Trajectory Control of Mobile Gateways for Range Extension in Ad Hoc Networks

Mohiuddin Ahmed [a,*] Srikanth Krishnamurthy [b] Randy Katz [c] Son Dao [a,1]

[a] *Information Sciences Lab, HRL Laboratories, Malibu, CA.*

[b] *Computer Science Dept., University of California, Riverside, CA.*

[c] *Computer Science Dept., University of California, Berkeley, CA.*

Abstract

In order to facilitate scalability in a mobile ad hoc network (MANET) and to provide connectivity between partitions that might occur in wireless networks as a consequence of mobility, we can envision a 'range extension' network that consists of airborne communication platforms, and satellites. These airborne or satellite nodes will maintain communication links with specific 'gateway' nodes among the mobile ground nodes. To communicate with a node that is geographically distant or belongs to a different network partition, an ad hoc node can relay its data packets through an appropriate mobile gateway and via the range extension network. If we envision that the MANET is divided into different groups and a mobile gateway is deployed for each such group, an objective then, is to determine the trajectory of the mobile gateway to best serve the ad hoc group to which it belongs to, in terms of network performance metrics such as throughput and latency. In this paper, this problem of computing the optimal position for a gateway is reduced to a linear optimization problem by means of some simplifying but realistic assumptions. We suggest methods that may be deployed to enable the gateway to follow this optimal trajectory as closely as possible (within the practical constraints imposed by its velocity and maneuverability). Simulation results for various scenarios show a 10-15% improvement in the throughput and in latency (per group containing a gateway) if a gateway has a dynamic trajectory whose locus follows the computed optimal position, as compared to a gateway that is statically placed at a regular position, or to a gateway that has a random trajectory.

Key words: mobile ad hoc networks, gateway, convex optimization, trajectory control.

1 Introduction

Mobile ad hoc networking technology [1] may be appropriate for linking mobile computers in an office or home environment, deploying wireless sensors in remote or inhospitable terrain, coordinating disaster relief efforts after natural catastrophes, or in tactical deployments for situation awareness applications [2]. A major challenge in the wide deployment of MANETs has been in achieving scalability. Furthermore, due to the range limitations of the nodes in the ad hoc network, the network might often be divided into isolated partitions. In order to achieve scalability in terms of efficient communications between geographically distant nodes or between nodes that belong to different isolated partitions (each of which is an ad hoc group by itself), it is desirable to provide a supporting infrastructure in the form of a range extension network. This infrastructure is also essential to interface the MANET with the Internet.

This range extension network could typically consist of airborne relay nodes or low earth orbit/geostationary satellites. In order to interface the ad hoc network with the range extension network, one can envision the deployment of special gateway nodes in the ad hoc network. These are 'on-ground' nodes that might be more power/processing capable than the other ad hoc nodes on the ground and are equipped with the appropriate hardware for communicating with the satellite/airborne nodes. This architecture can therefore be visualized to consist of two layers. The first layer includes the ad hoc network, and the second includes the range extension network consisting of satellites or airborne nodes. The mobile gateway provides the interface for the communications between the two layers and hence we shall call a gateway node a "Cross Layer Communication Agent" or CCA from this point onwards. Similar architectures have been previously considered for enabling hierarchical routing or multicasting ([3]–[4]). In [3], and [4], topological information is used to build spanning trees rooted at convenient nodes in the MANET. In [5], the authors have investigated the feasibility of partitioning MANETs into groups with cluster-heads or repositories of information to enable efficient information dissemination [5]. In contrast, our objective is the determination of the CCA trajectory that a mobile gateway or CCA has to follow in order to optimize inter-domain network performance.

A CCA may be assigned to a group of ad hoc nodes or it might be affiliated

* Corresponding author.
Email addresses: mohin@hrl.com (Mohiuddin Ahmed), krish@cs.ucr.edu (Srikanth Krishnamurthy), randy@cs.berkeley.edu (Randy Katz), skdao@hrl.com (Son Dao).
[1] This work was supported by Defense Advanced Research Projects Agency (DARPA) Contract No. N00014-99-C-0322.

with a geographical domain and placed statically at the center of the domain. In the former case, it links the range extension network with the specific group of ad hoc nodes. In the latter case, it provides the ad hoc nodes within the specific geographical domain with a link to the range extension network. If the CCA is mobile and is affiliated with an ad hoc team of mobile units, the question arises as to where the CCA ought to be located relative to the mobile units. In other words, the objective is to specifiy the locus of the position that a mobile CCA is to follow and the rules that govern this trajectory with respect to the relative motion of the other mobile units. We design a methodology for defining the CCA trajectory based upon the location, loading, etc. of the other mobile nodes in the ad hoc group that the CCA serves. We show that network performance improves (for communication involving nodes in different clusters of nodes), in terms of throughput and latency, if the CCA trajectory is computed based on our methodology.

We derive a relatively simple analytic formulation for the optimal CCA position, which is equivalent to a linear optimization problem. This is discussed in Section 3. We also provide an algorithmic implementation of the formulation, and discuss the effect of some of the parameters in this section. Various scenarios are considered for deliberation. In Section 4, we estimate the overhead for implementing this architecture with the aid of typically used media access control (MAC) and routing protocols. We also investigate the computational complexity of our CCA trajectory definition algorithm. In Section 5, we discuss our simulation framework and discuss the results. We conclude in the final section.

2 System Model

We motivate the discussion of the system architecture that we laid out in the previous section in the context of the following scenario. Consider separate groups of mobile ad hoc nodes operating in a terrain with blockages and deployment area restrictions (e.g. troop divisions deployed in a mountainous area). Each group would have one or more CCAs capable of communicating with an airborne or satellite node with which it has a direct line of sight connection. As an example, in Figure 1, we have considered two isolated groups of mobile nodes, each forming a MANET by themselves, and each having its own CCA. The ad hoc group of nodes that use a particular CCA to communicate via the range extension network are said to belong to that CCA's *domain*. The CCA in each group is then the conduit via which the ad hoc nodes in the separate groups can send data packets to each other, with the routing assistance of the airborne node. The airborne node can also serve to connect the clusters to a wired infrastructure (e.g. command and control centers outside the theater of operations).

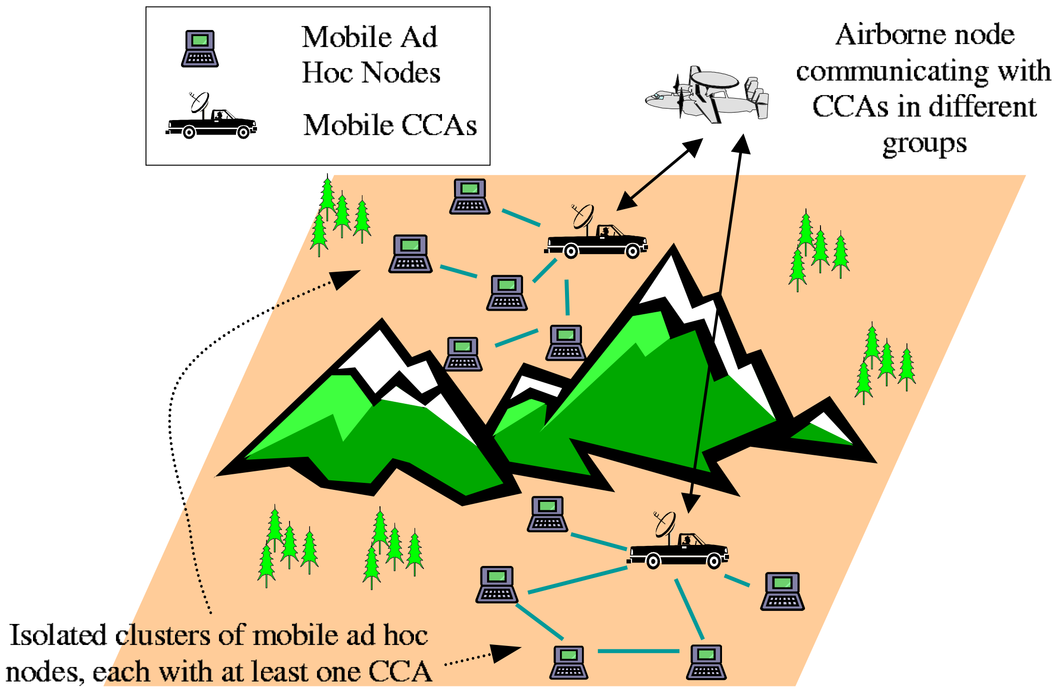
Mobile Ad Hoc Nodes
Mobile CCAs
Airborne node communicating with CCAs in different groups
Isolated clusters of mobile ad hoc nodes, each with at least one CCA

Fig. 1. Ad hoc network of two groups of mobile nodes and CCAs.

