A Framework for Evolutionary Networking

Gautam Kumar Gupta and S V Raghavan
Network Systems Laboratory
Department of Computer Science and Engineering
Indian Institute of Technology Madras
Chennai, INDIA 600 036

{gautam, svr}@cs.iitm.ernet.in

ABSTRACT

In Computer-Communication Networks, addressing and routing have been fundamental issues that have challenged researchers – resulting in myriads of addressing and routing protocols. In recent times, self-configuration of nodes has become a necessity due to large number of networked devices and pervasive use of networks. Emergence of autonomic networks based on wireless mesh or ad hoc approach underline the need for self-configuration. Besides, success in sensor technology resulting in proliferation of wireless sensor networks is rapidly pushing the frontiers of self-configuration in large scale. The solutions reported hitherto in literature, has an interesting underlying similarity – *Addressing, Routing and Mobility (A.R.M.) issues have been tackled separately.*

In this work, we propose *Protocol for Evolutionary Addressing* (PEA) Framework, pronounced as "P", which solves the problem of addressing and routing in unison - thereby eliminating the need for separate routing algorithm. In PEA Framework, nodes assume addresses and self-configure the forwarding tables to reflect the changes in the network topology. Besides, PEA Framework enables self-configuration of nodes in a network so that the network naturally evolves (or readjusts) as it grows (or changes).

We describe the framework, protocol, and evolution of network in addition to analyzing the time and message complexity of the protocol.

Categories and Subject Descriptors

C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design – *Distributed Networks, Network Communications, Network Topology, Wireless Communication.*

General Terms

Design, Performance.

Keywords

Self-Configuration, Next-Gen Network Architecture, Addressing.

1. INTRODUCTION

In networks, the emergence of new opportunities in speed of transmission, choice of technology, dynamism in topology and proliferation of devices, brings in new challenges. Perhaps, the need of the hour is to encourage a paradigm shift in the way we perceive the "address of a node" – from static to dynamic – that too dynamic and transient – we are in essence moving from the era of Assigning[1] or Acquiring[2] an address to *an era of Assuming an address.*

In our work, we argue that the nodes in a network can assume an *address with an associated context*. When the context changes, the address also changes. Such an addressing scheme has three inherent benefits as given below:

- The first and foremost benefit is the *relative ease of associating addresses to nodes*. By definition, the collision domain of addresses is localized. This results in scalable address management.

- The second benefit is the *ease in routing of packets in a network*. As the address of a node contains the Point of Attachment by definition, packets can be routed by simply interpreting the address, obviating the need for an explicit routing algorithm. We call this concept as Address Guided Forwarding (AGF).

- The third benefit is the *self-organization in the context of mobility*. When nodes move, they leave their previous logical domain and join new ones. In the process, they assume new addresses and update the corresponding forwarding tables.

All the three benefits mentioned above, traditionally required development of separate protocols. Departing from the tradition, we propose an integrated framework of definitions, axioms, and protocol to handle the *A.R.M. issues* in a consistent manner. Besides, inspired by the *simplicity*, *robustness* and *scalability* of *natural systems* (which are evolutionary in nature), we prove that the networks produced using the PEA Framework exhibit the characteristics of natural systems.

---

[1] Assigning refers to static IP addresses assigned to nodes in IP subnets.

[2] Acquiring refers to dynamic IP addresses acquired by nodes in DHCP environment.

2. PEA FRAMEWORK

A framework is a system (or an extensible structure) of describing concepts, principles or methods that is used to build something [1]. PEA Framework comprises of a *conceptual component* (Definitions, Axioms, Information and Properties) and an *operational component* (PEA Protocol). The conceptual and operational components of PEA Framework are the guiding principles for self-configuration of nodes. In the PEA Framework, networks evolve based on a Policy of Evolution. Conceptual component of PEA is *independent* of the Policy of Evolution and the operational component, i.e. the PEA Protocol, is dependent on the Policy of Evolution.

PEA Definitions

Node – Node is any communication device that is used to form a network. We call it a PEA node.

Address – A PEA node has two types of addresses - Local and Global.

Cluster – A group of nodes that are in the vicinity of each other in terms of signal strength.

Local Address (LA) – Local Address of a node is unique in its cluster.

Global Address (GA) – Global Address of a node is unique in whole network.

Family – A family is a collection of an L_1 cluster with all of its descendant clusters, as illustrated in Figure 1. Each family has a unique *family-id*.

Peer-to-Peer Join – Within a cluster, two nodes are connected through Peer-to-Peer (P-2-P) join.

Parent-to-Child Join – Two nodes at consecutive levels are connected through Parent-to-Child (P-2-C) join.

Cluster-to-Cluster Join – Two nodes at non-consecutive levels are connected through Cluster-to-Cluster (C-2-C) join.

Cluster Head Node (CHN) – A node is said to be Cluster Head Node if it forms a P-2-C join with its upper level node. In Figure 2, gray color nodes that have LA 1 and 2 are CHNs.

