

# Swarm Intelligence and Network Administration: Applications in Ad Hoc Wireless Auto-Configuration

Dr. Eric Cole

**Abstract:** *The Dynamic Host Configuration Protocol (DHCP) has been a commonly employed technique to distribute IP addresses on networks where static address allocation may not be appropriate [1]. Auto-configuration techniques such as DHCP may be very easily done where centralized servers are commonly available. Mobile ad hoc networks are infrastructure-less, multi-hop wireless networks, which can be deployed without any pre-existing setup. Ad-Hoc networks are mobile in nature and nodes can join and leave the network at any time. Due to mobility, ad hoc networks must be able to configure themselves without an administrator's intervention. In this paper we present a swarm intelligence based model targeted at network administration. The base model can be targeted for many applications without much overhead to the system. We demonstrate the performance of the model for auto-configuration in ad hoc wireless and sensor networks. Issues such as random IP renewals and merger of partitions are considered. The main purpose of the paper essentially lies in the experimentation of swarm intelligence to network administration and control [2, 3].*

**Keywords:** Network Administration, Distributed Control, Swarm Intelligence, Wireless networks, Auto-Configuration.

**Introduction:** Configuration (such as address assignment) of nodes in distributed network architectures that lacks centralized solutions is a critical issue. Many semi-centralized and distributed techniques have been discussed in literature to deal with auto-configuration of networks but none among them are robust enough to handle the varying topological nature of ad hoc networks [4, 5, 7]. The nodes in an ad-hoc network are more basically plug-and-play type, wherein any node could enter and exit a network configuration without much intervention to other nodes in the network. A similar work is the main focus of the Zero-Configuration network set-ups. The main problem that arises when applying the techniques followed in Zero-Configuration networks to ad hoc networks is that a set of reserved IP address (169.254/24) exist for use in such networks which may not be feasible for ad hoc network set ups.

The best method to assign IP address to network nodes in any network would be to assign them statically for each node in the network. This process could become highly tedious and vulnerable to errors for large scale networks from administrators. This is one of the main reasons that the Dynamic Host Configuration Protocol was designed to

make the address assignment in IP based network automated [4]. There are basically two modes of address assignment to a network configuration which are state-full and stateless [1]. In the state-full mode of configuration a predefined set of IP address are dynamically issues either permanently or on lease to the individual nodes. In the stateless configuration (usually applied to IPv6 networks) a function of the hardware address is used to assign IP address. One generally cannot assign IP address based on the MAC address even in IPv6 networks where one-to-one mapping of IP and MAC address are possible, due to a variety of reasons [6, 7]. Security reasons could constrain the IP address to originate from a distinct set, unavailability of unique hardware addresses is another reason. The fact that ad hoc networks are dynamically configured, combined and divided makes its quite unsuitable for stateless modes of auto-configuration. Moreover it's been a recommendation of the Internet Engineering Task Force that state-full modes of auto-configuration be done on ad hoc networks as against stateless configurations. The mobile nature of ad-hoc networks makes it highly difficult to devise a suitable mechanism for dynamic host configuration with a predefined set of IP addresses. It should be noted that mobility is quite different from connectivity. A network can be independent of physical hardware connectivity (wireless medium for example) and yet be non-mobile. Ad-hoc wireless networks combine these two aspects: independent connectivity and mobility, which makes it quite complex for most of present day configuration protocols [8, 9, 10]. In this paper, we explore the possibility of distributed control theories such as swarm intelligence being applied to auto-configuration in ad hoc wireless and sensor networks.

**The Dynamic Host Configuration Protocol:** The Dynamic Host Configuration Protocol (shortly DHCP) is an IETF based protocol that enables network administrators to configure networks, by giving the individual nodes IP addresses automatically, instead of statically configuring each node. Normally the issue of a particular address to a node is in the form of a lease, for a stipulated period. When the lease period ends the entire process is renewed and the same or a new IP address is issued again to the node. When a node is no longer in need of an IP address, it informs the DHCP server which in turn pools the IP into its resource for future use. Thus we can see the IP addresses being a resource that is provided to network nodes so that two nodes do not have the same address at any point of time. Also, when mobile computer

users travel between sites, they have had to relive this process for each different site from which they connected to a network. In order to simplify the process of adding machines to a network and assigning unique IP addresses manually, there is a need to automate the task. The simple task achieved by the DHCP becomes quite complicated when mobile nodes and users take effect and that the protocol has to be made applicable to mobile user-environments. Manual configuration requires the careful input of a unique IP address, subnet mask, default router address and a Domain Name Server (DNS) address. In an ideal world, manually assigning client addresses should be relatively straight forward and error free, but network administrators are prone to errors when large scale networks are configured.

