Facilitating Real-time Applications in VANETs through Fast Address Auto-configuration

Maria Fazio*, Claudio E. Palazzi†‡, Shirshanka Das†, Mario Gerla†
*Dipartimento di Matematica
Università di Messina, 98166, Messina, Italy
e-mail:mfazio@ingegneria.unime.it
† Computer Science Department
University of California, Los Angeles, CA 90095
e-mail:{cpalazzi|shanky|gerla}@cs.ucla.edu
‡ Dipartimento di Scienze dell'Informazione
Università di Bologna, 40126, Bologna, Italy

***Abstract*— Real-time applications are going to play a major role in Vehicular Ad-hoc Networks (VANETs). In this context, nodes' IP addresses need to be automatically configured in a very small time and with a reduced need for re-configurations. Due to the very high mobility of vehicles, however, traditional mechanisms for address auto-configuration fail to perform well. Aimed at solving this problem, we propose a novel Leader-based scheme that exploits the topology of VANETs and a distributed DHCP service to guarantee fast and stable address configuration.**

Keywords: Ad-hoc networks, VANET, address configuration.

I. Introduction

In the field of inter-vehicular communication (IVC) the emerging trend is to equip vehicles with communication capabilities in order to directly exchange information within a short communication range (i.e., DSRC/802.11p) [1].

Since depending solely on (geographically dispersed) Internet Gateways (IGs) for connectivity would lead to an intermittent and unsatisfactory Internet experience, ad-hoc networks have been proposed to extend the communication range of vehicles. In this context, car passengers will soon benefits from useful networked applications intended for their safety and also for entertainment.

As it is well known, this kind of applications generally requires a continuous connectivity among engaged players able to deliver each game event within a strict time threshold (i.e., 100-150ms) [**?**][**?**]. The same requirements for continuous connectivity and fast delivery are shared also by other entertainment applications (e.g., video/text chats, instant messaging, real-time video/audio streaming) and with other applications of prominent importance in VANETs such as safe-driving assistance. Still, to effectively facilitate these applications, a fast and efficient network configuration of cars is needed to minimize the time spent in control overhead and to ensure continuous connectivity and fast delivery to real-time (and other) applications.

Indeed, a very important topic, yet never investigated in VANETs, is represented by the address configuration. Existing VANET literature bypasses the address configuration task by assuming that nodes are configured *a priori*. However, this issue cannot be skipped so easily since neither address autoconfiguration protocols for traditional fixed networks nor solutions proposed for regular ad-hoc networks can be directly applied to VANETs [6][13][12][10][14][3]. VANETs, in fact, have unique characteristics that require a specific analysis of the problem [9]: very high mobility, theoretically infinite extension, absence of a centralized control, and intermittent connectivity through the sparse infrastructure.

To this aim, we propose here a novel automatic IP address configuration protocol named *Vehicular Address Configuration* (VAC), which is specifically designed for VANETs. In particular, VAC exploits the topology of a VANET and a distributed DHCP protocol run by dynamically elected Leader-vehicles to quickly provide unique identifiers and to reduce the occurrence of address re-configurations due to mobility.

This paper is organized as follows. In Section II we describe prominent issues in address configuration with reference to VANETs. We review solutions for address configuration in classic ad-hoc networks in Section III. Section IV describes our solution for automatic IP address configuration in VANETs. Simulation evaluation of VAC's performance is presented in Section V. Finally, Section VI concludes this paper.

II. Problem Statement: Dynamic Addressing in Vehicular Scenario

A VANET differs from usual ad-hoc networks in its vehicular environment, node distribution and movement. Even the applications run are often specifically devised to be more useful in this setting. The technical considerations in designing a VANET should reflect these features.

Vehicles are equipped with the Dedicated Short-Range Communication (DSRC) and have about the same computational and transmission capabilities. They move along a freeway and are able to communicate either directly or by using intermediate vehicles as relaying nodes. A specific ad-hoc routing protocol is used to support multi-hop packet delivery among nodes in the network. The most common

routing protocols (e.g., AODV [8][7] or DSR [4]) make use of IP addresses to identify nodes and to perform correct packet routing.

IGs can be spread along the freeway in order to provide Internet connectivity to vehicles; however, due to their mobility, vehicles pass fixed IGs in few seconds. Consequently, the IGs provide a cheap, but also intermittent and time-constrained, access to the Internet for transiting vehicles.