Gateway Node (GN) – A node is said to be Gateway Node if it forms a C-2-C join across families or networks. In Figure 2, checked nodes are GNs.
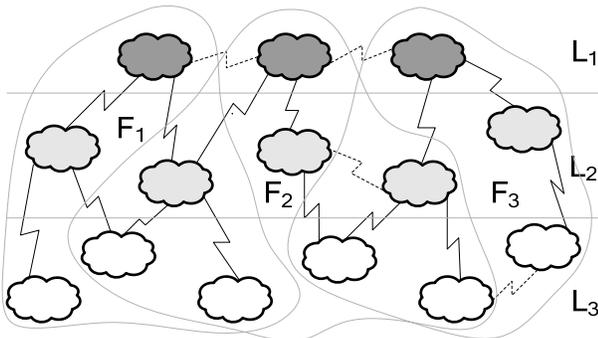
Figure 1. Fragment of a PEA network having 3 families, F_1, F_2, F_3 and 3 levels, L_1, L_2, L_3

To manage addressing and routing in any network, we need to impose some structure on the network of nodes. In system design, hierarchy is often used to organize entities in a system, as it can deal with complexity and scalability [2]. By the same token, hierarchy becomes the natural choice for the organization of nodes in a network. In PEA Framework, hierarchy is the fundamental structure and PEA networks have <u>different levels</u> of *clusters of nodes* viz. L_1, L_2, L_3 and so on, with L_1 as the root. Fragment of such a PEA network with three levels is shown in Figure 1.

PEA Axioms

1. A node can have only one Local Address but it can have more than one Global Addresses.

2. Within a cluster, all nodes are at the same hierarchical level.

3. At a time, there can be only one type of join between two nodes.

4. At a time, a node can be in one cluster only.

2.1 PEA Node and its Associated Information

In PEA Framework, each node has following information that is needed for the operation of the PEA protocol.

Neighborhood Information

a. Peer Address Set (PAS): It is set of Local Addresses (LAs) of all the nodes in the same cluster.

b. Cluster Head Address Set (CHAS): It is set of LAs of Cluster Head Nodes in the same cluster.

$$CHAS \subseteq PAS$$

c. Gateway Address Set (GAS): It is set of the LAs of all GNs with which a node establishes C-2-C join.

d. Child Address Set (CAS): It is set of LAs of all the child nodes of a node. It will be non empty only if the node is a parent node with one or more child nodes.

Hierarchy Information

e. Hierarchical level (L): It is the level of the node in the PEA network hierarchy.

f. Family Set (FS): It is set of all the *family-ids* of the families to which a node belongs. A node can be multi-homed to more than one family.

Packet Routing Information

g. Global Address Prefix Set (GAPS): It is set of prefixes of all the GAs of a node such that appending LA or ϕ (Null) to the prefix results in one of the GAs of the node.

h. Intra-Cluster Forwarding Table (ICFT): It is set of ordered pairs of LAs. For example ICFT of a node x is as follows.

$$ICFT\ (x) = \{(4, 1), (1, 1), (3, 3), (5, 3), (x, *)\}$$

In the ordered pair, the first element is LA of destination and the second element is LA of next hop that is used to reach the destination. The element $(x, *)$ signifies that x is the LA of the node itself. We can establish relation between ICFT

and PAS as follows.

$$ICFT \subset (PAS \times PAS) \cup \{(x, *)\}$$

i. Inter-Family Forwarding Table (IFFT): It is set of ordered pairs of *family-id* and LA. Example of IFFT of an L_1 node is as follows.

$$IFFT = \{(1, 5), (2, 4), (4, 7), (3, *)\}$$

In the ordered pair, the first element is *family-id* of the destination family and the second element is the LA of GN that is used to reach the destination family. The element (3, *) signifies that 3 is the *family-id* of the node.
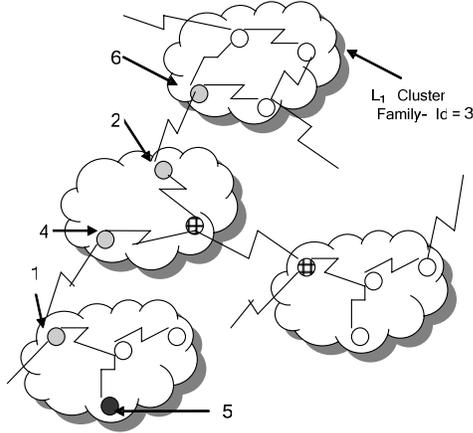
Figure 2. Illustration of CHN, GN and GA of a node

For an isolated node, the PAS, CHAS, GAS, CAS, FS, GAPS, ICFT and IFFT sets will be NULL. The value of L will be 0. As a node joins a network, there are appropriately updated by the PEA protocol.

2.2 Properties of PEA Nodes

The definitions, axioms and PEA node information described hitherto give rise to the following properties:

P1. Family Set of an L_1 node is always a singleton set.

P2. For L_1 nodes, Cluster Head Address Set (CHAS) is always an empty set.

P3. Inter-Family Forwarding Table (IFFT) of non L_1 nodes is always an empty set.