The working of the DHCP is based on the operating system that supports it. The client usually sends a request message to get an address from the DHCP servers. One of the DHCP servers responds with an *ack* message that it can provide DHCP services to it [11]. The client usually decides as to accept or reject the IP address allocated to it. If in case the client prompts an error message for a particular IP address the server takes up the task of reassigning a new and valid IP address. Many security and topological constraints govern this process which makes it more complicated. There are release messages from the clients that indicate that the client is no longer in need of a particular address or that the lease period has expired [11]. Most of the current operating systems provide for the inclusion of the DHCP in their configuration and cross operating system compatibility has not been a problem with its functioning. On the same hand things can become more complicated when in future many small scale non-PC based embedded devices with unconventional operating systems try to involve themselves in the auto-configuration activities of a network. Newer versions of the protocol which can include such embedded devices are in high demand.

**Duplicate Address Detection:** The duplicate address detection scheme is quite a successful scheme in auto-configuration of ad-hoc networks by Perkins et al [6]. Auto-configuration takes place in ad hoc networks by route discovery mechanisms. A node in the ad hoc set up picks up a randomly generated address which it has to verify for uniqueness. The randomly generated address is verified for its uniqueness by the node sending a route-request for the particular address along the entire network. If it get no route-acknowledgement for the particular address from any of its peers or other nodes in the network within a timeout period it retains the generated address for itself. If it gets a route-acknowledgement back it would have to drop it for another randomly generated address. If there are three or more way locks in addresses the node that gets to know that its address is being used by others it decides to drop it.

This method goes progressive until just one node is left with the address that was negotiated for. Clearly this method has flaws along various disciplines and does not work completely well in the case of ad hoc networks. Much of the difficulty stems from the fact that time bound in ad hoc networks are not well defined. Problems could arise when timeout bound are exceeded in address confirmation [4]. In ad hoc networks due to mode mobility and transient topology, often times time bounds can be indefinite and out of bounds. Particularly, when partitions merge, the resulting network may contain nodes with duplicate addresses [8, 11]. For correct behavior, this scheme must be augmented with a procedure that detects merging of partitions, and then takes suitable actions to detect duplicate addresses in the merged partitions. Distributed sciences could be used for devising such schemes effectively.

Other more suitable solutions would be in having 'leader selection' algorithms run on the networks that would be effective in taking responsibility of assigning addresses to other nodes in a network configuration. Such algorithms also tend to be more towards centralized which may not be always feasible with ad hoc and sensor networks where node-to-node independency is a highly appreciated attribute. Security may be another important issue with such centralized architectures. A hacker who takes over the centralized-functioning node would play havoc on the entire configuration. The centralized leader nodes are not very specialized nodes and are leaders for only this specific purpose. Moreover the leader among a configuration could be anybody and it is difficult to track it constantly. What would be ideal is a situation wherein no node occupies specialized responsibilities yet addressing of individual nodes is achieved uniquely. Mutual exclusion algorithms can be very handy in assigning addresses to network nodes in a configuration in a distributed manner. At the same time the distributed algorithms do not run as effectively as centralized ones in terms of computational and delay overheads. Optimizing the performances of distributed algorithms is a highly researchable topic and no general feasible solution in terms of optimality exists as of today.

**Distributed Techniques for Dynamic Configuration:** Dynamic configuration of a network which has a high number of nodes, is an intractable problem and closed form solutions for it are non-existent. On the other hand the configuration needs for ad hoc networks in principle demands a non-centralized solution. This necessitates the revision of distributed techniques for the problem where centralized entities are completely or partially eliminated. Individual nodes themselves must be responsible for collectively processing global statistics (auto-configuration for example) with partners in the network to achieve such solutions. This is where distributed control theory plays an important role. With distributed set-ups auto-configuration

becomes a problem of how nodes communicate and coordinate with each other to reach a specific stable state. Mostly the convergence to stable and non-oscillating final configuration requires carefully designed feedback mechanisms operating on a node-to-node basis. An important point to be noted in the case of distributed architectures is that the entire feedback component of the system itself is distributed. What makes the aggregated feedback at any point in the system is the addition of these feedback forces due to the individual nodes and is highly unrealizable for direct measurement. Often times in improper design conditions the results may not be totally feasible.

