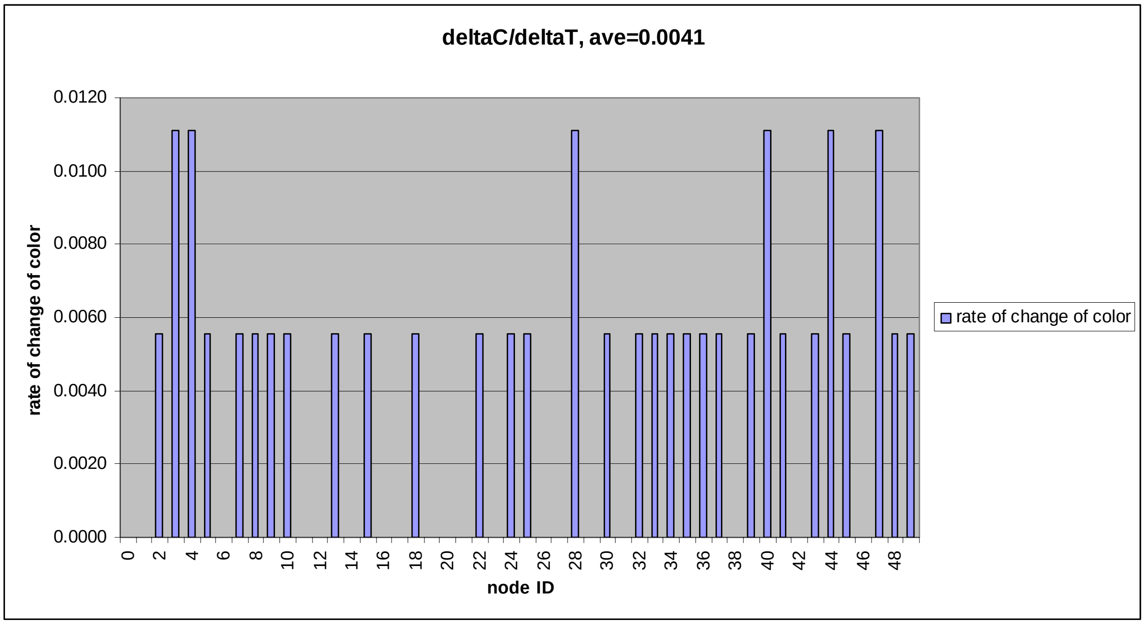

Figure 5.3: Average rate of change of color is high enough to prevent any meaningful cluster formation

4.5.3.2 MANET nodes are static

Experiment: Exp-1-4_run2

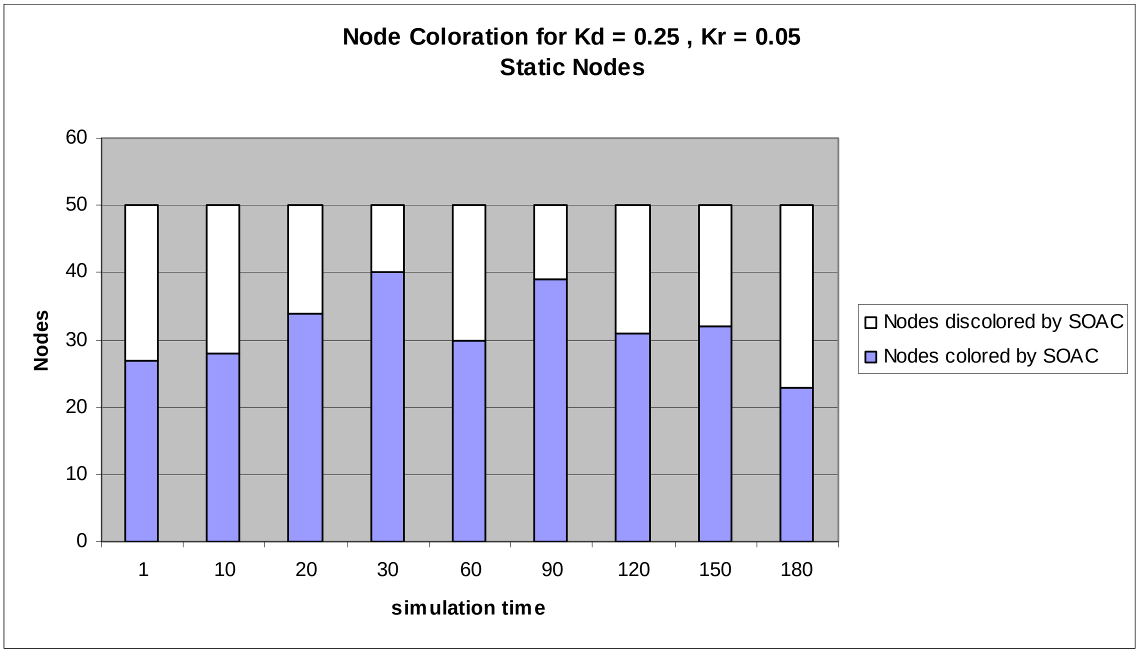

Figure 5.4: MANET nodes are static

Since all nodes are static, there is scattered coloring and the high fluctuation in number of colored nodes is caused by initial SOAC eagerness to recolor nodes as neighbors are static. As nodes common to clusters of different color are recognized by the algorithm, they lose their color and the amplitude of fluctuation of colored nodes tapers off to a mean value (around 22 here).