2.3 Anatomy of Global Address (GA)

The GA of a node consists of *family-id*, *LAs of intermediary P-2-C Join nodes*, and the *LA of the node* itself. For example, the GA of the node shown in black color in Figure 2 is 3-62-41-5. Here 3 is the *family-id*, 5 is the LA, and the middle part - 6, 2, 4 and 1 - is the LAs of the intermediate P-2-C join nodes (shown in gray color in Figure 3). These intermediate nodes – 6, 2, 4 and 1 – come into the picture while traversing path from L_1 cluster to the node under discussion. Since there can be more than one path from L_1 cluster to the node, it can potentially have more than one GA. Besides, forwarding of packets is guided by the GA of the destination. This results in a novel concept of *Address Guided Forwarding* (AGF) [3].

If c is the number of bits in a LA, f is the number of bits in *family-id*, l is the number of bits in hierarchical level, L, then the minimum and maximum length of *Global Address (GA)* of a node will be $f + c$ and $f + 2 * c * (2^l - 1) + c$ respectively.

3. PEA PROTOCOL

PEA protocol is designed for self-configuration of nodes' address and the corresponding forwarding tables. PEA protocol consists of three sub-protocols called JDP (Join Decision Protocol), JEP (Join Establishment Protocol), and JTP (Join Termination Protocol). JEP and JTP handle join establishment and termination between two nodes. PEA has three types of joins: P-2-P, P-2-C and C-2-C. Using the Policy of Evolution, JDP decides the type of join to be established. One can visualize the role of the three joins as follows: P-2-P join fills a cluster, P-2-C join forms a hierarchy of clusters, and C-2-C join fuses all the hierarchies forming the PEA network.

When a new node enters the PEA network, JDP decides the type of join to be established. JEP establishes this join, thus initializing the address and the forwarding tables of the node. When the node leaves, JTP terminates the existing join, and updates the forwarding tables of the remaining nodes. *Mobility* of a node is handled as *JTP followed by JDP and then JEP*, as the existing join has to be terminated before the establishment of a new join. In summary, new nodes are handled by JDP and JEP and mobility is handled by JTP, JDP, and JEP in a cycle.

Since a node's GA changes due to mobility, to avoid (or minimize) disruptions at upper (transport) layer, mechanisms similar to Extended TCP [4], MSOCKS [5], SCTP [6] etc. can be used. After assuming the new address, the node informs its peer communicating nodes and their cluster head nodes about the new address so that the communication continues without disruption. Upper bound on the time a node takes to assume a new address is presented in section 4.

Network Evolution using HELLO Messages: PEA network evolves using *HELLO messages* broadcast by nodes as beacons. If a node is already in the network, then the beacon signifies that the node is alive and if the node is "new", beacon signifies a join. The nodes then exchange messages based on Challenge – Response (C-R) protocol and JDP then takes over to decide the type of join. C-R based approach prevents circular transmission resulting in "infinite" join.

Illustration of PEA Protocol Operation: Evolution of a PEA network begins when two isolated PEA nodes come in the vicinity of each other and forms an L_1 cluster. Using P-2-P joins the cluster fills up to the maximum cluster size. Once the cluster is full, next level of clusters and other L_1 clusters are formed using P-2-C and C-2-C joins respectively. The order in which P-2-C and C-2-C joins are established is decided by policy of evolution. Thus, hierarchy of clusters and fusion of these hierarchies occur as per policy of evolution. In Figure 3, a fragment of an already evolved PEA network is shown. Here, we illustrate the instances of the three joins: P-2-P, P-2-C and C-2-C.

In order to appreciate the operational simplicity of PEA protocol, one should understand the three join operations and the context in which they happen. It is important to note that when a P-2-P join is used (between two nodes in a cluster), the information associated with all the nodes in the cluster is updated.
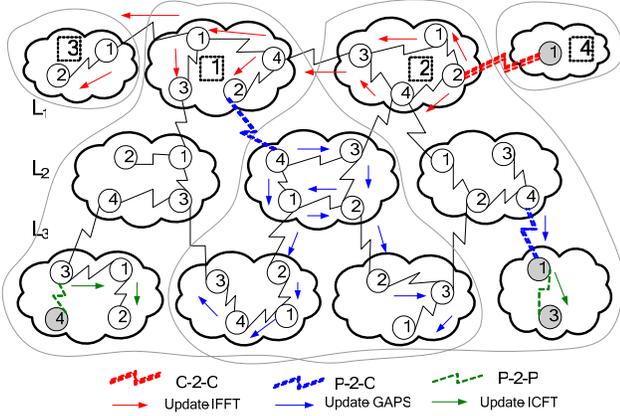
Figure 3. Illustration of PEA Protocol working

Similarly, when a P-2-C join is used, the information associated with all the nodes that are descendant to P-2-C join is updated. Extending the same logic, when a C-2-C join is used, the information associated with all the L_1 nodes is updated. These three operations are illustrated in the sequel using appropriate annotations in Figure 3.