The application of distributed techniques for ad-hoc networks requires a model that is effective propagating feedback information quickly and effectively. When the network size is big, communicational and delay overheads would place themselves in front of the designer as an impediment. The assignment of IP addresses to the nodes in a distributed set-up would not totally have information of whether or not another node is being given the same IP address at some other part of the network. In such circumstances the feedback information of the current state of the system has to travel distances on the network configuration to spread it. Not only spreading relevant information becomes important, the removal of obsolete and redundant information quite quickly is also a must. The performance of these distributed algorithms should take in to their account the merger of two or more different ad hoc networks. When two different ad hoc networks merge there may exist difference in lease period of address, non-unique IP addressees, non-compatible routing and communicational set ups. The DHCP in such cases should be able to distinguish the salient features of such networks and modify itself to adapt to the given changed configuration.

**The Swarm Model:** The swarm model basically consists of an ant-colony model that is tuned for any target application [2, 3]. In this paper we have targeted its application to auto configuration in wireless ad hoc networks. The body of the swarm consists of a various data structures aimed for different applications. In our Ant Colony based system, every node in the network has an *originator ant* which is born at the node. The originator ant has a very special relation to the node that it is associated with. Information exchanged between the originator ant and the respective node would be mutually beneficial to both. In other words, the originator ant has the only privilege to update any information that node might be attributed to. Similarly the node has the privilege of destroying, reproducing or duplicating the originator ant. In any system there is at least one originator ant for each node. The ants carry various information segments associated with its originating node that would be

propagated along its journey on the network. The information interchange at the various nodes by the ants would be embedded from and to the environments. The environment is normally realized as a small segment of memory that is exclusive to interactions with the incoming ants. The interaction would normally be two way, with the ants and the environments updating themselves. This forms the feedback mechanism of the system based on the principles of stigmergy.

The ants are normally identified on their Media Access Control address or some kind of its hash. This is highly useful for the nodes to realize its originator ant and vice versa. A colony of swarm that is employed on a network need not be restricted to a single functional purpose. The interacting swarm could be designed to handle simultaneous functionalities which normally would require exclusive mechanisms for each. This increases the flexibility of the system in terms of executing various administrative controls with a single system. For instance the same swarm system that is used for auto-configuration could be used for network monitoring, security, resource and domain name discovery, routing, self-healing and so on. This makes the swarm intelligence based system highly generic in that multiple network administrative goals can be achieved with little overhead addition on an existent design. Even though the system may be sub-optimal in terms of individual administrative functionalities, the flexibility of the system in getting modified to additional functionalities makes it a highly interesting approach for network administration.

**Ad Hoc Network Auto-configuration:** Ad Hoc wireless networks forms on the less understood and controllable network set ups among network configurations. Since no central entity is present to control (most of the times the central entity is undesired or unwanted) distributed sciences and services are required for network maintenance and control. The Duplicate Address Detection method (Perkins et al) is a much favored solution that has been employed for auto-configuration does not suit to every as hoc network realization [6]. High latency and delays within the network could be a drawback to the system [8, 11]. Network swarms are employed to carry Internet Protocol (IP) and MAC address of the nodes that it has visited associated with a timestamp.

Timestamps would denote when an ant has visited a particular node to receive a particular IP address that it carries. The timestamps could be realized as a counter running on each node towards a large maximum value in discrete increments of time. A higher timestamp associated with a particular entry (IP addresses) can point to more recent information and thus more accurate for updating. Once the timestamp counter at any node reaches its maximum value, it can wait for its originator ant to destroy

it. Once an originator ant is killed or destroyed by a node, it creates a new ant for itself to the originator ant with a fresh counter. An important point to be noted is that there could be real time situations where the travel of ants from one node to another could be significantly high enough to affect the entire system. At such instances a more sophisticated technique such as Lamport's logical clock scheme could be employed.