Experiment: Exp-1-4_run2

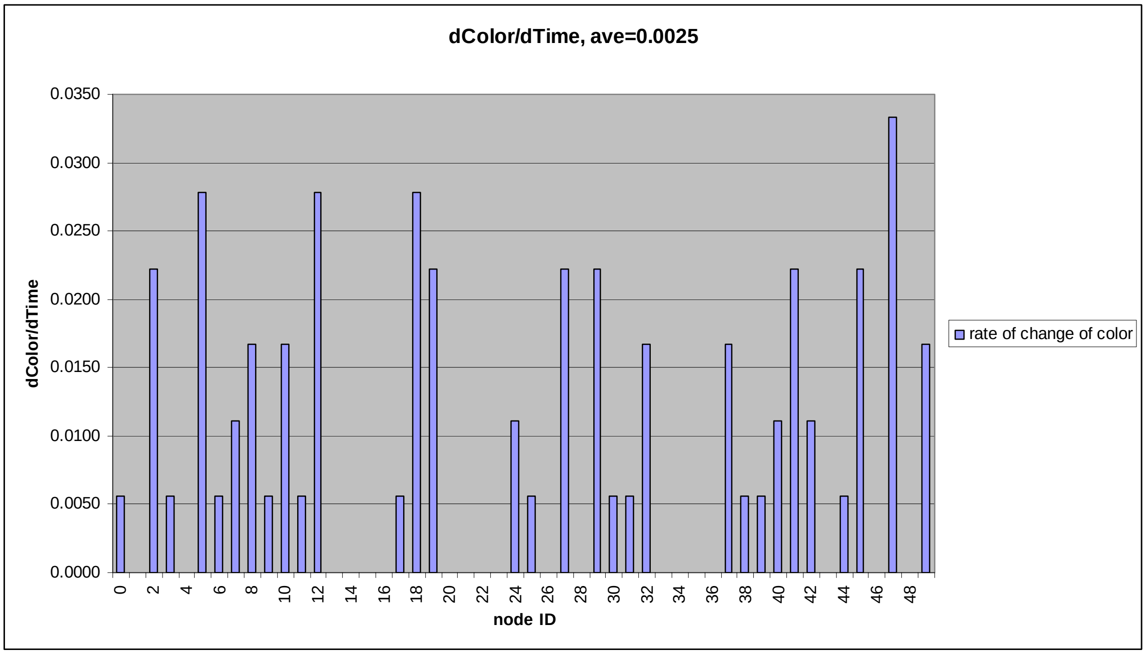

Figure 5.5: Lower rate of change of color favors cluster formation

The rate of change of color is low. The average hovers around 0.0026 for changing values of K_d and K_r but much lower than average of 0.0061 as seen in the case of nodes on Random Walk. There are many nodes where rate of change of color is 0 for the whole simulation period which indirectly points to a relatively static environment. At the same time SOAC keeps the exploration of solution alive as evident by a substantial rate of change of color in other nodes which are within radio distance to each other.

4.5.3.3 MANET nodes in stable oscillation

Experiment: Exp-1-4_run1

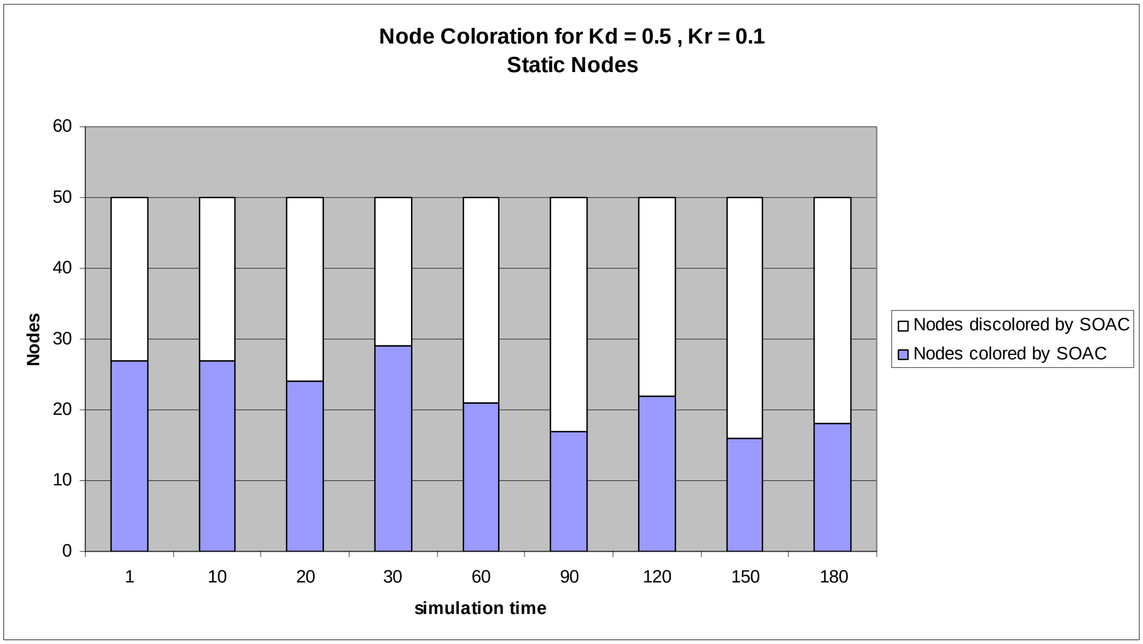

Figure 5.6: With typical values of K_d and K_r the number of colored nodes oscillates around a mean value (approx 18 here)

With these derived [Appendix A] values of K_d and K_r, the number of colored nodes oscillates around a mean value. This MANET framework has been tuned for stable oscillation even when the nodes are static. The reason behind it is to make sure that new automaton (ants) is always exploring nodes to find the optimal cluster.

Experiment: Exp-1-4_run1

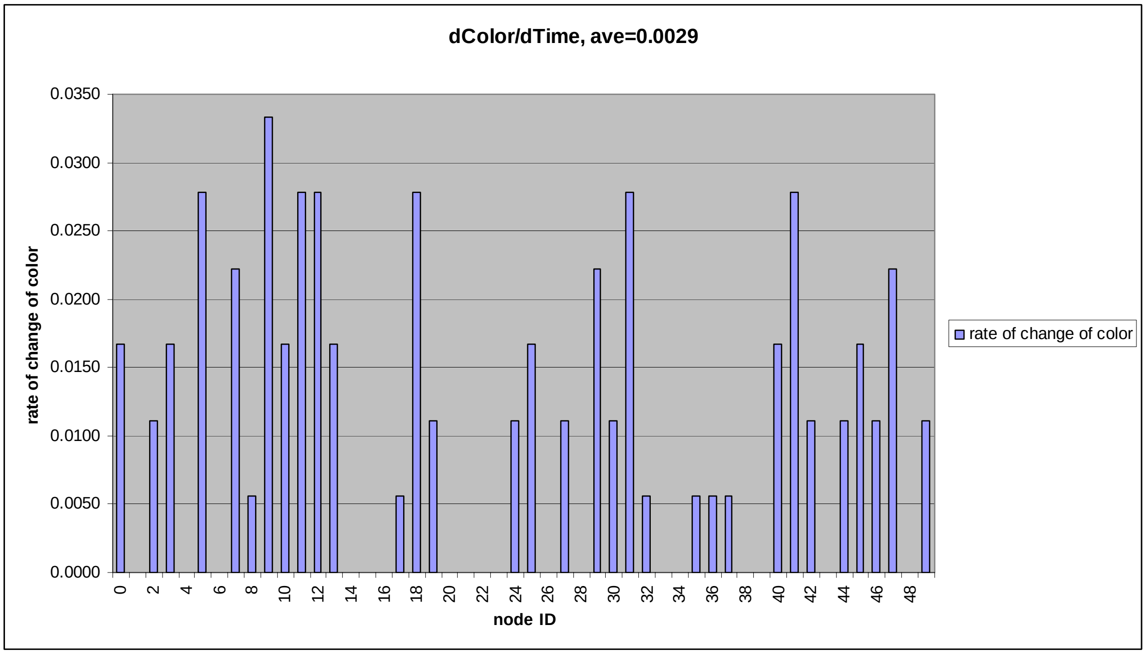

Figure 5.7: Lower rate of change of color favors cluster formation

Chapter 5

PERFORMANCE EVALUATION

5.1 Introduction

The algorithm is evaluated against Known network metrics and SOAC specific network metrics. Comparison is done with typical non-SOAC MANET scenarios. SOAC overhead is calculated for SOAC-MANET and network state information is calculated on a per node basis which has scalability implications. Fault tolerance characteristics of SOAC-MANET are explored. Comparison is done with a similar Swarm inspired routing algorithm.