Table 1. Impact Matrix of PEA Protocol

Info. ---------- Action	P A S	C H A S	C A S	G A S	L	F S	*IC FT*	*GA PS*	*IF FT*
P-2-P	✓				✓	✓	✓		
P-2-C		✓	✓		✓	✓		✓	
C-2-C				✓	✓	✓			✓

Table 1 summarizes the PEA node information that is updated when a P-2-P, P-2-C or C-2-C join occurs. The updates in the neighborhood and hierarchy information is minor (used for book keeping purpose only) and they occur only in the *nodes directly involved in the join*. Therefore, we focus on routing information (shown in gray color) of PEA nodes while illustrating the three types of join updates.

A P-2-P join (shown in Figure 3 as single zigzag line) is updated between node 4 and 3 at L_3. Before and after the establishment of the P-2-P join, the ICFT of cluster nodes is shown in Table 2. The updated entries are shown in bold face.

Table 2. Illustration of ICFT updates in the cluster

Node	ICFT (Before Join)	ICFT (After Join)
1	(1, *), (2, 2), (3, 3)	(1, *), (2, 2), (3, 3), **(4, 3)**
2	(1, 1), (2, *), (3, 1)	(1, 1), (2, *), (3, 1), **(4, 1)**
3	(1, 1), (2, 1), (3, *)	(1, 1), (2, 1), (3, *), **(4, 4)**
4	---	**(1, 3), (2, 3), (3, 3), (4, *)**

A P-2-C join (shown in Figure 3 as double zigzag line) is updated between node 2 of L_1 and node 4 of L_2. Before and after the establishment of P-2-C join, the GAPS of nodes of L_3 cluster (the one with 4 nodes and blue color arrows) is as follows. The updated entry is shown in bold face.

GAPS (Before Join) = {2-43-12, 1-31-33}
GAPS (After Join) = {2-43-12, 1-31-33, **1-24-12**}

A C-2-C join (shown in Figure 3 as triple zigzag line) between node 1 and 2 is updated. Before and after the establishment of the C-2-C join, the IFFT of L_1 nodes (with *family-id* 1) is shown in Table 3. The updated entry is shown in bold face.

Table 3. Illustration of IFFT updates in L_1 clusters

Node	IFFT (Before Join)	IFFT (After Join)
1	(1, *), (2, 4), (3, 1*)	(1, *), (2, 4), (3, 1*), **(4, 4)**
2	(1, *), (2, 4), (3, 1)	(1, *), (2, 4), (3, 1), **(4, 4)**
3	(1, *), (2, 4), (3, 1)	(1, *), (2, 4), (3, 1), **(4, 4)**
4	(1, *), (2, 3*), (3, 1)	(1, *), (2, 3*), (3, 1), **(4, 3*)**

JDP is based on a policy of evolution to decide the *type of join* to be established. In fact, the policy of evolution decides the structural and topological properties of PEA networks. It is therefore interesting to know the impact of the policy of evolution on the operational efficiency of the PEA protocol.

4. ANALYSIS OF PEA PROTOCOL

In system design, from control theoretic perspective, Observability and Controllability are two important aspects to be considered during analysis. In the work reported in this paper, *we concentrate mainly on the Observability of PEA Framework*.

An Approach to Analyze PEA Evolution: Since PEA protocol pre-supposes the existence of a policy of evolution, we begin the analysis of PEA protocol by considering policy of evolution.

For different policies of evolution, different PEA networks evolve. Basically, a policy of evolution governs the size of core (L_1 nodes) that has implications on time and message complexity of PEA protocol operation. One such policy of evolution is represented in the following relation.

$$L_k(1) \rightarrow \begin{cases} L_k(c/2) + L_{k+1}(c/2) & \text{if } k = 1 \\ L_{k+1}(c) & \text{if } k \neq 1 \end{cases}$$

Here c = maximum size of cluster and $L_k(x)$ = x nodes at level k. Essentially, an L_1 node gives rise to c/2 nodes of L_1 level and c/2 nodes of L_2 level. But a non L_1 node gives rise to c nodes of next level. We call this policy of evolution as $L_2L_1…$ because at L_1 level, L_2 and L_1 levels evolve alternately.

Intuitively speaking, use of such a policy of evolution ensures that there will be (always) enough number of L_1 clusters when a network evolves. It is necessary to have *enough* L_1 clusters as their role in PEA network operation is critical (explained in subsequent analysis).

In our analysis, we explore the evolutionary pattern in terms of rounds of clusters. In first round, there is only one L_1 cluster. As nodes come from all the directions, other rounds of clusters are formed. After filling the kth round, k+1th round starts. We call such a formation of rounds as the Ripple Round Model of PEA evolution. If we want to get regular distribution of load on the core of network, we need to control the evolution and foresee how far L_1 nodes grow in PEA network evolution. Using a policy of evolution such as $L_2L_1…$, one can evaluate the maximum number of rounds in which the core (L_1 nodes) of a PEA network is present.