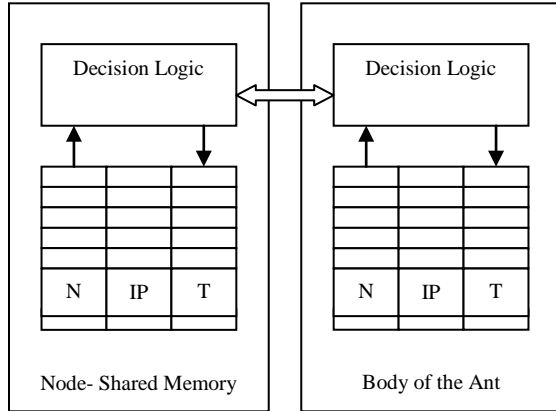


Figure 1. Information transfer between a node and an arriving ant

The exchange of information between a node and an ant would be based on the timestamps the ants carry on a per entry basis. For example, on a network with  $N$  number of nodes, the ants would carry  $N$  IP address one for each node (usually the most recent one according to its knowledge). Each of these  $N$  IP address would be associated with a timestamp that points to the time that the ant had visited each of nodes for the last time (at any point of time). When information exchange between the node (a shared memory segment at every node pointing to the 'environment' in Ant Colony System) and the arriving ants takes places, either of them updates itself based on the timestamps. Let  $P_{i,node}$  and  $P_{i,ant}$  point to the  $i$ th entry in the table of entries at the node and the ant respectively and  $T_{i,node}$  and  $T_{i,ant}$  be the corresponding timestamps. In the case for the ad hoc auto-configuration application the entries would be the IP addresses. Based on whether  $T_{i,node}$  is greater than or less than  $T_{i,ant}$ ,  $P_{i,node}$  would retain itself or change to  $P_{i,ant}$ . The updating of  $P_{i,ant}$  would be done on a similar basis. Eventually after updation both  $P_{i,ant}$  and  $P_{i,node}$  would point to same value depending on the timestamp. In addition to the IP addresses getting updated as per the above-mentioned logic, there is updation of the timestamps themselves too. For instance, when an ant

meets with a node which has a particular  $T_{i,ant}$  different from the corresponding  $T_{i,node}$  present in the shared memory segment at the node, for the same MAC address, IP address combination, updation of timestamps takes place. Both the ant and the node would get themselves updated to the larger of the two timestamps  $T_{i,ant}$  or  $T_{i,node}$ . Figure 1 depicts the interchange of information between an ant arriving at a node and the node.

The originating ants as said have exclusive rights to change the IP address of the nodes. When the originating ant is lost on the network or does not return for a long time, the node can create other originating ants for itself so that the system can be maintained. Whenever an ant during the process of its journey detects a potential *conflict* for the node it has originated from, it takes responsibility to inform it about it and have it changed. A conflict is defined as a situation when two or more nodes have chosen the same IP address for itself. Conflict resolution mechanism is based on mechanisms followed in Zero-Configuration networks. The node that has the least MAC address (or a hash of it) takes the responsibility to have its node change its IP address to a different one. Since the MAC address are most often unique when more than one node has the same IP address, all the nodes except the one with the highest MAC address among them change. This is not a one step process but is a result of various interactions among the swarms. The conflict resolution mechanism would continue until a state wherein all the nodes have unique IP address is reached.

Most often in many approached changes to the network topology, renewal of IP addresses, node and link failures are too difficult to consider for, when devising a system for auto-configuration. In the swarm based system any node loss would not affect the system in total as the information. Swarm intelligence poses a completely distributed control and feedback flow and nowhere does feedback gets accumulated. This property ensures that even on a case of node or link failure, only a partial component of information flow is lost and given sufficient time the system can recover from it. Such properties would be excellent when practical networks are considered which can be riddled with abrupt link and node failures. The most important point to node about configuring an ad hoc network is when partitions break up or merge together. The main drawback of the Duplicate Address Detection scheme is its inability to completely handle network partitions and network mergers. An important feature of the swarm based model is partitions need not be a special case and does not require special consideration. The swarms are exchanged across the partitions with dynamic information that can be harnessed by both the partitions and their conglomeration onto one big partition becomes

much easier. However the time to converge depends upon various parameters such as topology, the sizes of the different partitions, and the number of ants deployed in the whole system and so on.