5.2 Metrics for Evaluation

5.2.1 Known Network Metrics

The known network metrics to be used for performance evaluation are traffic throughput and latency. Compared to routing algorithms with SOAC, the change in average packet latency is due to the fact that the cluster member will use the cluster-head to route the packet. In such cases the worst-case latency is when the packet is being routed within the cluster to maximum of two links before leaving the cluster. Thus added latency due to introduction of SOAC is bounded within [T_{HOP}, $2T_{HOP}$], where T_{HOP} is the average time taken by a whole fixed size data packets to traverse an error-free wireless link from transmitter buffer to receiver buffer. A metric very popular in wireless world is good-put. It can be defined as number of payload data bytes per IP packet. Consider a typical Voice over IP scenario, the data payload is typically 40 bytes equivalent to 40 ms of coded voice using a standard vocoder rate of 8Kbps. The associated IP+UDP+RTP header is 44 bytes (20 + 8 + 16). At MANET node rate of 128 Kbps the actual good-put is:

$$\left(\frac{40}{84}\right)*128 = 60 \text{ Kbps}$$

Using label switching, in this case after SOAC has elected cluster-head, the IP+UDP header is replaced by 2 bytes of switching information. In the same scenario as above, the good-put now increases to:

$$\left(\frac{40}{58}\right)*128 = 88 \text{ Kbps}$$

This improvement is given the fact that the MANET nodes have static IP addresses and static port numbers. In real-life a header compression algorithm like ROHC applied after SOAC cluster-head is elected may yield similar results.

5.2.2 Traffic throughput for MANET-SOAC with increasing speeds

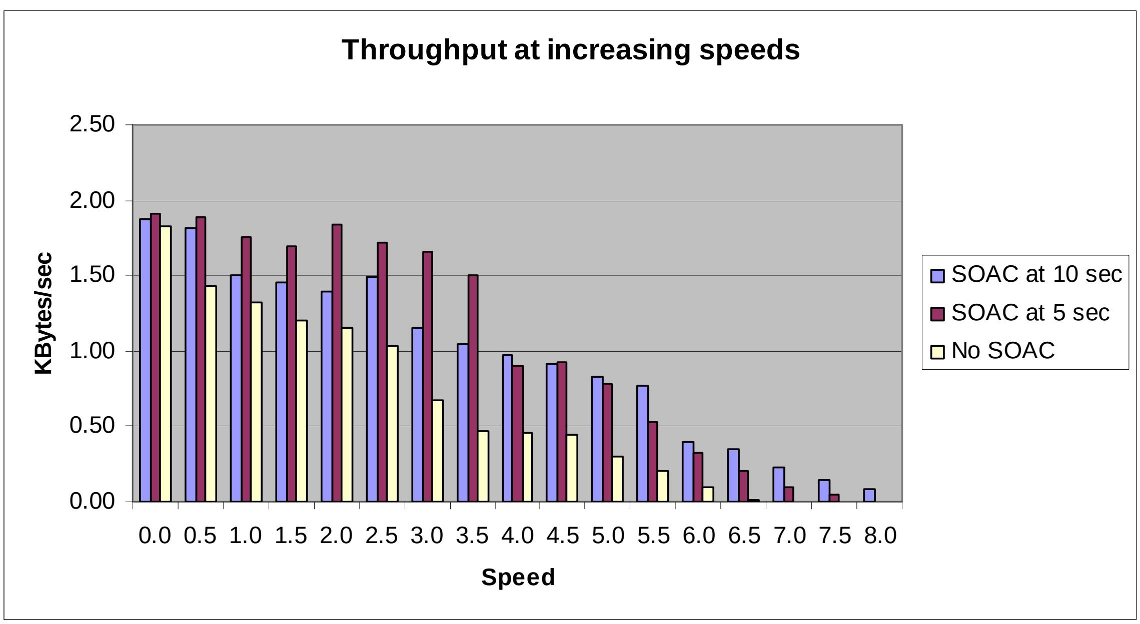

Table 6.1: The variation of MANET thru-put with increasing speed

5.2.3 Latency comparison between MANET and SOAC-MANET

	MANET (% nodes)	SOAC-MANET (% nodes)
Latency < T_{TYP}	17	13
T_{TYP} < Latency ≤ (T_{TYP} + T_{HOP})	69	52
(T_{TYP} + T_{HOP}) < Latency < (T_{TYP} + 2T_{HOP})	11	30
(T_{TYP} + 2T_{HOP}) < Latency	3	5

Table 6.2: The variation of MANET thru-put with increasing speed

5.2.4 SOAC specific metrics

The derived metrics for SOAC based MANET clustering are **isolation**, cluster-density, cluster-stability and clustering-index ratio of [colored/total_nodes]. The derived metrics can be explained as below:

Isolation Index(N_{iso}): It is defined as the size of neighbor table. The neighbor table consists of nodes of different color. This is per node metric.

Cluster-density ($C_{density}$): This is proportional to number of nodes in a particular cluster. The cluster-density varies from cluster to cluster. For any given cluster-head the **$C_{density}$(t)** is a measure of the number of nodes participating in the cluster formation at that given time t. This is per cluster metric.

Cluster-stability ($C_{stability}$): This is defined as the rate of change of **$C_{density}$(t)** with time. Higher values of **$C_{density}$** mean that the cluster is in a transient state.

5.2.5 Relation between SOAC metrics

Using the NS2 simulation framework and logs generated from simulation run, the following observations were made.