(a) Small Core   (b) Medium Core   (c) Large Core
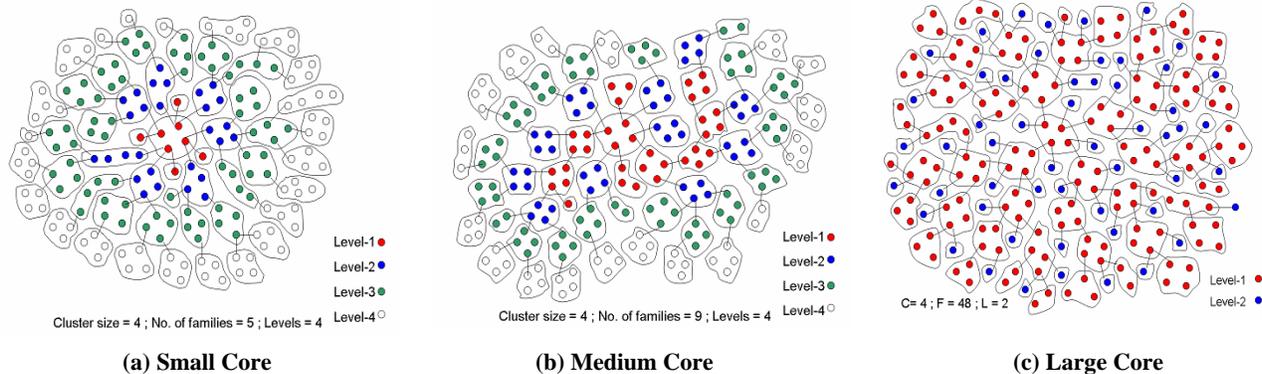
Figure 4. Three PEA networks evolved using Policy of Evolution $L_2L_1…$

For a given number of L_1 clusters (N_f) and given cluster size (c), the extent, up to which the core of network is present, can be given as follows:

$$N_f = 1 + (c^2/2)((q^k - 1)/(q - 1)) \text{ where } q = (c^2/2) - 1$$

This information can be used to estimate the upper bound on forwarding latency.

The four parameters that control the performance of PEA networks are the *arrival rate of nodes*, the *arrival direction of nodes,* the *cluster size* and the *policy of evolution.* Arrival rate of nodes governs the rate at which PEA network evolves. If the nodes arrive at increased rate, the PEA *network will evolve quicker* than otherwise. Cluster size governs the rate at which next levels of clusters are formed. As cluster size increases, the rate at which *newer levels are formed slows down*. But the trend in time and message complexity does not change. Therefore, in our simulation experiments, we concentrate on the *policy of evolution* in conjunction with *arrival direction of nodes.*

For the $L_2L_1…$ policy of evolution, with controlled arrival direction, PEA network evolution is shown for three different scenarios in Figure 4. Here, evolution of PEA networks can be visualized as blooming flower starting from first L_1 cluster. Basically these three scenarios of evolution differ in the size of core (L_1 nodes). For small core, arrival direction is chosen such that new nodes go to fill non L_1 clusters so that core evolution is restricted. For large core, arrival direction is chosen such that new nodes always fill L_1 clusters so that the core evolves at a rapid rate. But if we release the constraint of controlled arrival direction and allow nodes to come from all directions uniformly, the medium core PEA network is evolved. In fact, we observed in all our simulation experiments that the medium core PEA network is the most probable in PEA evolution. It should be noted that the evolution of these three scenarios of PEA networks is done manually so that the arrival direction can be controlled to contain the core at desired size.

We consider these three scenarios of evolution because large and small core PEA networks are two extreme cases of PEA evolution (even though they are extremely rare) while medium core PEA network is a realistic case of PEA evolution. In the sequel, we report on simulation experiments from the performance point of view.

Comparison of the three scenarios of PEA Evolution: We

compare the three scenarios of PEA evolution – large, medium and small core - with respect to the following two criteria: Join Establishment Time (JET) and Message overhead.

In a PEA network, the amount of time a node takes to establish a join is called Join Establishment Time (JET). In case of node mobility, JET is equivalent to handover latency. Establishment of a join consists of associating an address to the node, initializing the forwarding tables of the node and updating the information of other corresponding nodes.
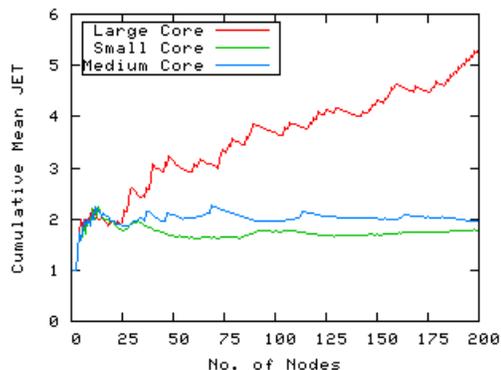
Figure 5. – Cumulative Mean JET vs. Nodes

Figure 5 shows how cumulative mean JET varies as the network evolves in the three scenarios. From the point of view of JET, large core network is the worst while small core network is the best. It happens because in large core most of the nodes are L_1 nodes and in small core L_1 nodes are very few. As explained in section 3, when L_1 nodes establish C-2-C joins, it results in update of all L_1 nodes. In other words, a C-2-C join takes more time to establish than other types of join. Therefore, as core size (L_1 nodes) increases, JET also increases. *This disfavors large core PEA evolution.*