**Analysis and Results:** Simulations were carried out to analyze the performance of the interacting swarms for ad hoc auto-configuration. Node and link failures were considered during the simulation process during burst intervals. Every node was given a set of neighbor nodes to which it can directly communicate in a duplex manner. Note that the neighborhood set for any node is transient with time as node and link failures occur. Simulations were carried out for a total of hundred nodes with 5 to 6 node neighborhood. An entire class C (IPv4) domain IP range was used for auto-configuration. This would amount to 256 different IP addresses to begin with for the 100 nodes to pick up a unique IP address each eventually.

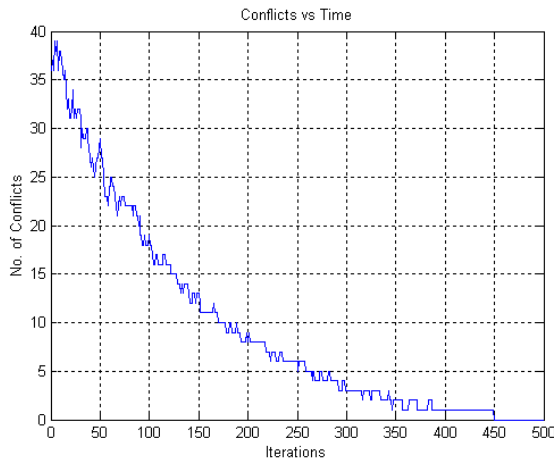


Figure 1. Performance with no IP renewals

Time is measured in iteration counts of the simulation process. Figure 1 show the performance characteristics when 100 nodes (with 100 ants) attempted to configure for a Class C network. The performance was measured as the number of conflicts which is also defined as the number of non-unique IP addresses on the network, against time. For the characteristic it's clear that the number of conflicts goes down with time and finally attains a point where there are no conflicts at all. When the system reaches the state where no conflicts are present it is guaranteed that each node has a unique IP address and thus auto-configuration is achieved.

Figure 2 shows the same performance characteristic but IP renewals were allowed for nodes. IP renewals occur when a node happens to lose a configured IP address and tries to address itself with a newer one. This could happen when

the user of the particular node decides to change his/her IP address due to a variety of reasons (security, mobility etc). Since random IP renewals were allowed to occur the convergence of the system with the same number of nodes, ants and IP addresses (as in Figure 1) is a little slower. The renewal of IP addresses depicts itself as glitches and performance shows that the system adaptively responds to it. In both cases, we can see progressive convergence as will be the case with any adaptive system. The simulation results postulate a reason for the systems applicability on real time networks.

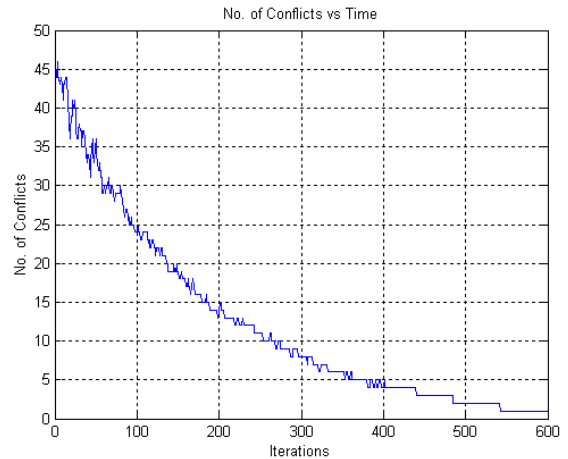


Figure 2. Performance with random IP renewals

An important consideration that was considered on the simulation was when two different partitions merge. When partitions merge there is a sudden increase in the number of IP address conflicts and the system has to respond to the new environmental change. The sudden increase of IP address conflicts is a result of many nodes having same IP addresses but in different partitions, happen to come together. Figure 3 shows the performance characteristics when two different partitions (with 50 nodes in each) happen to merge into a single partition. The figure represents a newer partition coming into an already converging partition at time (T= 250 iterations). There occurs a sudden spike at around the 250 mark which shows the merger and the convergence of the new network henceforth.