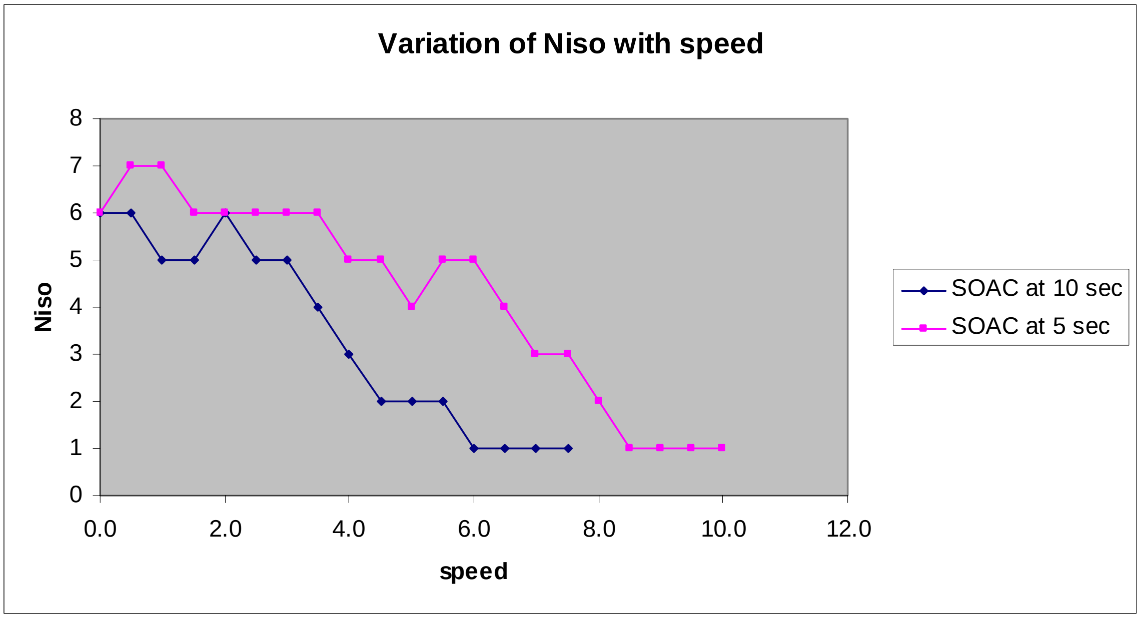

Table 6.3: The isolation parameter N_{iso} variation with speed

For increasing speeds, the rate of fall of N_{iso} is steeper in case of SOAC broadcast interval being low. This is consistent with the expected behavior. With lower SOAC broadcast interval, more agents are launched into the MANET which results in neighbor tables being updated more frequently.

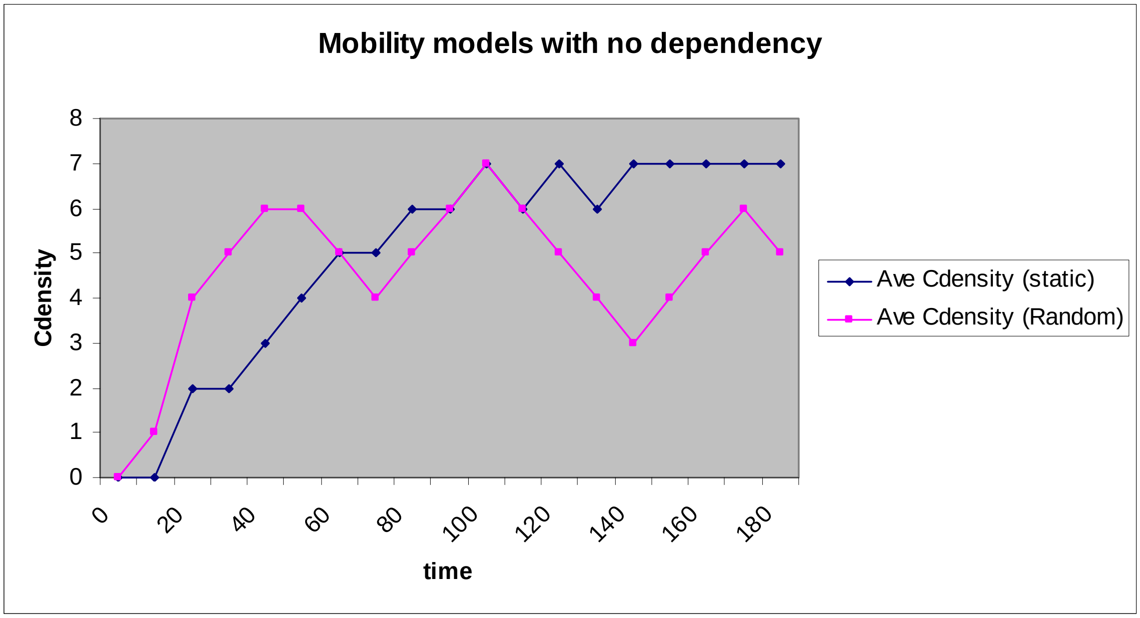

Table 6.4: The variation of $C_{density}$ with time for Random Mobility Models

The two border cases are considered here; static nodes with no movement and totally random node movement. In case of static nodes, the cluster formation starts a little late but is steady. In case of random node movement, all number of nodes in a cluster oscillates heavily around $C_{density}$ = ~5. All other cases of correlated mobility models are expected to fall between these two border cases.

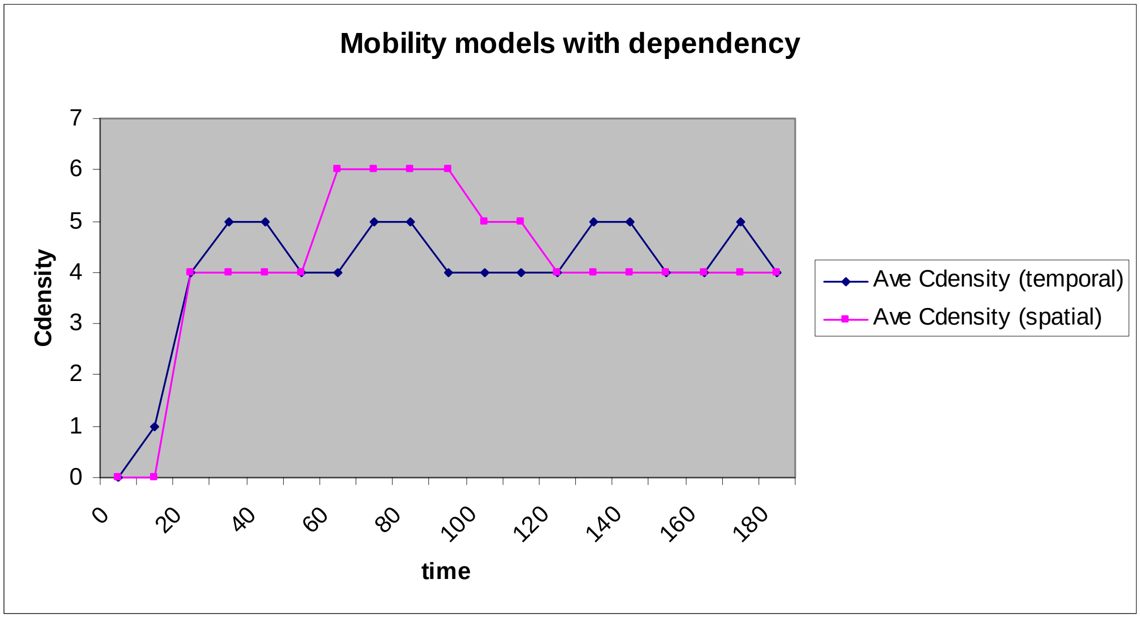

Table 6.5: The variation of $C_{density}$ with time for Pseudo-Random Models.