As the nodes in a particular group move, the objective then is to specify the 'optimal' trajectory for the mobile ground CCA associated with that group. For communication intended for nodes within a given group, the nodes would not be compelled to use the CCA, but would instead rely on the underlying MANET architecture using traditional routing, MAC protocols, etc. By intelligently positioning the CCA, we might expect to achieve better network performance for ***inter***-domain node communications, (i.e. data communication between nodes that are in the domains of different, possibly geographically isolated CCAs) than if the CCA were allowed to move randomly with respect to the nodes it its domain.

We can assume any suitable mobility model for depicting the motion of the ad hoc nodes [6]. Thus, some of the nodes may be statically deployed (sensor nodes) or, the nodes may move according to a random waypoint model [7]; or, they may move according to a model in which they approximately follow one particular node that is considered the 'leader', etc.

The terrain in which the MANETs are deployed may contain regions where either the node, or the CCA, or both, cannot reside. These can be regions where, e.g., channel impediments create radio nulls, or hazard zones which expose the CCA or specific nodes to harm (in tactical deployments), or where the terrain is inaccessible, etc. We refer to these regions as *blockages*.

The performance metrics that could potentially be improved by optimizing the CCA trajectory include, but may not be limited to, the following:

- *inter-domain* network data throughput
- *inter-domain* network packet transport delay
- total power expended (or maximum power consumed per packet/bit per node in the CCA)
- data transmission reliability (packet drop/error rate)
- volume of the network control messages and resulting signaling overhead.

The procedure by which the weighted centroid is computed requires the following:

(1) Each node is equipped with a GPS device that enables the node to determine its position.
(2) Each node can estimate its offered load in real time
(3) Terrain information (such as specific coordinates or boundaries of radio null regions where radio signals cannot propagate, inaccessible terrain, etc.) is available at each CCA. This can easily be made available at network inception [2].

The details of the actual communication mechanisms that enable the MANET to function are not directly relevant in the development of our analytical formulation for computing the optimum trajectory that our CCA ought to follow (Section 3). For ***intra***-domain node communications (i.e. communication between nodes that are in the domain of the same CCA), the MANET could rely on well-established protocols such as the IEEE 802.11 MAC protocol for media access control [8], DSR, DSDV, or AODV for routing [7,9], etc. to establish and maintain connectivity. For ***inter***-domain node communications, data will have to be routed through the CCA and via the range extension network.

3 CCA Trajectory Update Algorithm: Formulation and Analysis

In this section we describe the algorithm for determining the trajectory of mobile CCAs such that it is optimal in terms of 'relative position' with respect to the group of ad hoc nodes that it serves. By having the CCA follow the trajectory determined by this algorithm, we expect (and later show by means of simulations) that we will achieve a significant improvement in network performance, in comparison with a scheme that has static CCAs or has the CCAs follow a trajectory defined by a random way-point model. We describe our algorithm assuming that there is single CCA per domain in Section 3.1. However, it is possible for several nodes among a cluster of nodes to be capable

[2] Position based schemes have previously been suggested and studied for ad hoc networks, [3], [4].

of communications with the range extension network and hence any of these nodes could assume the role of a CCA. Fortunately, in all these cases, the base algorithm that we propose remains unchanged. Increased layers of complexity can be added to the base algorithm to enable the CCAs to participate in node 'hand-off' as in cellular networks, or to intelligently share the load generated by the nodes in the overlapping regions of intersecting domains. These features may be exploited to achieve further performance improvements. A brief discussion of the possibilities is presented in Section 3.2.1.

3.1 Node Domains Containing a Single CCA

As mentioned earlier, one might expect that the performance of the network would be best if we could optimally position the CCA within the group of ad hoc nodes that it belongs to. We refer to this optimal position as the *weighted geographic centroid*. This section describes the procedure that computes this weighted centroid. We formulate an optimization problem that the CCA solves periodically with the help of information that it has gathered from the other ad hoc nodes, to determine its trajectory. The parameters that the CCA can take into account in formulating the optimization problem could include node positions, each node's offered load, data traffic patterns, priority of the generated traffic, the channel signal to interference noise ratio (SIR), among others. The choice of which parameters to include is determined by the specific network metrics (listed in Section 2) that are of importance. We can easily have explicit formulations for every desired optimization objective. As we shall see, the optimization formulations could essentially be represented as either linear or convex programs [10]. For the purposes of this discussion, we consider positions and their offered load as our primary parameters and the network throughput and the average delay experienced by inter-domain data packets as our basic performance metrics.

Figure 2 represents a typical scenario showing ad hoc groups with a single CCA per group. As shown in the figure, terrain contains blockages (described in Section 2) that can always be represented by simple rectangular regions. In particular, each terrain blockage can be bounded by the smallest rectangular region that contains that blockage. If we further assume that the ad hoc group is deployed in a finite area which represents the 'domain' of a CCA, then the area of deployment may also be similarly bounded by a rectangular region. This representation is useful since the constraints that govern the position of a CCA can then be described by simple linear equations (based on the coordinates of the rectangle boundaries). Note that it is extremely hard if not impossible to characterize the arbitrary geometries that these regions might have by exact mathematical expressions. Thus, our model for the region over which the CCA can move consists of a rectangular area with smaller

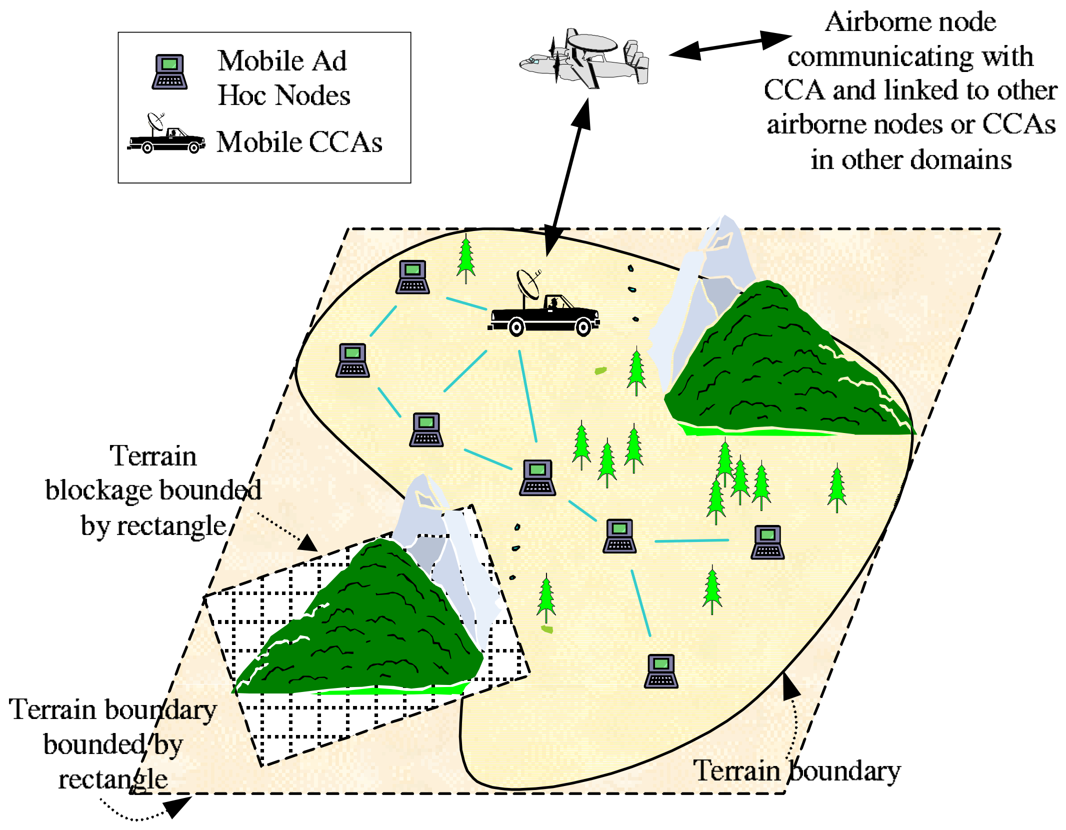
Mobile Ad Hoc Nodes
Mobile CCAs
Airborne node communicating with CCA and linked to other airborne nodes or CCAs in other domains
Terrain blockage bounded by rectangle
Terrain boundary bounded by rectangle
Terrain boundary

Fig. 2. CCA domain with location bounds.

rectangular regions marked as ‘no-go’ areas.