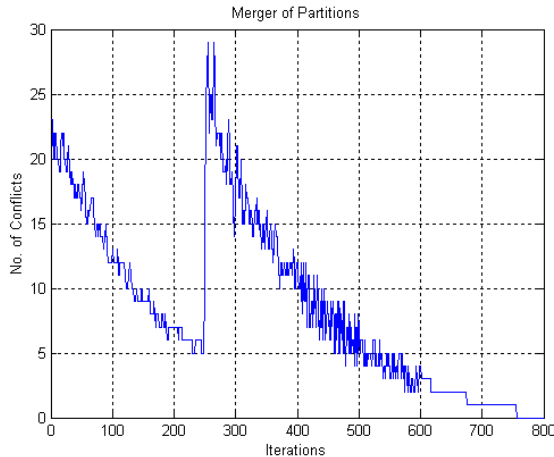


Figure 3. Performance during a partition merger

**Future Work:** Many interesting aspects of the swarm based system are yet to be researched and studied. The applicability of swarm intelligence to network engineering is a fairly recent endeavor and needs high end research. Analysis need not be curtailed to simulations but also should be carried forward to be realized on practical wireless and sensor networks. There are much to be dealt with the simulation set up and analysis itself. To enrich the flow of feedback in the system, multiple originator ants can be incorporated for every node. This would be highly helpful in propagating information in large networks faster. But before we employ such techniques a complete analysis on its fallouts has to be done. Another issue comes up when time timer reaches threshold and has to start over again. To deal with such instances, a maximum hop count limit could be incorporated for keeping entries pointing to the nodes which facilitate dropping redundant entries. Such an observation is yet to be observed on a simulative basis.

**Conclusion:** Dynamic Host Configuration Protocols in their present state cannot be extended to ad hoc and sensor networks. The problem with their applicability to these networks is that the DHCP requires a centralized server for executing its functions. Dissatisfaction exists among network administrators in implementing dynamic configuration for most wireless and mobile set-ups. Even today administrators prefer to manually configure nodes in mobile networks rather than automate it. Though manual configuration is the best in a lot of circumstances, highly mobile and large nodes requires something less tedious. Techniques such as duplicate address detection attempts to solve the problem of auto configuration, yet their relevance to present day implementation is merely theoretical. Many problems exist when porting such solution on practical basis. The fact that the establishment of large scale wireless and sensor networks is just incited and has many levels to travel has made this an indispensable problem to

consider. Research on the fast catching large scale mobile networks would attempt to solve the problem more efficiently. The applicability of swarm intelligence to network problems such as DHCP provides a new facet to algorithm development and design of future network architectures.

**Acknowledgements:** This work is a part of the research efforts conducted at the Advanced Technology Research Center, TSGI

#### References:

- [1] [www.cisco.com](http://www.cisco.com)
- [2] E. Bonabeau, M. Dorigo, G. Theraulz, "Swarm Intelligence- From Natural to Artificial Systems", Oxford University Press, 1999
- [3] M. Dorigo, V. Maniezzo, A. Coloni, "The Ant System: Optimization by a group of cooperating agents", *IEEE Transactions on Systems, Man and Cybernetics-Part B*, 1996
- [4] R. Droms, Dynamic host configuration protocol., RFC 2131, Mar. 1997.
- [5] T. Narten and R. Draves, "Privacy extensions for stateless address auto configuration in IPv6", RFC 3041 2001
- [6] C. Perkins, J. T. Malinen, R. Wakikawa, E. M. Belding-Royer, and Y. Sun, "IP address autoconfiguration for ad hoc networks," IETF Draft, 2001
- [7] S. Thomson and T. Narten, "IPv6 stateless address auto-configuration", RFC 2462, 1998
- [8] S. Nesargi and R. Prakash., "MANETconf: Configuration of hosts in a mobile ad hoc network", *Proc. of IEEE Infocom 2002*, New York, June 2002
- [9] M. Ilyas, "The Handbook of Ad Hoc Mobile Networks", CRC Press 2003
- [10] M. S. Gast, "802.11 Wireless Networks- The Definitive Guide", O'Reilly Press, 2002
- [11] N. Vaidia, "Weak duplicate address detection in mobile ad hoc networks", *Proceedings of the 3rd ACM international symposium on Mobile ad hoc networking & computing*, 2002