The behavior of correlated mobility falls between the borders set by the experiment enumerated in Table 6.6. The average, steady state $C_{density}$ was predicted to be within {4, 5}. The experiment result above validates the prediction

5.3 SOAC Overhead

The SOAC overhead to MANET

Interval	**% of nodes with automaton (ants)**	**% nodes in active connection**	**SOAC traffic overhead in MANET**
5 seconds	12%	20%	7%
10 seconds	3%	20%	2%

5.4 Network State Information

Amount of routing or network state information stored in nodes which are members of 1 cluster reduces to 1 unit. For nodes which are members to multiple clusters (say **m**), the amount of stored information reduces to **m** units. The memory requirements of the Cluster-head, for an **n** member cluster, increases depending upon the number of active connections maintained by each node and whether the cluster-head is participating as a forwarding engine with other cluster-heads. All this is expected in Role-based routing where the cluster-head selection guarantees that only nodes with sufficient resources are able to become cluster-heads.

5.4.1 Information Stored per Node

Statement: **The numbers of rows in Neighbor Table increase linearly as neighbors are added**

Let n be the number of neighbors and C(n) be the number of possible color combinations for the "n" neighbors,

For n = 2

R	B
R	R

C(n) = 2

For n = 3

R	B	G
R	R	G
R	R	R

C(n) = 3

For n = 4

Y	R	B	G
Y	R	R	G
Y	R	R	R
Y	Y	R	R

Y	Y	Y	Y

C(n) = 5

For n =5

P	Y	R	B	G
P	Y	R	R	G
P	Y	R	R	R
P	Y	Y	R	R
P	Y	Y	Y	Y
P	P	Y	Y	Y
P	P	P	P	P

We can see a pattern in the computation of C(n)
For n =6

O	P	Y	R	B	G
O	P	Y	R	R	G
O	P	Y	R	R	R
O	P	Y	Y	R	R
O	P	Y	Y	Y	Y
O	P	P	Y	Y	Y
O	P	P	P	P	P
O	O	P	P	P	P
O	O	O	P	P	P
O	O	P	P	R	R
O	O	O	O	O	O

For n = 5, 6, etc. we can formulize C(n) recursively in the following way:

C(n) = C(n-1) + X, where X is computed depending on the parity of n

If number (n) of nodes = ODD
All the combinations in n-1 with a new color added (in case of n =5, color P) add to the value of C(n) + the new combinations aroused due to addition of a new color (SHOWN IN BLUE FOR N=5, 6). The new combinations are as follows:
2 nodes of color M and (n-2) nodes of color N,
3 nodes of color M and (n-3) nodes of color N,
:
:
:
$\lfloor n/2 \rfloor$ nodes of color M and (n-$\lfloor n/2 \rfloor$) nodes of color N.
The pattern repeats from $\lfloor n/2 \rfloor$+1 to n.
The above combinations add $\lfloor n/2 \rfloor$-1 to the combination count. One more is added to the count due to all the n nodes of the same color(SHOWN IN GREEN FOR N=5, 6).

Thus X = $\lfloor n/2 \rfloor$ +1 -1 = $\lfloor n/2 \rfloor$
Thus for n = ODD,
C(n) = C(n-1) + $\lfloor n/2 \rfloor$

If number (n) of nodes = EVEN, the same logic holds. One more combination results in even nodes because of pairs of n/2 nodes with same colors (SHOWN IN PINK FOR N= 6).

Thus for n = EVEN, X = n/2 + 1
C(n) = C(n-1) +n/2 + 1

[Q.E.D.]

5.4.2 Fault Tolerance Characteristics

Fault tolerance of MANET is enhanced by multi-path routing. Border nodes which are within radio distance with more than one cluster-head can serve as forwarding nodes to form multi-path routing between two or more cluster-heads. Further research in this area is beyond the scope of this thesis and is an important added benefit of SOAC which can be exploited to improve fault tolerance characteristics of SOAC optimized MANET.

5.4.3 Comparison with other schemes

A paper dealing with clustering of MANET nodes is cited here [4]. This paper proposes a hybrid clustering algorithm by making the probability of path availability bounded. I am concerned about the computational overhead associated with this algorithm for clustering.

In this section I compare SOAC with another Swarm Intelligence based routing optimization scheme titled **Biologically Inspired Discrete Event Network**

Modeling by Hessam S. Sarjoughian. The biological inspiration is provided by Honey-Bee Scout-Recruit System.

5.4.3.1 Introduction to Honey Bee optimized routing algorithm

In their implementation, analogous to honeybee scout-recruit system, each network node is a beehive. Network corresponds to the world of honeybees which seek rich nectar sources, find paths with higher capacity to profitable nectar sources, find light-weight scout entities searching for nectar and control packets foraging for information to aid survival of the network (honeybee colonies). Each hive deploys a number of scouts to find the most profitable paths for a given destination. Each router then uses the information received from all the nodes in the network obtained by its scouts to calculate the shortest path to each destination in terms of a chosen metric. Scouts control congestion by making alterations to routing tables in order to route new traffic away from congested nodes. Then, packets are dispatched from a source to a destination according to information gathered by scouts. The cost metric can be based on the bandwidth of the link or can be dynamically measured as in the case of delay or load.

5.4.3.2 Comparison: SOAC and Honey Bee optimized routing algorithm

SOAC also operates on the same methodology of automaton distribution. The differences are:

- SOAC can be made to operate at a layer over or integrated into the routing layer; the honey-bee is implemented at the routing layer

- SOAC builds up a hierarchical networking structure over flat MANET routing plane; the honey-bee keeps the flat routing plane and tunes it
- SOAC is meant for MANET and honey-bee has no such preference

Chapter 6

Simulation Constraints and Ways to Improve Turn-around Time

6.1 Constraints of NS2 simulation framework

The limitations of the NS2 simulation framework are due to its inherent complexity, preprocessing load, scalability issues, large memory requirements, not being dynamic in nature and not distributed.

The NS2 simulation environment is complex. A substantial amount of time is required to understand the internal working and code-base before you can use it. Competency is required in TCL/OTCL to start even the basic simulation. OTCL provides a simplified interface but increases simulation turn-around time for large simulation initiatives. Use of C++ interface is the fastest way to build new features into NS2. The NS2::MANET framework requires generation of mobility files whose size and generation time increases exponentially with increase in number of nodes. This also implies that the time needed to load the movement file into computer memory increases exponentially and so does the simulation completion time.