From the point of view of the address configuration issue, IGs cannot work as DHCP servers to assign dynamic IP addresses to vehicles. In infrastructure mode, whenever a car X passes through the communication range of an IG, it can utilize the dynamic IP address IP_X provided by that IG. The same address IP_X could be assigned again as soon as X goes out of the range of the IG. This implies that only vehicles within the coverage of an IG have unique identifiers. On the other hand, nodes that work in ad-hoc mode cannot make use of such identifiers, because the VANET expands outside the coverage of single IGs. A possible solution could be represented by a modified multi-hop DHCP that simultaneously involves several IGs and extends the DHCP service to n-hop faraway nodes. However, this approach raises other issues like IG discovery, IG information synchronization, coverage width (IGs can be more than 10 miles apart), and delays. Moreover, frequent handoffs among IGs due to high and fast mobility lead to costly IP address changes. For all these reasons, the address management system cannot reside on the fixed infrastructure along the roadside. To support ad-hoc IVC, not only a specific protocol for address management should be designed, but it should also be in the form of a distributed scheme for mobile nodes.

To perform both ad-hoc communication among vehicles and infrastructure-based communication through IGs, orthogonal channels can be used by communication devices endowed with two interfaces, one for the ad-hoc and one for the infrastructure mode. A two radio interfaces solution exploiting orthogonal channels to minimize interferences is both practical and feasible. Indeed, multiple commercial vendors are coming out with multi-band chipsets that allow communication on two or more channels [2].

Applications in ad-hoc networks are expected to be based on service discovery and data sharing mechanisms. Users take advantage from such services and data at their convenience and usually this involves communications among users few hops away. These assumptions remain true when we focus on VANETs. For instance, safe driving applications generate alert messages which are exchanged among cars close to each others.

Finally, an efficient address auto-configuration system has to be able to assign unique IP addresses in a very fast way and requiring as few re-configurations as possible. This is true for every application but becomes crucial for time sensitive ones. Think for instance to safe driving applications, where wasting even few hundreds of milliseconds in configuring addresses instead of quickly propagating alert messages could be paid in terms of human lives [**?**]. Analogously, consider entertainment applications such as interactive online games, which are known to provide amusement to players when ensuring a responsive and network-fair game event delivery [**?**]. Obviously, these conditions cannot be satisfied when some participant is not connected due to (momentary) lack of unique IP address.

III. Background

In spite of significant research effort in the general area of auto-configuration, an efficient solution for IP address configuration in VANETs is still missing. Researchers have addressed this issue with reference to generic ad-hoc networks, without considering VANET's topology and mobility model. At the same time, previous techniques for ad-hoc networks represent a fundamental background. Therefore, in this section, we present address configuration approaches in ad-hoc networks categorizing them into three groups: *Decentralized*, *Best-effort*, and *Leader-based*.

In the Decentralized approach, the auto-configuration problem is solved in a distributed fashion. A node that needs an address broadcasts a request and receives the configuration parameters through its interaction with other nodes. Even though this is the simplest solution in ad-hoc networks, it could generate large amounts of signaling traffic in large and dynamic networks with a high density of nodes such as VANETs [6][5].

Best-effort solutions do not ensure that every address is unique in the network: if users with duplicated addresses do not communicate with each other, there is no need to waste time and resources in solving the duplication. However, problems occur if nodes with duplicate addresses start to communicate; in this case, the conflict needs to be solved. Unfortunately, transmissions have to be delayed until a configuration with unique addresses is restored and this delay leads to poor performance, especially for real-time applications [13][14].

General Leader-based approaches make use of a hierarchical structure to perform the address configuration task. Some nodes in the network, called Leaders, maintain lists of addresses in use and assign new addresses to nodes that join the network. Obviously, this architecture needs efficient network merging/splitting systems with dynamic Leader dismissal/election and duplicate addresses verification when two Leaders become close to each other [12]. Moreover, especially in VANETs, communication among Leaders can be complicated by node mobility, which can break the communication path, and by the fact that such path cannot be guaranteed to be free of duplicate addresses. Specific mechanisms have hence to be introduced [10].

IV. Vehicular Address Configuration Protocol

VAC represents the first protocol for IP address configuration in VANETs. We designed VAC aiming at guaranteeing a fast and reliable address configuration service for both the initial configuration of nodes that join the network and for the later verification of duplicates through the Duplicate Address Detection (DAD) task.

VAC is a Leader-based protocol and has been designed to be able to operate even in networks with theoretically infinite extension such as VANETs. VAC organizes Leaders in

a connected chain so as to have every node in the communication range of at least one Leader. However, conversely from traditional ad-hoc solutions, each Leader needs to be aware of the presence of only few other Leaders (those in proximity).

The hierarchical organization of the network allows limiting the signal overhead for the address management tasks. Indeed, only Leaders communicate each others and maintain updated information on configured addresses in the network. Normal nodes ask Leaders for a valid IP address whenever they need to be configured and snoop Leaders' packets to catch information they need.