A fundamental characteristic of MANETs, (also shown in Figure 2) is that, not all the nodes in the domain have a direct link to the CCA. Some nodes will be outside the single-hop radio range of the CCA and will have to route their data packets to the CCA via multiple hops through other ad hoc nodes[3]. The cost function that we use in computing the weighted geographic centroid takes the offered load of the individual nodes and the priorities of the packets generated at each node into account. This ensures that the CCA is closest to the most heavily loaded nodes, or to the nodes that generate packets of the highest priority, as the requirement might be. Thus, when all packets are of the same priority, the CCA is positioned such that most of the data packets reach the CCA in a single hop, thereby ensuring a better utilization of the available resources. In the case wherein nodes generate packets of different priorities, the CCA's position would ensure that most of the higher priority packets would reach the CCA in a single hop.

[3] These data packets are destined for nodes in other domains and thus *must* be routed through the CCA to the range extension network.

3.1.1 Optimization Formulation

We assume that the CCA acquires the coordinates of each node in its domain, that needs to send inter-domain data packets[4]. The weighted centroid is simply a factored distance norm, and the constraints are linear inequalities. For a domain with n nodes, the optimization problem can be formulated as:

$$\text{minimize} \sum_{i=1}^{n} f(\tau_i, \rho_i) \cdot |\mathbf{x}_0 - \mathbf{x}_i| \tag{1}$$

$$\text{subject to} \quad \mathbf{w}_1 \leq \mathbf{x}_0 \leq \mathbf{w}_2 \tag{2}$$

$$\mathbf{x}_0 \geq \mathbf{b}_{2k} \tag{3}$$

$$\mathbf{x}_0 \leq \mathbf{b}_{1k} \tag{4}$$

The optimization variable in the norm minimization expression, Eq. 1, is the CCA position, represented by the 2-D (ground) position vector, $\mathbf{x}_0$, with reference to any suitable origin in the terrain of interest. The other $\mathbf{x}_i$'s represent the coordinate vectors of the mobile nodes with respect to the same origin. The values of $\mathbf{x}_i$'s are obtained at each sampling instant and a new value of $\mathbf{x}_0$ is computed. The weighting factor, $f(\rho_i, \tau_i)$ is a user defined function that depends on the i^{th} node's load, ρ_i, and priority, τ_i. Note that when we use the term load, we in fact refer to the data traffic that the i^{th} node wants to send to nodes in other domains. We do not consider the data load due to intra-domain communication among the nodes, although this may affect the available bandwidth for inter-domain communications[5]. Depending on the type of traffic being generated by the nodes, the function $f(\rho_i, \tau_i)$ can be defined appropriately to reflect CBR, or variable bit rate (VBR) traffic and with or without defined priorities. The terms $\mathbf{w}_i$ are the vector coordinates representing the outer rectangle circumscribing the domain (bottom left and the top right points), and $\mathbf{b}_{ik}$'s are similar vectors that represent the boundary of the k^{th} blockage. Therefore, we are minimizing the sum of the weighted geometric distance from the CCA to each of the nodes, subject to the boundary and blockage constraints.

As stated earlier, this problem is a non-linear optimization problem [11]. However, if we ensure that the cost function is the $\mathcal{L}^2$ norm, since the constraints are linear, the problem is a convex program (all norm functions are convex). This problem, therefore, has a global optimum and any of the standard convex optimization algorithms can be used to find the optimal position, $\mathbf{x}_0$ for the

[4] We will discuss the technique by which this aquisition is realized and the resulting overhead incurred involved in Section 4.

[5] Alternatively, we can assume that separate channels exist for intra-domain and inter-domain communications.

CCA (e.g. steepest descent, Newton's methods, etc. [11], [10]) [6]. Furthermore, the convex optimization problem itself can be easily transformed into a simple linear program (LP) and the cost function can be replaced by an equivalent linear cost function. The resulting LP can then be solved far more efficiently via modern interior point methods [12]. An implementation would require the CCA to perform the **CCA Trajectory Update Algorithm** which is as follows.

- *Input constants (set 'a priori')*: terrain and blockage boundaries, sampling times, optimization metric of interest.
- *Output*: optimum CCA location computed at specific sampling times.
- {*While nodes in the domain have inter-domain data packets to send*}, **DO**:

(1) Collect or estimate the position of each node, $\mathbf{x}_i$, at each sampling instant.
(2) Collect from each node, an estimate of its current load and the priority that it desires [7].
(3) Perform a local computation to solve the LP equivalent to the optimization problem in Equation 1 and obtain optimum CCA location for that sampling instant.
(4) Move towards the optimal location in the most suitable manner, as allowed by the physical constraints [8].
(5) Repeat at next sampling instant.

The motion of the CCA can be further governed by certain rules to prevent race conditions and such. As an example, one can have a hysteresis rule that helps to prevent excessive CCA sensitivity, wherein a computed 'new' CCA location has be greater than a minimum of some pre-specified δ units from the present location before we decide to move the CCA. Another hysteresis rule might require the CCA to remain at a newly computed location that it has moved to, for a specific time period (usually several frame update intervals).

3.2 Overlapping CCA Domains

In the methodology discussed so far, we have only considered domains with a single CCA serving a group of ad hoc nodes. In this section, we consider a case in which ad hoc nodes have the ability to chose one CCA among a set of CCAs. Without loss of generality, we can think of this scenario to be equivalent

[6] Since the problem is formulated as a convex program, the solution is found by numerical methods.
[7] The priority is determined by policy and is not discussed in this paper.
[8] The problem of navigating from one location to another, on a 2-dimensional surface with obstacles with no pre-established paths, is a vast area of research in robotics and is not discussed in this paper [13].

to the case of two or more domains with single CCAs that intersect due to their proximity. A node that lies in the overlapping region of the intersecting domains can choose any of the CCAs to relay its inter-domain traffic to the range extension network.

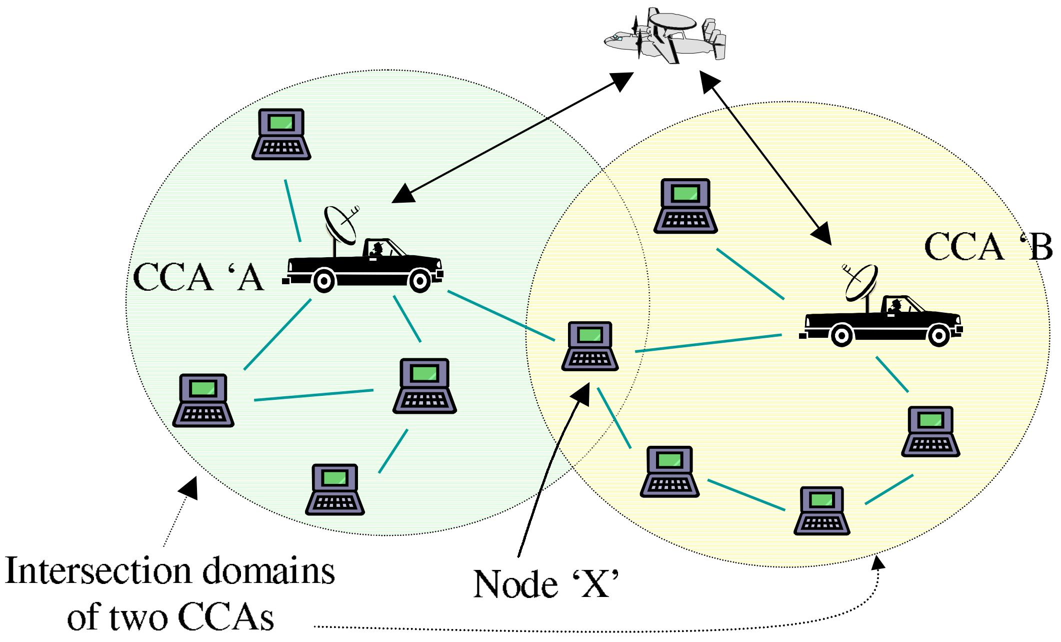

Fig. 3. Overlapping CCA domains

In Figure 3, each CCA has a specific domain space defined by a geographical area (the circles enclosing the CCAs A and B) and serves a set of ad hoc nodes that are confined to this region. Although the specific motion of the individual nodes may themselves be essentially random, they remain affiliated with, and within the boundary of the domain of a particular CCA. When two or more such independent domains intersect, then one or more of the ad hoc nodes may lie within the domains of multiple CCAs. In Figure 3 node X was originally affiliated with CCA A, but is now also in the domain of CCA B. It is now possible for node X to communicate with CCA B (appropriate signalling methods will be required but are not discussed in this paper). In fact node X might want to migrate from the domain of CCA A to that of CCA B, i.e. it might wish to switch its affiliation. In such a case, node X will indicate its intent to CCA B. Then one of the following is possible:

- CCA B may completely ignore this message from node X. In this case, its trajectory remains unaffected by node X’s message.
- CCA B may agree to relay the data message from/to node X to/from the range extension network. However, it may not use the control information from node X (that reflect the position of node X, its offered load, etc.) in determining its trajectory.
- CCA B may consider node X to temporarily belong to its domain. In this case, not only does CCA B act as node X’s interface to the range extension

network, but it also takes the position, the offered load, and other parameters associated with node X into account while determining its trajectory.

Node X might choose to affiliate with both CCAs A and B. In that case it transmits an intent message to CCA B as in the previous case and CCA B could react in accordance to the policy defined in one of the earlier cases, listed above. If node X is granted partial affiliation by CCA B, it can distribute its inter-domain load between CCA A and CCA B. Alternatively, the CCAs may elect to 'handoff' selective nodes to each other, as in cellular networks [9] . Thus, one might in this case, be able to uniformly balance the inter-domain load in the network between the CCAs, to the extent possible. This feature and its effect on the optimal CCA trajectory are described in Section 3.2.1.

For each of the cases above, a different optimization rule may be formulated, and the choice of the rule to use is application specific. However the crucial point to note is that in spite of this added complexity, the CCAs can still use the same convex programming formulation that was presented earlier.

3.2.1 Balancing Load Between CCAs

We consider again, the scenario shown in Figure 3 as an example. Our objective is to have the nodes that are in the overlapping regions of the intersection of multiple CCA domains choose an appropriate CCA so as to evenly distribute the offered load between the individual domains. We reiterate that by 'load' we refer to the offered data load that is generated due to the packets that are to be transported via the range extension network.

We assume that the aggregate offered loads at CCA A and B are ρ_A and ρ_B packets per unit time, respectively, and that node X offers a data load of ρ. Node X can now assist in load balancing by directing the data packets that it either generates or relays to the appropriate CCA, such that the loads in the two domains are as close to each other as possible. Thus, if $\rho_A > \beta\rho_B$ where β is some preset threshold, then node X can be instructed to route a fraction α, $0 \leq \alpha \leq 1$, of its load to CCA B, while the remaining load $(1-\alpha)\rho$ is routed to CCA A. When the two CCA domains have their loads balanced, we essentially have:

$$\rho_A - \alpha\rho \approx \rho_B + (1-\alpha)\rho \tag{5}$$

If node X does not generate enough data packets to satisfy the equation, then it can route all its packets to CCA B ($\alpha = 1$) for the duration that it

[9] We are assuming that this type of CCA to CCA cooperation can be efficiently done since they will both be one hop away from a common aerial/satellite node.

remains in the overlapping region of the domains of two CCAs. Note that in this discussion, we assume that the nodes that belong to an ad hoc group will have to stay with that group. Thus only a temporary switch in CCA affiliation is permitted. However, it is easy to extend the method to consider permanent migration of nodes from one CCA's domain to that of another.

The relatively simple load-balancing technique described in the previous paragraphs can be extended to include cases in which multiple nodes are in the overlapping regions of multiple intersecting domains. In that case, in order to prevent more than one node from simultaneously initiating load transfer, the CCAs can coordinate with each other to orchestrate the procedure and instruct the node that is most heavily loaded to attempt to split its load between the two domains. This procedure may then be iteratively repeated until the loads in the two domains are balanced[10] .

4 Computational Complexity and Implementation Overhead

In implementing our algorithm, it is essential that the incurred overhead in terms of the number of control message that are exchanged is small. In addition, the algorithm should be not be computationally expensive since the time taken by the CCA to compute the optimal location must be small as compared to the time taken by the CCA to move to that location. In this section, we provide estimates of the incurred overhead and discuss the computational complexity of our algorithm.

4.1 MAC Protocol, Routing Support and Overhead

For implementing step 3, two tasks are required. First, the identity of the CCA (e.g. an IP address or MAC address) has to be made known to all the nodes. This information can be software programmed into all the nodes before network deployment, or can be broadcast to all the nodes in the domain. This broadcast is, in fact, required if there are multiple CCAs in the domain, and they are operating in some cyclic order for specified periods of time. Alternatively, each node could obtain the address of the CCA on a reactive basis, that is, if and when they have inter-domain packets to send. The overhead in this case is about the same as in discovering a specific node within a MANET.

Second, the CCA has to obtain state information from all the nodes. If we consider the Open Systems Interface (OSI) or the TCP/IP layered architecture,

[10] This procedure has been simulated and the results are discussed in Section 5.

the CCA Trajectory Algorithm can be considered as a program running at the network layer. It requires the CCA's network layer to obtain the coordinates, the offered load, and other parameters from all the other nodes in its domain. To do so, the CCA would need to broadcast a *global query message* to all the nodes in its domain, once every sampling period.

If there are n mobile nodes in the network, then a 'proactive' scheme requires that these nodes transmit an update every sampling period. Then number of routing messages in the worst case is on the order of $O(n^2)$ per sampling period (assuming flooding is employed to transport these messages) [5]. However, if the nodes that have inter-domain data to send constitute only a small fraction of the total number of nodes in the network, say α, then the number of transmissions will be reduced by this factor to $\alpha \cdot n^2$, which is again $O(n^2)$. By intelligently using the routing tables in order to relay control information, this overhead can be further reduced.

The sampling period is dependent on the rate at which the topology of the network changes. This rate depends on the density of the network and the velocity of the nodes in the network. If a table driven routing protocol is used, routing updates are required to be disseminated in order to cope with the changes in topology. The control information that is required for trajectory control may simply be 'piggy-backed' onto the routing update messages[11]. In fact this control information may be embedded in the MAC layer 'hello' or even piggy-backed onto the data payload that is routed to the CCA. The CCA would then, appropriately, extract the appropriate control information. This makes the overhead required for gathering state information for the algorithm to work to be essentially the same as, or mariginally incremental to the overhead required for enabling the underlying routing and MAC protocols. For the overhead incurred in deploying various MANET protocols, the reader is referred to [14]. We note here that in network simulations, we modified the standard Destination-Sequenced Distance-Vector Routing (DSDV) protocol [15] by piggy-backing appropriate control information onto routing update messages.

4.2 Optimization Complexity

It is important that the optimization algorithms that we describe in Section 3.1.1 *not* be computationally intensive, since our objective is to administer trajectory control of the CCA in real-time. In other words, the time that it takes for the CCA to compute its new position based on the data that it gathers during a sampling period should be less than the sampling period

[11] Most of these parameters are usually single 8-bit or 16-bit numbers, per node, so payload data length is not an issue.

itself. The computation of the trajectory involves solving a linear program numerically. It is well known that modern interior point LP solvers have a worst-case performance of $O(n^3)$ [10], where n is the number of variables in the LP. In our formulation, n would correspond to the number of nodes in the network. Thus, for a network of 100 nodes, each of which is assumed to generate inter-domain data packets, we would expect calculations on the order of 100^3 or 1 million iterations per update period. A typical update period is of the order of 0.5 seconds in our simulations. Currently available 'off-the-shelf', inexpensive microprocessors can process on the order of tens of millions, or hundreds of millions of instructions per second [16]. For a network with 1000 nodes, the complexity increases significantly, requiring 1 billion iterations per update period. In such cases, however, it is far more efficient to simply deploy more CCAs, and thus sub-divide the larger domain into smaller domains of ad hoc networks.