NS-2 is too slow with big scenarios. Experience with simulation time-around time both from this thesis and those experienced by others using NS2 are graphically compiled in Appendix B. NS-2 has too much runtime memory requirements and has very conservative output file size limitations. Once the NS2 simulation is loaded and simulation is in progress, there is no built-in method to dynamically configure it. There is no support for distributed simulation which reduces scalability of NS2 simulation framework as number of MANET node increases.

6.2 Integrating NS2 simulation framework with DEVSJAVA

DEVSJAVA is a generic system simulation tool based on Discrete Event System Specification which is a well defined mathematical formalism. NS2 on the other hand is a discrete event scheduler built specifically for network simulation. These two simulators have their own event scheduling methods and time synchronization becomes challenging in integration efforts and are addressed in work done by Kim et al [5].

Given the flexibility of DEVSJAVA and its built-in support for distributed simulation, NS2 integrated with DEVSJAVA may help us reduce simulation turn-around time by distributing computation across various workstations.

Chapter 7

CONCLUSION AND FUTURE WORK

The SOAC optimization of MANET is interesting and shows substantial improvement in MANET good-put on in the case of correlated MANET mobility models and low speeds. In a metropolitan area with predictable node movement and correlation, these pseudo-random models are nearer to real-life scenarios than purely random models. It is believed that these models do not capture the real picture of node mobility and trace-based models are most appropriate. Till MANET sees substantial deployment, the possibility of trace-based models is scarce. Till then pseudo-random mobility models with temporal, spatial and geographic correlation will serve to feed MANET simulation environments.

Much of future work in this field involves proving the benefits associated SOAC like fault tolerance, multi-path routing, power saving etc. SOAC algorithm proposed in this thesis is very generic and thus there is always room to improve it further depending upon the situation in hand. But while trying to adapt it to various scenarios, the core property of collective intelligence should be kept intact. Future research work needs to be conducted to compare the good-put of SOAC optimized MANET with other adaptive clustering algorithms [4] and similar swarm based clustering [8].

Substantial work needs to be done to improve turn-around time of simulation for high node count. An attempt to integrate NS2 with a generic distributed simulation framework like DEVSJAVA will help in that effort.

REFERENCES

[1] Dorigo, M., Maniezzo, V., & Colorni, A. (1996). The Ant System: Optimization by a colony of cooperating agents. IEEE Transactions on Systems, Man, and Cybernetics-PartB, 26 (1), 29-41.

[2] Gianni Di Caro and Marco Dorigo (1999), Mobile Agents for Adaptive Routing IRIDIA, Universite Libre de Bruxelles

[3] Fan Bai and Ahmed Helmy (1999), A Survey of Mobility Models in Wireless Adhoc Networks, University of Southern California,U.S.A

[4] A Mobility Based Framework for Adaptive Clustering in Wireless Ad-Hoc Networks

A. Bruce McDonald and Taieb Znati, IEEE Journal on Selected Areas in Communications, Vol. 17, No. 8, August 1999

[5] DEVS-NS2 ENVIRONMENT; An integrated tool for efficient networks modeling and simulation

ECE, University of AZ, M.S. Thesis, May 2006

[6] Swarm Intelligence: From Natural to Artificial Systems (Santa Fe Institute Studies on the Sciences of Complexity)

[7] Internet Resource: Ant Colony Optimization and Swarm Intelligence: 4th International Workshop, ANTS 2004, Brussels, Belgium, September 5-8, 2004, Proceeding (Lecture Notes in Computer Science)

[8] Biologically Inspired Discrete Event Network Modeling by Hessam S. Sarjoughian. The biological inspiration is provided by Honey-Bee Scout-Recruit System.

[9] Internet Resource: Wikipedia.org

APPENDIX A

5 Colors:

Grouping	Unique neighbors	$f(m_i)$	Kd	P_d	Kr	P_r	P_u	Notes
{R, G, B, Y, P}	5	0.00	0.50	1.00	0.10	0.00	0.00	
{R, R, G, B, P}	4	0.10	0.50	0.69	0.10	0.25	0.06	
{R, R, G, G, P}	3	0.25	0.50	0.44	0.10	0.51	0.05	
{R, R, R, G, G}	2	0.50	0.50	0.25	0.10	0.69	0.06	
{R, R, R, R, R}	1	1.00	0.50	0.11	0.10	0.83	0.06	
								Kr of 0.1 makes Pr > 0.8 when all the nodes are of the same color. This value of Kr was found through trial and error. Dependence of Pr on Kr is found to be independent of number of neighbors. Similarly Kd was set to 0.5 by constraint that Pd(mi) > 0.5 when at least (N-3) i.e RED ROW - neighbors are different, where N > 4