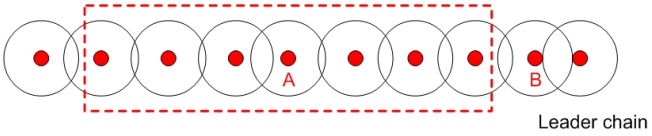

Fig. 1. Leader chain

Leaders act as servers of a distributed DHCP protocol. Each of them manages a different subset of possible addresses ($addrSet$) to serve requests for address assignment coming from normal nodes located within their communication range. The configuration of nodes is very fast because it does not suffer from delays caused by information discovery or by the delivery of configuration parameters along the network.

Many applications run on VANETs engage vehicles located within a limited area (e.g., highly interactive online games that take advantage of the short distance among players to ensure very fast game event delivery, safe-driving assistance through alert messages from/to cars in proximity). Likewise, VAC guarantees unique IP addresses within a delimited area around each Leader, called *SCOPE*.

To elaborate, the *SCOPE* of Leader A is the set of Leaders whose distance from A is less or equal to $scope$ hops. In Figure 1 we depict the *SCOPE* for Leader A considering $scope = 3$. Considering the Normal node Y that received the *IPy* address from A, *IPy* will be unique as long as Y moves within the *SCOPE* of A. If Y goes out of the *SCOPE* of A, in order to still ensure address uniqueness, Y has to ask for another address to the new Leader (for instance, to Leader B in Figure 1). Considering that the relative speed between nodes is low, changes in the address configuration due to having left the own Leader's $SCOPE$ are not frequent.

This represents an important contribution of our work since, in any case, these changes would be much more frequent if nodes (i.e., travelling cars) had to rely on fixed IGs to obtain their IP addresses. To this aim, Figure 2 shows the time duration of an IP address from when it is assigned to a node to when the node needs to be reconfigured. Three cases are analytically compared: i) a car travelling through the coverage area of a fixed IG at 60mph, ii) a car travelling through the coverage area (i.e., the *SCOPE*) of a Leader implementing VAC with $scope = 0$, and iii) a car travelling through the coverage area of Leaders implementing VAC with $scope = 4$. For all the compared cases we considered 400m as the transmission range, while in cases ii) and iii) leaders were located at regular intervals of 200m and various relative speeds were tested. Obviously, case i) presents a constant outcome (30s), while cases ii) and iii) are able to ensure much higher stability to the address configuration.

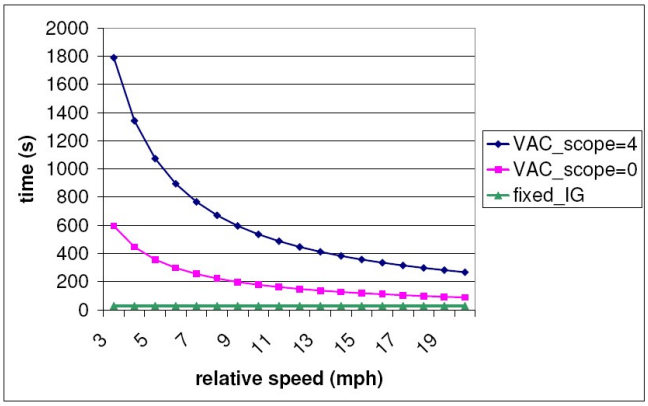

Fig. 2. Address validity time.

VAC functionalities can be grouped into two main tasks:

A. Building and maintenance of the Leader chain: how to elect Leaders and how to change them when node mobility makes it necessary;

B. Address configuration and maintenance: management of addresses that can be assigned to nodes.

In the next subsections we provide details on how VAC performs tasks *A.* and *B.*.

A. Building and Maintenance of the Leader Chain

The metric used for building the Leader chain is the distance among nodes. If the distance between two Leaders L1 and L2 exceeds a threshold *TH_max*, a node in the middle has to become a new Leader. We are sure that there is at least a node that can become Leader as the freeway scenario is characterized by high density of nodes. On the contrary, if the distance falls under a threshold *TH_min* then one between L1 and L2 will become Normal.

B. Address Configuration and Maintenance

This task is composed by three main components:

1) synchronization of address information among Leaders;

2) modified DHCP protocol to assign addresses to nodes that make a request;

3) DAD procedure to verify whether an address in the *SCOPE* ceased to be unique due to node mobility.

These components operates as follows.