To prevent a possible computational bottleneck at the CCA (which is burdened with both the inter-domain communications of the ad hoc network and the computation of the optimal position), a dedicated processor may be employed on each CCA. This would enable the optimization computations and the communication operations to proceed in parallel.

5 Simulation Framework and Results

In order to evaluate the performance of our trajectory control algorithm, we used the *ns-2* network simulator, release 2.1b6 [17] as our primary simulation platform. We also used Carnegie Mellon University's code extensions for supporting wireless ad hoc networks. We employed the following implementations that the code-extensions provide: (a) the link layer and the 802.11 MAC protocol modules; (b) CBR, VBR data sources; (c) the physical channel model, which is the two-ray radio propagation model where signals experience attenuation in accordance to a fourth power path loss model, and fading/shadowing effects are not included; (d) a random waypoint motion model to govern the motion of the mobile nodes. The supporting routing protocol that was used was DSDV [15]. We also conducted simulations with appropriate modifications to the Dynamic Source Routing (DSR) algorithm [18]. Our results were similar and we omit a discussion of this due to space limitations.

We created data structures that dynamically maintained the coordinates of each node and other control information, in the form required for a LP solver. Next, the LP algorithm itself was integrated into the *ns-2* simulation framework by means of a C function call from the main program code. To implement the optimization algorithm, we used a modified version of PCx [19]. PCx is a linear program solver developed at the Optimization Technology Center at

Argonne National Laboratory. To successfully integrate this module into our simulation, we incorporated a hook in the main *ns-2* code and passed the optimization parameters to the PCx LP solver. These parameters are the data sets that have been alluded to earlier (node positions, blockage locations, the load offered by each node, priority requested, etc.). With this information, at each sampling instance, PCx invokes a variant of Mehrotra's predictor-corrector algorithm [12] with the higher order correction strategy of Gondzio [20]. This approach is among the most effective methods currently known for solving linear programs.

We were interested only in the inter-domain networking performance and consequently, all the data packets that the nodes generated in each domain were always deterministically addressed to the CCA[12]. We assume that upon receiving these packets, the CCA delivers them to the range extension network by invoking a separate set of communication protocols and mechanisms. Furthermore, we have assumed that the CCA's trajectory is governed only by 'uplink' traffic when we factor in the offered load in our optimization formulation. It is fairly straightforward to extend the same methods to take the 'downlink traffic' into account while determining the CCA trajectory.

The first case that we consider is that of equally loaded nodes generating packets of the same priority. In this case, $f(\rho_i, \tau_i) = 1$, for all i, and the problem is now equivalent to minimizing the sum of the distances from the CCA to the nodes, subject to the boundary and blockage constraints.

The results shown in Figure 4 through Figure 8 are for the following system parameters. We recall that a domain refers to the geographical area that bounds the realms of operation of a particular ad hoc group. We chose this domain to be a rectangular region of size 10,000 units by 10,000 units. The number of mobile ad hoc nodes per domain varies from 10 to 100; the nodes are assigned velocities chosen in accordance to a uniform distribution between a minimum of 0 units/s (stationary) to a maximum of 25 units/s, and move in line with the random waypoint model [7]. The CCA velocity is chosen to be at most one and a half times the maximum speed of the ad hoc nodes. All the nodes are equally loaded and generate traffic 50% of the time. The mobile nodes transmit their coordinates to the CCA once every 0.5 seconds, and the optimization calculations are also repeated with this frequency. The simulations are run for a total of 5000 seconds.

In Figure 4, simulation data was collected for three different scenarios: a CCA that is placed statically at the center of its domain; a CCA that is moving according to a random waypoint model; and finally, a CCA, the locus of whose trajectory is being updated using the optimization calculations discussed ear-

[12] Intra-domain traffic in the MANET does not affect the CCA trajectory.

lier. As expected, the results comparing the throughputs in the three different cases show that the network throughput is the best when the CCA moves in accordance with the computed optimal trajectory.

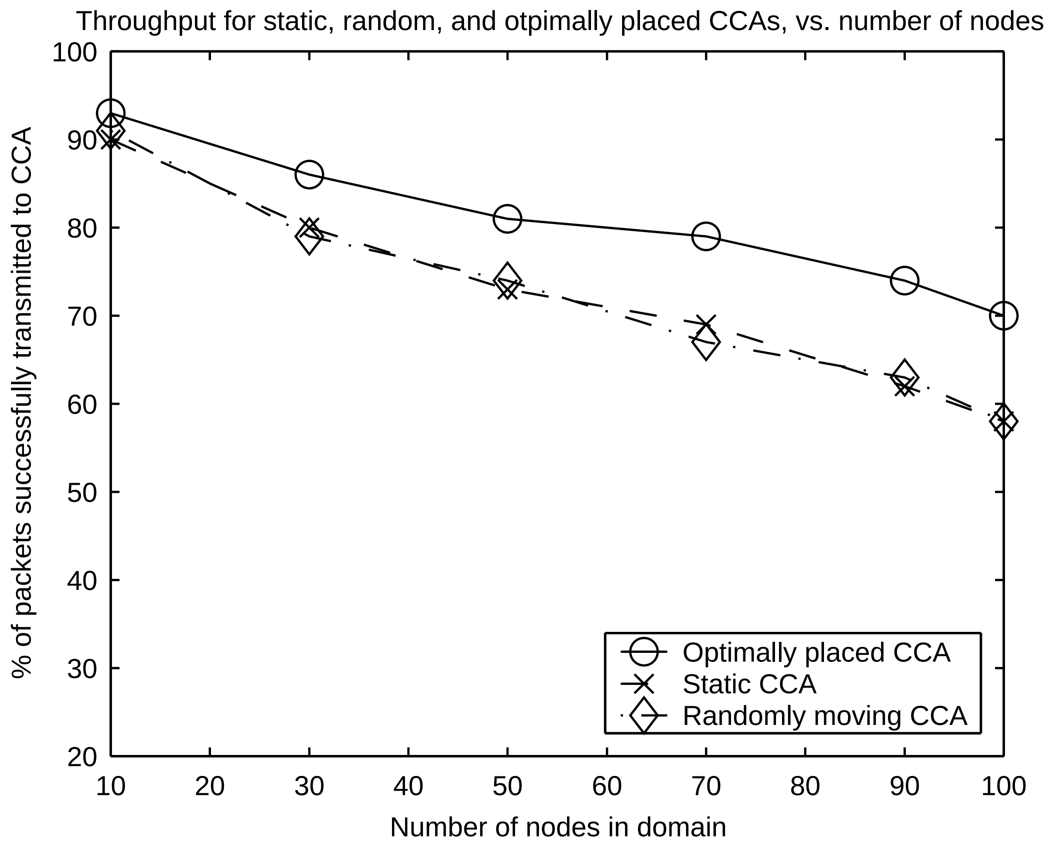

Fig. 4. Comparison of network throughput with optimally placed CCA versus statically placed or randomly moving CCA.

Note that when our algorithm is implemented, the improvement in throughput is as high as 10% per domain (ignoring inter-domain interference effects). This is the improvement seen per single CCA domain. Since a region of interest may contain several domains, the overall throughput improvement can be very significant for the network as whole.

The advantage of the dynamic CCA placement technique is much more evident when we increase the area covered by the objects blocking radio signals in each domain (Figure 5).