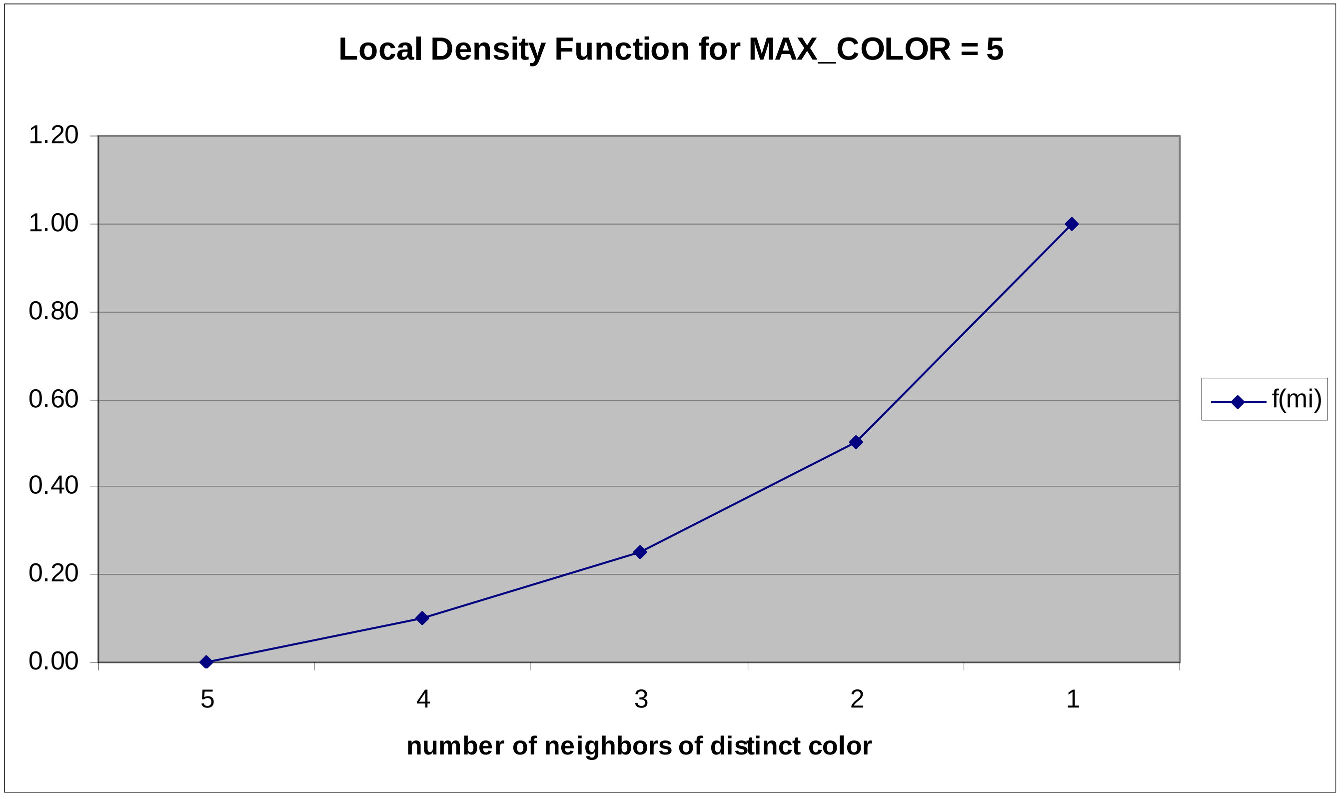

Figure A.x: Drawn as per table data in Appendix

APPENDIX B

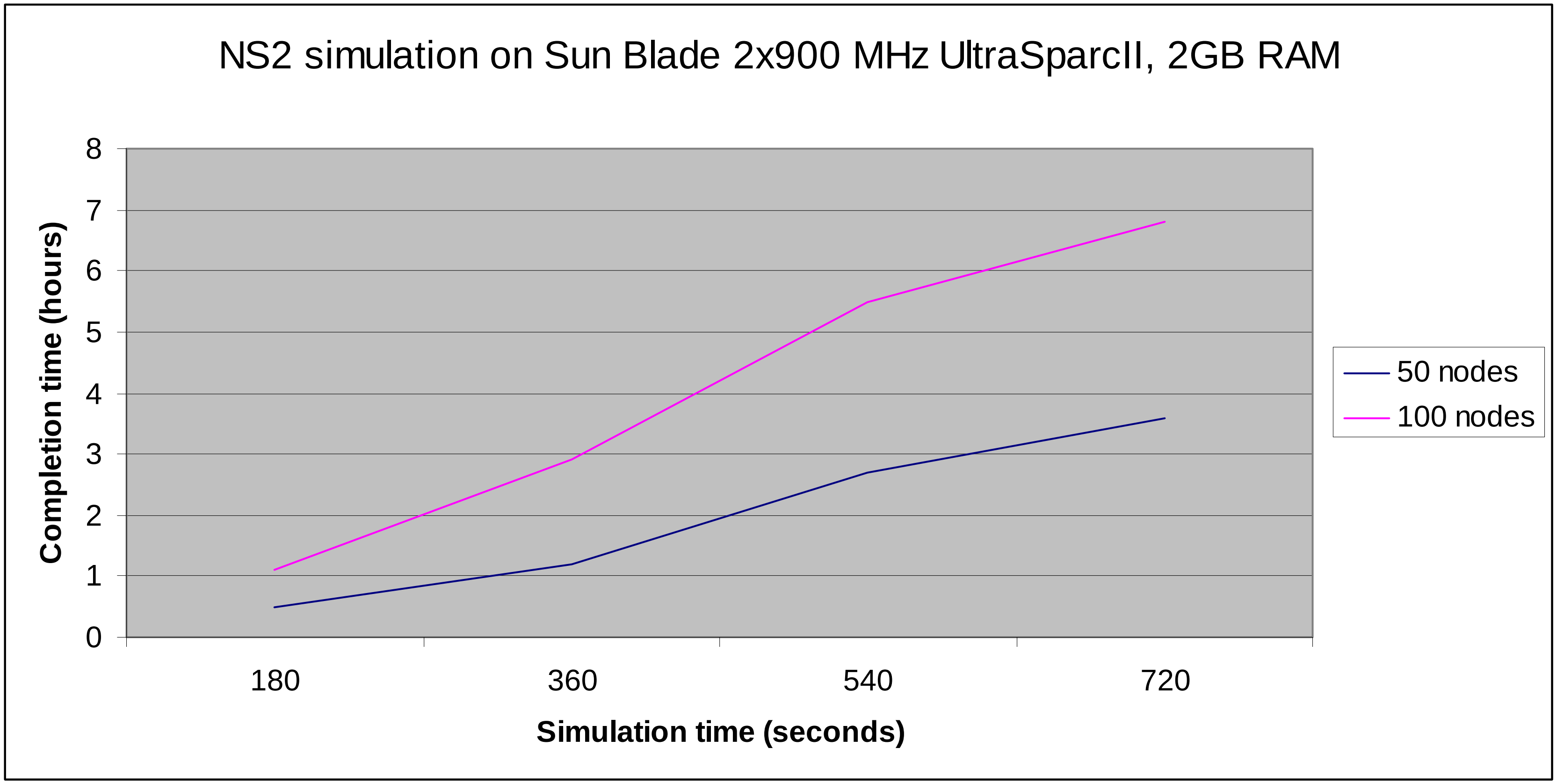

Figure above shows variation of NS2 completion time against actual simulation time for different node count