When a join is established, a number of update messages are transmitted to disseminate information about the updated join. This Message overhead is shown for three different scenarios of PEA evolution in Figure 6. Again, from the point of view of message overhead, large core network is the worst while small core network is the best. The reason is that in large core most of the nodes are L_1 nodes and establish C-2-C joins. And as explained earlier in section 3, each C-2-C join entails update of all the L_1 nodes. Therefore, as the core size (L_1 nodes) increases,

message overhead also increases. *Again, this disfavors large core PEA evolution.*

$L_2L_2L_2L_1$…and $L_2L_2L_1$…, number of L_1 nodes are lesser and therefore JET and message overhead is also lesser.
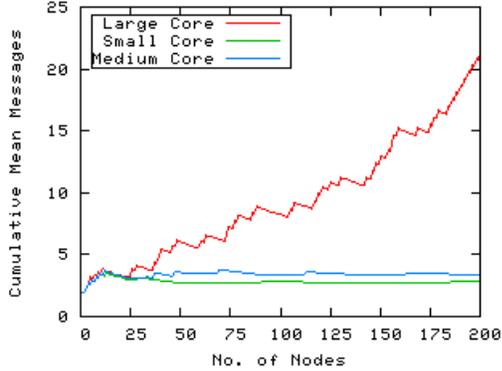
Figure 6. – Cumulative Mean Messages vs. Nodes

Figure 7. – Cumulative Mean JET vs. Nodes

From JET and Message overhead analysis, one can safely *exclude large core* PEA networks. And one may conclude that PEA network with small or medium core is desirable. But there is a tradeoff between time-message complexity and other performance criteria. Small core signifies lesser no. of L_1 clusters which in turn implies lesser no. of GAs per node. This results in lesser no. of paths between any two nodes and *overloads* the small no. of L_1 nodes. Besides, small core PEA network is more prone to disconnectivity due to scarce L_1 clusters. These results reinforce our intuition that *medium core PEA network* has desirable tradeoff between time-message complexity and other performance criteria such as no. of paths etc. Medium core PEA network that is natural and realistic, and perhaps the best scenario in PEA network evolution, from performance point of view.

So far, we have seen that a given policy of evolution with controlled arrival direction results in PEA networks of varying core sizes and the resulting medium core PEA network is the most desirable and realistic scenario of PEA evolution. In the following section, we analyze performance of PEA networks evolved using various policies of evolution and *identify the ideal policy of evolution.* While analyzing different policies of evolution, it is worth noticing that the nodes come from all the directions with uniform distribution.

Performance of different Policies of Evolution: In PEA Framework, role of the policy of evolution is to determine the core size for the evolved network. Instead of L_2L_1… as policy of evolution, one can have some other policy of evolution such as $L_2L_2L_1$… which means that L_1 nodes form two L_2 clusters then one L_1 cluster alternately and like wise. We take following five policies of evolution in consideration: $L_2L_2L_2L_1$…, $L_2L_2L_1$…, L_2L_1…, $L_2L_1L_1$…, and $L_2L_1L_1L_1$…To compare these five policies, again we use the same two criteria: JET and Message overhead. The results reported in this section are obtained through simulation using MATLAB [7] with uniform arrival direction and cluster size 4. Due to uniform arrival direction, PEA networks with medium core evolve.

From Figure 7 and 8, one can observe, as core size (L_1 nodes) increases due to policy of evolution, JET and message overhead also increases. It happens for $L_2L_1L_1$… and $L_2L_1L_1L_1$… because no. of L_1 nodes and therefore C-2-C joins increases which are costly in terms of JET and message overhead. On the contrary, for
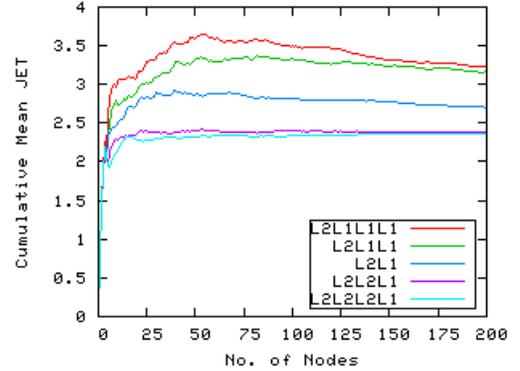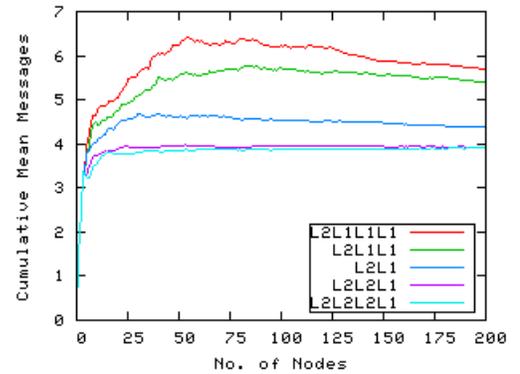
Figure 8. – Cumulative Mean Messages vs. Nodes

The next logical step is to identify (construct) the ideal policy of evolution? From the above analysis, one can conclude that policy of evolution has direct impact on performance of evolved network. One should choose a policy of evolution such that the core size is neither too large nor too small. Considering the tradeoff between time-message complexity and other performance criteria such as no. of paths etc., our choice of the policy of evolution L_2L_1… satisfies this criterion because it produces L_1 and non L_1 nodes alternately.