In Figure 5, initially, with very few blockages in the domain, the number of packets that are successfully transmitted to the CCA is roughly the same for the two cases, i.e., the case in which the CCA is statically placed, and the case in which the CCA is optimally positioned. However, as the number of blockages is increased causing the area covered by the blockages to increase, the open space in the domain decreases as a result, and the throughput drops

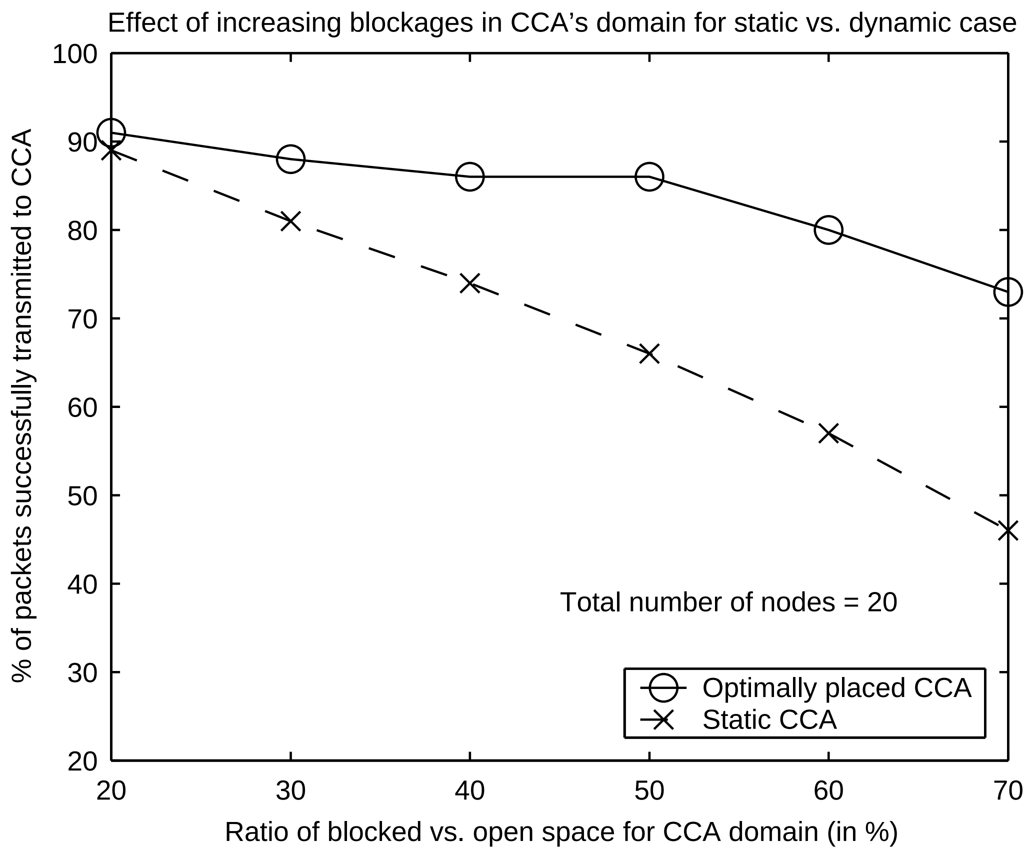

Fig. 5. Effect of blockages on the performance of the static vs. optimally placed CCAs.

dramatically if the CCA is statically placed. By comparison, when the CCA is optimally placed, we noticed the improvement in throughput to be as high as 60% per domain. In these simulations, we have assumed that the blockages permit no signal transmission through them whatsoever, so some of the nodes may be completely cut off and isolated from the CCA. For very large numbers of blockages, the performance for the dynamically placed CCA also suffers because, then, the CCA may not be able to move quickly enough to the optimal positions, due to the constraints imposed by the blockages, and by its velocity.

Next, we performed simulations in which the ad hoc nodes offered unequal loads to the network. We first compared the network throughput when the CCA position was optimized with, and then without taking the offered load into account. The cost functions in the two cases were $f(\rho_i, \tau_i) = \rho_i$, and $f(\rho_i, \tau_i) = 1$, respectively. Each ad hoc node generated data packets so as to offer a load that uniformly varied from 10% to 90% . The results are shown in Figure 6. As expected, the network performs better, in terms of throughput if the effect of the load is incorporated into the cost function, with improvements of up to 30% per domain. This may be expected, since for the case in which the loading parameter is included in the cost function, the CCA is closer to the nodes generating the bulk of the data traffic, and hence most of the packets can be delivered directly to the CCA without relays. The network performance in terms of throughput thus improves.

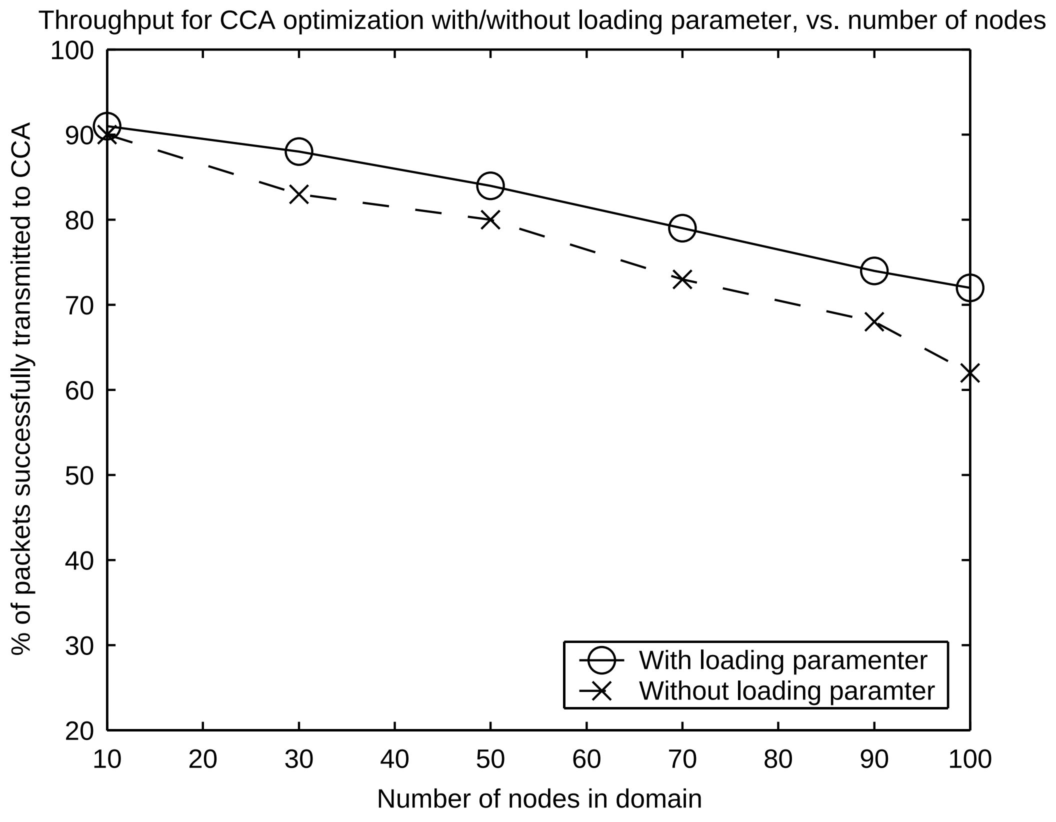

Fig. 6. Comparison of CCA optimization calculations with/without the effect of loading parameters.

We next considered the effects of choosing the optimal CCA trajectory on data packet latency. The number of nodes in a domain was kept fixed at 20, and the offered load of an arbitrarily chosen member of the ad hoc group was varied from 10% to 90% while the other nodes generated a constant load of 10%. The results are shown in Figure 7. If the CCA is optimally positioned, we expect that the data packets will experience a lower latency, on the average, than if the CCA were to be statically placed. This is because in the case in which the CCA is optimally positioned, it is closer to the heavily loaded node, and thus a large number of the data packets would no longer have to hop multiple times in order to reach the CCA, which might be the case if the CCA were to be statically placed. Thus, by positioning the CCA in accordance with the optimally computed trajectory (as opposed to positioning it statically at the center of the domain), packets are seen to experience an overall reduction in latency (by as much as 30%).