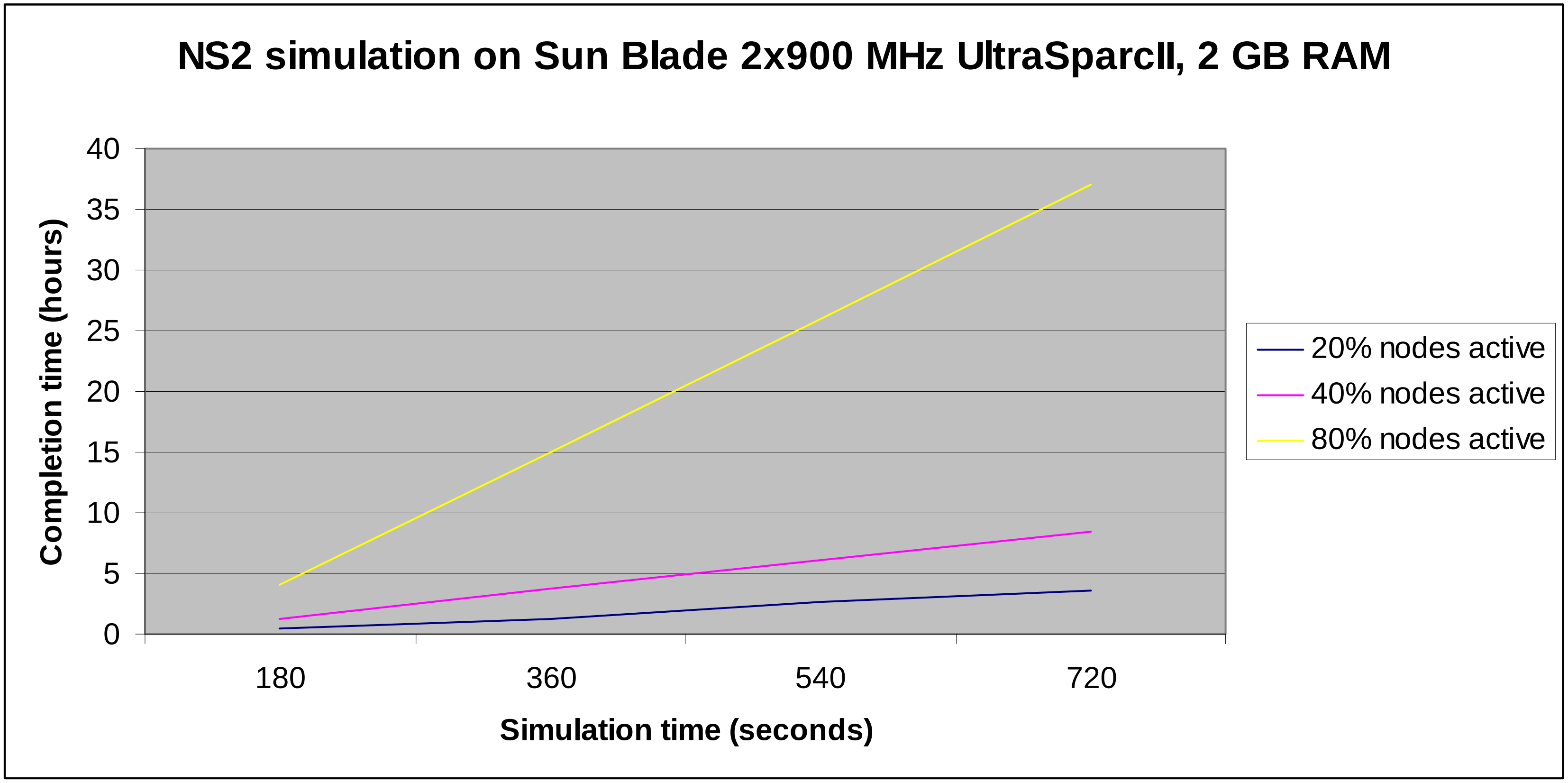

Figure above shows variation of NS2 completion time against actual simulation time for different traffic intensity

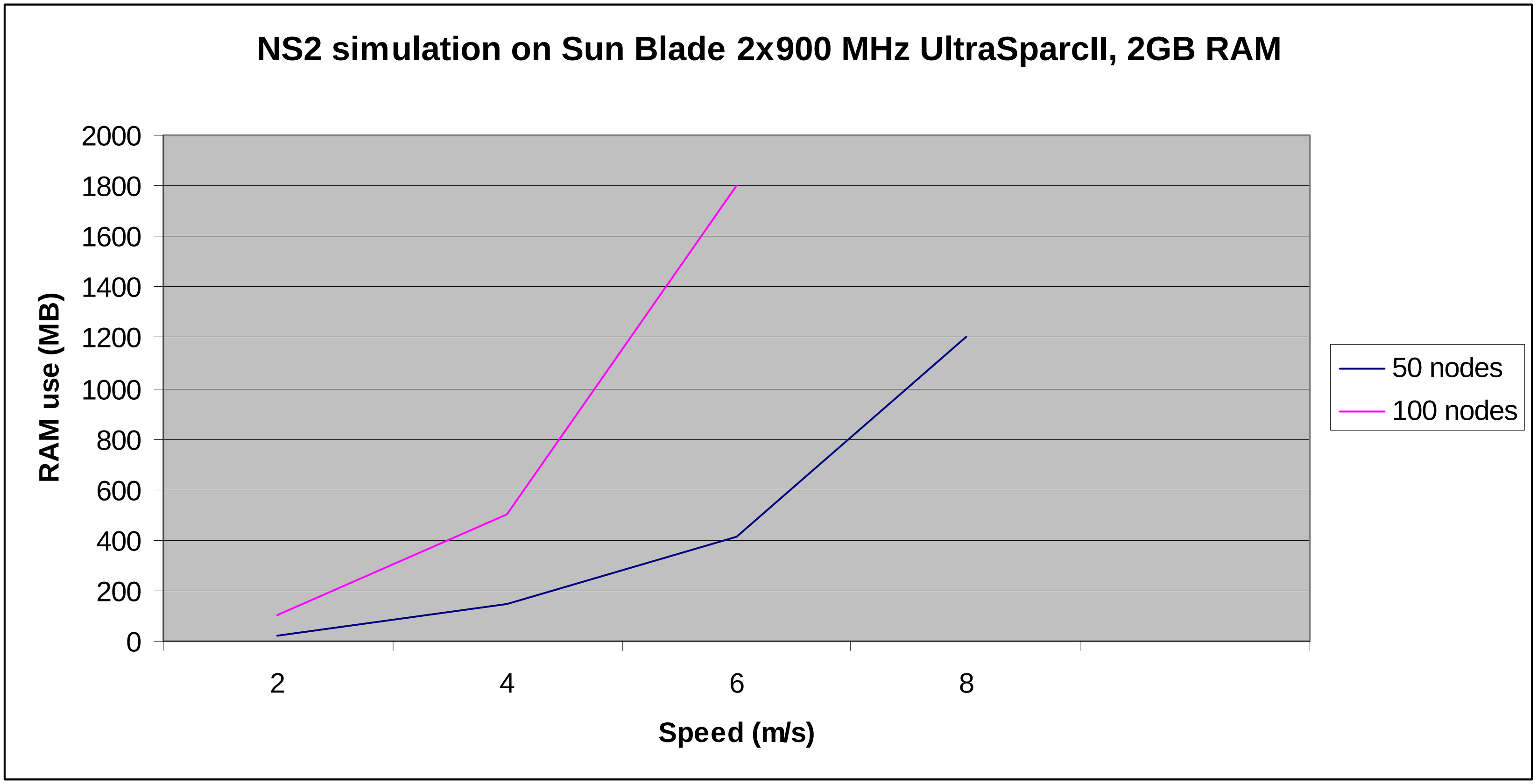

Figure above shows variation of NS2 memory usage against average node speed for different node count