4.1 Asymptotic Analysis

To understand the upper and lower bounds on time and message complexity of PEA protocol, we perform the asymptotic analysis of the PEA protocol. It is worth noticing that asymptotic analysis is independent of policy of evolution. Since PEA protocol consists of three sub-protocols, we investigate each of the sub-protocol for analysis.

4.1.1 Time Complexity

The Time Complexity of an algorithm is a measure of time it takes in execution of algorithm. Here, we compute time taken by PEA Protocol (i.e. JDP, JEP and JTP). Time complexity of JDP is always $\Theta(1)$ because in each case, JDP decides the type of join in constant time.

The time complexity of JEP and JTP depends on Update-ICFT( ), Update-GAPS( ) and Update-IFFT( ) procedures that are invoked due to update in P-2-P, P-2-C and C-2-C joins respectively. Since

Update-ICFT messages are transmitted in one cluster only as shown in Figure 3, the time complexity of Update-ICFT( ) procedure is $\Theta(c)$, where c is the cluster size. The time complexity of Update-GAPS( ) depends on which P-2-C join is updated. In case of establishment of a P-2-C join with an isolated node, the time complexity is always $\Theta(1)$. If a P-2-C join is updated between two non isolated nodes, the time complexity depends on the level at which P-2-C join is updated as shown in Figure 3. Suppose, there are l levels below the updated P-2-C join, the time complexity will be $\Theta(c*l)$. Update-IFFT( ) procedure updates the IFFT of all the L_1 level nodes, as shown in Figure 3. In the best case, L_1 nodes of only one cluster are updated – thus resulting in time complexity as $\Theta(c)$. In the worst case, when L_1 nodes are spread throughout the network, the time complexity of Update-IFFT( ) procedure will be $\Theta(r)$, where r is the radius of the evolved network. In Table 4, these time complexities are tabulated.

Table 4. Time Complexity of PEA Protocol

Action	Algorithm	Best Case	Worst Case
P-2-P	Update-ICFT( )	Θ (1)	Θ (c)
P-2-C	Update-GAPS( )	Θ (1)	Θ (c*l)
C-2-C	Update-IFFT( )	Θ (c)	Θ (r)

4.1.2 Message Complexity

Message complexity of an algorithm is a measure of number of messages that are passed during the execution of algorithm. While calculating message complexity of PEA protocol, we do not include *HELLO Messages* because they are transmitted periodically.

The messages complexity of JDP is always $\Theta(1)$ because in all the three scenarios of JDP, at most two messages are transmitted. The message complexity of JEP and JTP depends on Update-ICFT( ), Update-GAPS( ), and Update-IFFT( ) procedures. Due to unique *Message-Id*, each node transmits update messages only once. Therefore, message complexity depends on the number of nodes involved in update. In case of update of a P-2-P join, nodes only within a cluster have to be updated. In case of update of a P-2-C join, nodes of the cluster and the descendant clusters have to be updated. In case of update of a C-2-C join, all the L_1 nodes have to be updated. We tabulate the best case and worst case message complexity for all three update procedures in Table 5 as follows:

Table 5. Message Complexity of PEA Protocol

Action	Algorithm	Best Case	Worst Case
P-2-P	Update-ICFT( )	Θ (1)	Θ (c)
P-2-C	Update-GAPS( )	Θ (1)	Θ (c*n)
C-2-C	Update-IFFT( )	Θ (c)	Θ (c*f)

Here c = cluster size, n = number of clusters under the P-2-C join that is updated, and f = number of families or L_1 clusters.

The above mentioned time and message complexity helps in appreciating the upper and lower bounds on handover latency and message overhead irrespective of the choice of policy of evolution.

5. RELATED WORKS

The idea of hierarchical addressing [8] is as old as computer networks itself. Recent research related to wireless ad hoc networks has mandated the self-configuration of addresses in nodes. Besides, the Ad-hoc Network Auto configuration (autoconf) [9] working group of IETF aims at self-configuration of nodes in MANETs. The proposals addressing the concerns, (attempt to) solve just one piece of the general addressing problem [10]. For example, most of them use some variation of DAD [11, 12, 13] but either they do not guarantee unique addresses to nodes or they do auto-configuration but at link level only [14]. On the contrary, the PEA Framework does not impose any such restrictions and solves the problem in totality. There are also attempts [15] that make use of the fact that addressing affects routing; but they rely on a single source to control the address.