To test the performance of the network when the cost function is altered to in-

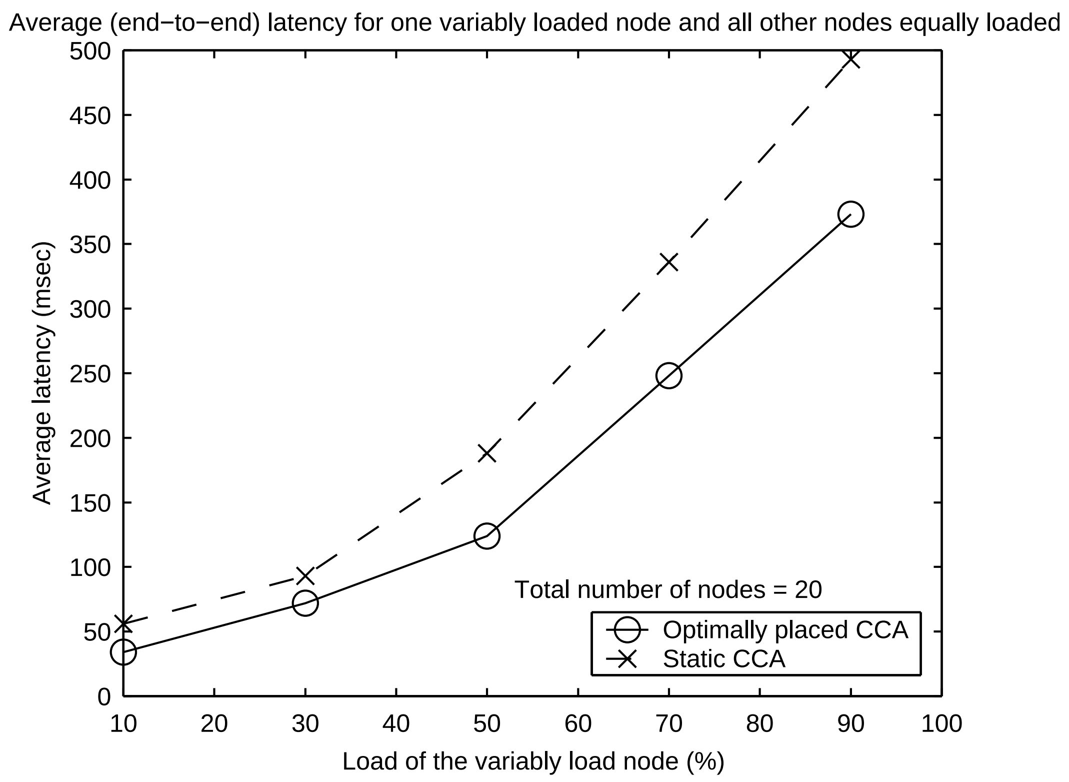

Fig. 7. The effect of loading on latency of transmitted data packets for the static and dynamically placed CCAs

corporate different priorities for different nodes[13] (Section 3.2.1), the following simulation set-up was used: 100 nodes were deployed and the load generated by each node was progressively increased from 10% to 90%. For a specific value of the offered load, half the nodes (chosen randomly in accordance to a uniform distribution) generated high priority traffic ($f(\rho_i, \tau_i) = \tau_i = 10$), whereas the remaining nodes generated low priority data traffic ($f(\rho_i, \tau_i) = \tau_i = 1$). For each of these cases, the average message latency was measured while using the CCA trajectory update algorithm with, and without, the priority class as a parameter in the cost function. The results are depicted in Figure 8.

As discussed in Section 3.1.1, the CCA now favors the nodes generating the higher priority traffic. These higher priority packets are delivered more efficiently–directly instead of via multiple hops–and with improved latency. Since the system capacity is fixed, the price that is paid, however, is that the lower priority traffic suffers increased latency and reduced throughput as a consequence. This is evident in the plots shown in Figure 8. We also note that, as expected, when the cost function in our optimization formulation does

[13] Note that a node's priority depends on the priority of the packets that it generates at the given time. This is dynamic, (akin to the offered load), and will change with time.

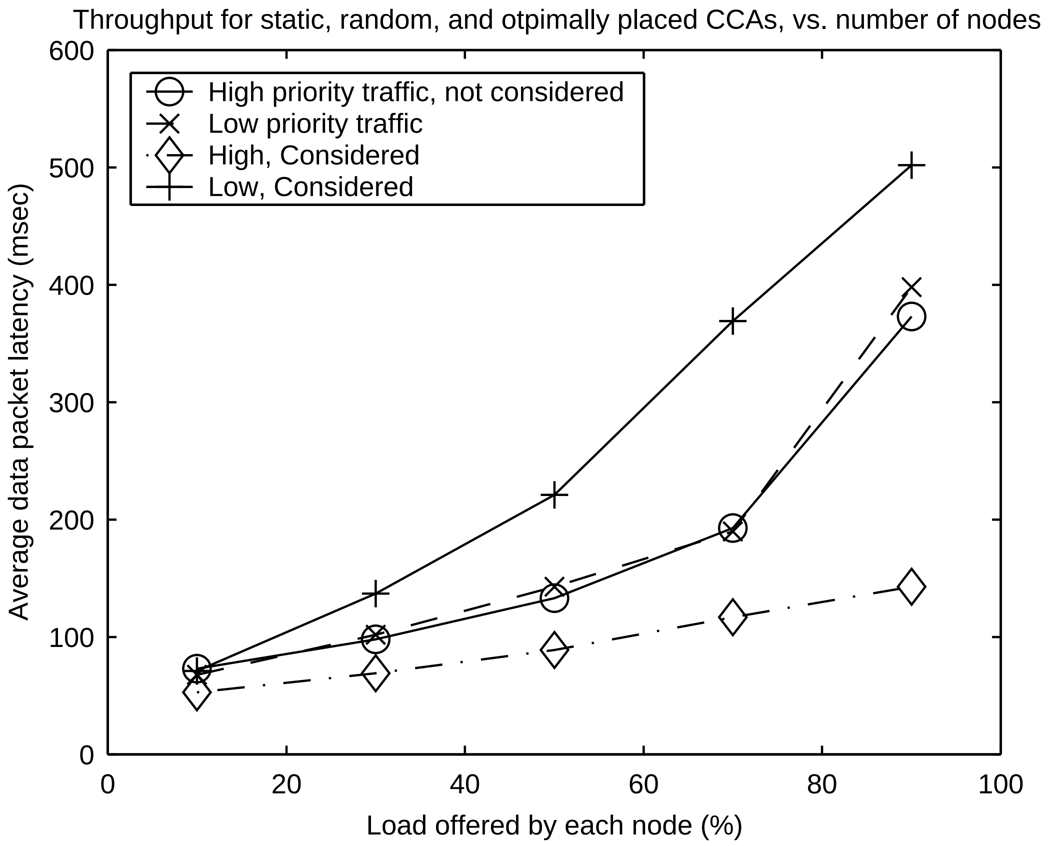

Fig. 8. Effect of including the effect of priority in the optimization cost function.

not take packet priorities into account, but considers only the coordinates and the offered load of each node, there is no significant difference in the latency incurred by the different classes of traffic.

Finally, several simulations were performed for the cases in which multiple domains intersect, and the nodes that are in the overlapping regions of the intersecting domains now have the ability to choose their affiliations among the available CCAs. For manageable simulation run-times, the set-up consisted of four domains, each with 50 ad hoc nodes forming a group, deployed over a geographical area of 50000 units x 50000 units. We assume that 20% of the geographical area is covered by harsh terrain that act as blockages that do not allow radio signals to pass through. Each node generates an offered load of 50% and the control messages that report updates are generated every 0.5s. Initially the domains are placed such that they not intersect. Each ad hoc group can move randomly over the entire region. The group is confined to a geographical area defined by a circular domain with radius 10,000 units and moves together as a single entity. Each ad hoc node within the group can move independently in accordance with a random waypoint model, within the area of its domain. When domains intersect, nodes can change affiliations as defined by the policy in place. As described in Section 3.2.1, we can choose an affiliation strategy that would force nodes that are in the overlapping region to affiliate with the least loaded domain. This would result in balancing the load among the CCAs in the intersecting domains. One might expect a gain in the throughput when we implement this strategy as compared to an affiliation strategy in which nodes are statically affiliated with domains. In our simulations, we compared the above two affiliation strategies to quantify performance enhancements that could be seen in typical scenarios. The initial load offered in each of the four domains was chosen to be different (20%, 40%, 60% 90%). As the simulation time progressed, the domains intersected, and the nodes in the overlapping regions attempted to compensate for the difference in the offered loads in the four domains by switching affiliations as described. The offered load in each