The concept of Address Guided Forwarding (AGF) which is the foundation of PEA may appear similar to Self-Routing [16], but authors are not aware of any architecture or framework that allows *the self-configuration of nodes with respect to addressing and also enable self-routing*. The hierarchy built by PEA Framework may appear parallel to Landmark Hierarchy [17] and related works such as LANMAR [18]. While the landmark hierarchy based protocols assume that each node in the network has a unique identifier, PEA Framework does not make any such assumption. In fact such an assumption defeats the purpose of PEA Framework.

6. CONCLUSIONS AND FUTURE WORK

In this work, we proposed an addressing framework called Protocol for Evolutionary Addressing (PEA) that enables self-configuration of nodes in an autonomic network. We introduced the concept of Address Guided Forwarding (AGF) obviating the need for separate routing algorithm. To understand the time and message complexity, we studied PEA networks with varying core sizes and with different policies of evolution. We understood that there is a tradeoff between time-message complexity and other performance criteria such as disconnectivity. Besides, we analyzed the PEA Protocol by calculating the asymptotic time and message complexity. The proposed framework can be used for contemporary Internet and autonomic networks such as hospital network, campus network and PANs etc.

Being a nascent concept, there are many avenues yet to be explored in the PEA Framework. For given number of nodes and given routing latency, one can back calculate the core size. Authors are working on it. Further, we intend to extend PEA Framework so that it can handle the union of two separately evolved PEA networks. Naming issue (mapping from address to name and vice versa) needs to be handled before PEA Framework can be realized fully. The built-in hierarchy of PEA networks can be used to provide naming service and can also be studied for mobility management as in [19]. PEA Framework can also be extended to handle the automatic reorganization of the PEA network. Besides, the feasibility of PEA Framework can be shown by emulating the framework on top of the contemporary TCP/IP framework.

7. REFERENCES

[1] MSN Encarta. Available from http://encarta.msn.com/

[2]   Herbert A Simon, "The Architecture of Complexity: Hierarchic Systems," *In Proceedings of the American Philosophical Society*, December 1962.

[3]   G K Gupta and S V Raghavan, "PEA Framework: An Evolutionary Approach to Networking," *Technical Report IITM-NSL-TR-01*, April 2007. Available from http://netlab.cs.iitm.ernet.in/publications_files/TR/IITM-NSL-TR-01.pdf

[4]   C Huitema, "Multi-Homed TCP," *Internet Draft, IETF*, May 1995. Available from http://tools.ietf.org/html/draft-huitema-multi-homed-0.txt

[5]   D A Maltz and P Bhagwat, "MSOCKS: An Architecture for Transport Layer Mobility," *In Proceedings of IEEE INFOCOM 1998*, March 1998.

[6]   R Stewart and et al, "Stream Control Transmission Protocol," *RFC 2960, IETF*, October 2000.

[7]   MATLAB Version 7.0.0.19901 (R14), http://www.mathworks.com/

[8]   J McQuillan, "Adaptive Routing Algorithms for Distributed Computer Networks," *BBN Report 2831*, Bolt Beranek and Newman Inc., Cambridge, MA, May 1974.

[9]   Ad-Hoc Network Autoconfiguration (autoconf), *IETF Working Group.* Available from http://ietf.org/html.charters/autoconf-charter.html

[10]   C Bernardos and M Calderon. "Survey of IP Address Autoconfiguration Mechanisms for MANETs" *Internet Draft, IETF*, July 2005. Available from http://tools.ietf.org/id/draft-bernardos-manet-autoconf-survey-00.txt

[11]   S Thomson and et al, "IPv6 Stateless Address Autoconfiguration," *RFC 2462, IETF*, December 1998.

[12]   Nitin H Vaidya, "Weak Duplicate Address Detection in Mobile Ad Hoc Networks," *In Proceedings of ACM MobiHoc 2002*, September 2002.

[13]   K Weniger, "Passive Duplicate Address Detection in Mobile Ad Hoc Networks," *In Proceedings of IEEE WCNC 2003*, March 2003.

[14]   M E Chamlee, E W Zegura, and A Mankin, "Design and Evaluation of a Protocol for Automated Hierarchical Address Assignment," *In Proceedings of IEEE ICCCN 2000*, October 2000.

[15]   S Cheshire and et al, "Dynamic Configuration of IPv4 Link-Local Addresses," *RFC 3927, IETF,* May 2005.

[16]   S Lee and M Lu, "New Self-Routing Permutation Networks," *IEEE Transactions on Computers*, vol. 43, issue 11, November 1994.

[17]   Paul F Tsuchiya, "The Landmark Hierarchy: A New Hierarchy for Routing in Very Large Networks," *In Proceedings of ACM SIGCOMM 1988*, September 1988.

[18]   G Pei, M Gerla and X Hong, "LANMAR: Landmark Routing for Large Scale Wireless Ad Hoc Networks with Group Mobility," *In Proceedings of ACM MobiHoc 2000*, August 2000.

[19]   Alex C Snoeren and H Balakrishnan, "An End-to-End Approach to Host Mobility," *In Proceedings of ACM MobiCom 2000*, August 2000.