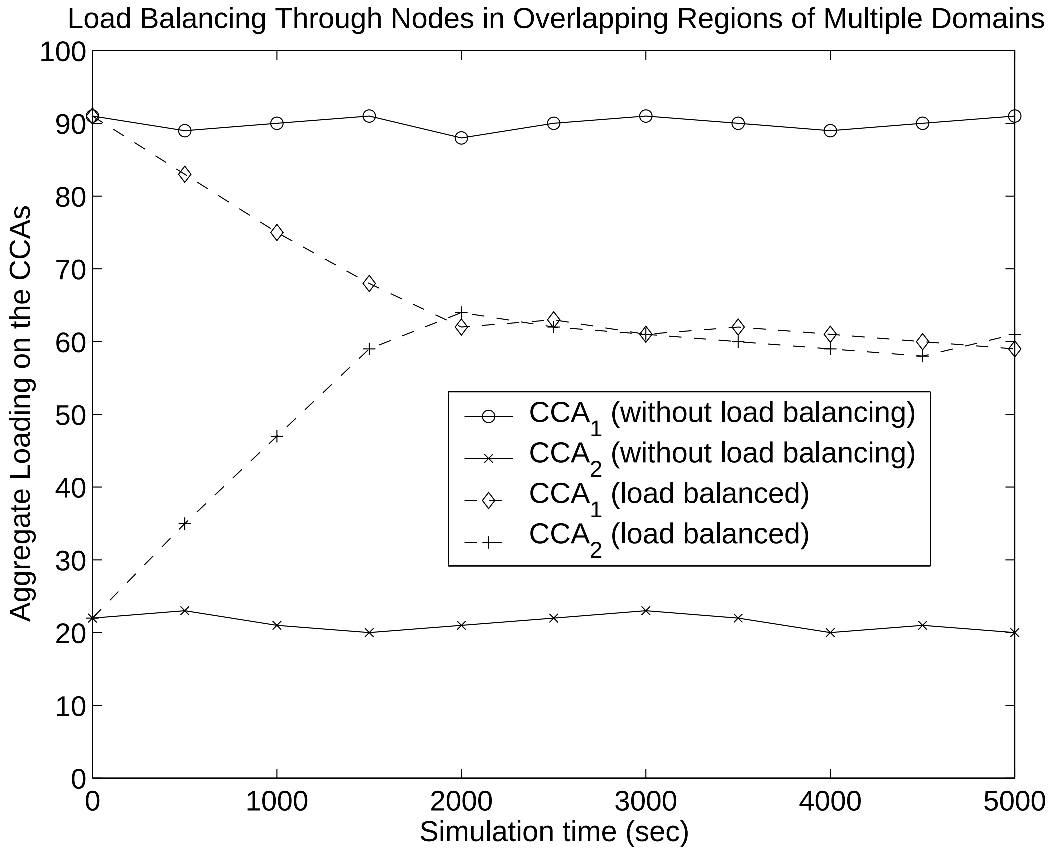

Fig. 9. Comparison of CCA loads with and without having a method to balance loads when domains intersect.

domain converged to the same value within a short period of time, after the domains intersected. This time depended on the velocity of the nodes in the domains. In Figure 9, we show the offered loads on two of the four CCA's. The loads differ to begin with, but as the domains intersect load from the heavily loaded CCA starts being switched to the lightly loaded CCA until the loads on the two CCAs are balanced. Note that without the dynamic affiliation strategy, the loads in each domain remain static, and equal to the offered load at the initiation of the simulation.

6 Conclusions and Future Work

In order to support scalability in ad hoc networks, one can envision the deployment of a range extension network that consists of airborne nodes, satellites, power transmission towers and such. In order to interface the ad hoc network with the range extension network, one approach could be to deploy gateways that we refer to as CCAs to relay data traffic from/to an ad hoc group to/from the range extension network. This is akin to providing a centralized infrastructure within the infrastructure-less ad hoc network. The objective of this work is to determine where the CCA is to be placed relative to the ad hoc group of nodes such that certain network performance metrics are optimized. This objective can be formulated as a set of convex optimization problems. By means of suitable modifications, we simplify these convex formulations such that they can be very efficiently solved by numerical methods.

We evaluate the improvements in network performance that we achieve by means of extensive simulations, where the CCA is enforced to follow the computed optimal trajectory. Simulation results indicate that the network throughput improves by about 10-15% per CCA domain[14] (an ad hoc domain consisting of about 50 nodes), if the CCA moves in accordance with the optimally computed trajectory as opposed to being static or moving in accordance to a random waypoint model. A similar improvement is seen in terms of a reduction in latency that the data packets experience. The cost function in our optimization formulation can also be appropriately modified to support better performance for high priority traffic, as compared to the performance for lower priority traffic. We also consider cases wherein the nodes have the ability to choose different CCAs for relaying their inter-domain traffic and show that this ability can be exploited to balance the loads in different domains, thereby improving performance further.

We also show that the operations that are required in order to thus define an optimal trajectory for the mobile CCAs, can be performed efficiently and quickly by most commercial, off-the-shelf micro-processors, and with little additional overhead.

One particular extension of interest, is to incorporate fault tolerance in our composite ad hoc network. In this scenario, we can envision multiple nodes in a domain, each of which is capable of acting as the CCA. In that case, should the primary CCA fail, then, according to some pre-determined rule, one of these dormant CCAs can take over as the new CCA. This extension is currently being investigated.

[14] The overall improvement would be several multiples of this value, since a region of interest will likely contain several CCA domains.

References

[1] Z. Haas, et al., Guest editorial on wireless ad hoc networks, IEEE Journal on Selected Areas in Communications 17 (8).

[2] B. M. Leiner, Goals and challenges of the darpa glomo program (global mobile information systems), IEEE Personal Communications Magazine 3 (6) (1996) 34–43.

[3] Y.-B. Ko, N. Vaidya, Using location information in wireless ad hoc network, in: Proceedings of the 49^{th}IEEE Vehicular Technology Conference, Vol. 3, 1999, pp. 1952–1956.

[4] K. Amouris, et al., A position-based multi-zone routing protocol for wide area mobile ad hoc networks, in: Proceedings of the 49^{th}IEEE Vehicular Technology Conference, Vol. 2, 1999, pp. 1365–1369.

[5] G. Karumanchi, et al., Information dissemination in partitionable mobile ad hoc networks, in: Proceedings of the 18^{th} IEEE Symposium on Reliable Distributed Systems, 1998, pp. 4–13.

[6] Y.-C. Hu, D. B. Johnson, Caching strategies in on-demand routing protocols for wireless ad hoc networks, in: Proceedings of the 6^{th} AnnualIEEE/ACM International Conference on Mobile Computing and Networking(MOBICOM 2000), ACM, 2000, pp. 231–242.

[7] D. B. Johnson, D. A. Maltz, Dynamic source routing in ad hoc wireless networks, in: T. Imielinski, H. Korth (Eds.), Mobile Computing, Kluwer Academic Publishers, 1996, Ch. 5, pp. 153–181.

[8] IEEE Local and Metropolitan Area Network Standards Committee, Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, IEEE Std 802.11-1997, Tech. rep., The Institute of Electrical and Electronics Engineers, New York (1997).

[9] C. E. Perkins, E. M. Royer, Ad-hoc on-demand distance vector routing, in: Proceedings of the 2^{nd} IEEE Workshop on Mobile Computing Systems and Applications, New Orleans, Louisiana, 1999, pp. 90–100.

[10] D. J. Bertsimas, J. N. Tsitsiklis, Introduction to Linear Optimization, Athena Scientific, Belmont, Massachusetts, 1997.

[11] D. Bertsekas, Nonlinear Programming, Athena Scientific, Belmont, Massachusetts, 1995.

[12] S. Mehrotra, On the implementation of a primal-dual interior point method, SIAM Journal on Optimization 2 (1992) 575–601.

[13] N. Jet Propulsion Laboratory, http://www.jpl.nasa.gov.

[14] E. Royer, C.-K. Toh, A review of current routing protocols for ad hoc wireless networks, IEEE Personal Communications Magazine 6 (2) (1999) 46–55.

[15] C. E. Perkins, P. Bhagwa, Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers, in: ACM SIGCOMM'94 Conference on Communications Architectures, Protocols and Applications, 1994, pp. 234–244.

[16] I. Corporation, http://www.intel.com.

[17] U.-I. S. Institure, **ns-2** network simulator, http://www.isi.edu/nsnam/ns.

[18] J. Raju, J. J. Garcia-Luna-Aceves, A comparison of on-demand and table driven routing for ad-hoc wireless networks, in: Proceedings of theIEEE International Conference on Communications(ICC 2000), Vol. 3, New Orleans, Louisiana, 2000, pp. 1702–1706.

[19] Mathematics, A. N. L. Computer Science Division, http://www.mcs.anl.gov.

[20] J. Gondzio, Multiple centrality corrections in a primal-dual method for linear programming, Computational Optimization and Applications 6 (1996) 137–156.