1) Synchronization of address information: to carry out the address configuration of nodes, Leaders have to know which addresses are available to be assigned. For this purpose, the address space is divided into subsets ($addrSet$) and each Leader distributes addresses taking them from one of these subsets. Obviously, Leaders within the same *SCOPE* have to manage different $addrSets$. The size of the $addrSet$ depends on the width of the *SCOPE*. A large *SCOPE* (that corresponds to a high value for $scope$) implies a small $addrSets$.

The design of VAC is based on a proactive approach to synchronize Leaders' subsets of addresses. Periodically, each Leader sends in broadcast an *Hello* packet holding the $addrSet$ it can assign and the list all Leaders in its *SCOPE*. Whenever a Leader receives an *Hello* packet, it updates its view of Leaders in its *SCOPE*.

2) Modified DHCP protocol: the configuration of nodes entering the VANET is performed through a modified DHCP protocol. The node X that is not configured yet checks if there are any Leaders. This means that X gathers the Hello packets sent from Leaders in proximity for a time period *T_start*. Then, it estimates who is the nearest Leader and sends it an address request.

3) DAD procedure: the IP address received from a Leader is unique only within the *SCOPE* of the Leader. If a node X is configured with the IP_X address and it moves away from its Leader, it could get close to a different node that is using the same IP_X address in another location of the network. For this reason, nodes have to perform a DAD procedure.

Since, *Hello* packets sent by a Leader contain the list of other Leaders in its *SCOPE*, by simply listening to *Hello* packets, each Normal node is able to know when its Leader is too far and a new address is needed.

V. Simulation Results

In order to test the protocol performance, VAC has been implemented as a module for version 3.7 of the Qualnet simulator [11]. Each scenario was configured with 50 mobile nodes (cars) with a radio range of $400m$ that moved over a $15000m$x$20m$ terrain. Nodes joined the network at the point $(0.0; 10.0)$ with a certain $inter\_arrival$ time in the range $0.5 - 2s$, and moved along the Cartesian x terrain dimension with a random speed in the range $26 \pm (vel\_gap/2)\ m/s$, where vel_gap is the gap between the minimum and the maximum speed of cars in the scenario. In our simulations we used several values for vel_gap in order to test the behavior of the protocol with reference to different degrees of node mobility. Each simulation was run for $250s$ in order to have all nodes joining the network and shifting along the terrain of reference.

We have evaluated the VAC performance with reference to three parameters:

- *SCOPE* size: it settles the width of the area in which unique addresses are guaranteed. We have used the following set of values for the variable $scope$: $2, 3, 4, 5, 6$.
- vel_gap: this parameter controls the relative mobility among nodes. High vel_gap values imply higher probability that normal nodes go out of the Leader's *SCOPE* and higher instability in the Leader chain configuration. For the vel_gap parameter we have used values $5, 10, 15, 20m/s$.
- $inter\_arrival$ time: this parameter allows changing the node density in the network. We have set the $inter\_arrival$ time to $0.5, 1, 1.5, 2s$.

A. Configuration Time

The main goal of VAC is to perform a reliable address configuration service with a low configuration time. To this aim, we report in Table I the results of our simulations referring to the time on average that a node spends to configure a new address. It shows that nodes are able to configure valid addresses in less than 70ms regardless of the $SCOPE$ size ($scope$), the node density ($inter_arrival$), and the relative speeds among cars (vel_gap). This represents a very good result for it proves that VAC is suitable even for those applications that are based on a continuous stream of data where each packet has to be delivered within very strict time threshold (i.e., real-time ones).

$inter\_arrival = 1.0s$; $vel\_gap = 10m/s$

scope	**3**	**4**	**5**	**6**
t_config (s)	0.0623	0.0427	0.0559	0.0521

$scope = 4$; $vel\_gap = 10m/s$

inter_arrival (s)	**0.5**	**1**	**1.5**	**2**
t_config (s)	0.0505	0.0427	0.0458	0.0521

$scope = 4$; $inter\_arrival = 1.0s$

vel_gap (m/s)	**5**	**10**	**15**	**20**
t_config (s)	0.0405	0.0427	0.0541	0.0581

TABLE I
CONFIGURATION TIME IN SECONDS

B. Address Configuration Stability

Another important property of VAC is the stability in the node configuration obtained by limiting the number of changes in the network setting. In particular, the variable num_config represents the average number of address assignments per node (including also the very first one). When a node becomes Leader, it changes its previously assigned address into a new one in order to be consistent with its $addrSet$. Therefore, each Leader obtains an address at least twice (i.e., num_config is at least 2 for Leaders), whereas the minimum number of configurations for a normal node is just 1. We show in Figure 3-5 that the num_config value is low; in fact, each node changes address two-three times on average. In essence, VAC protocol does not cause flickering in the address configuration.

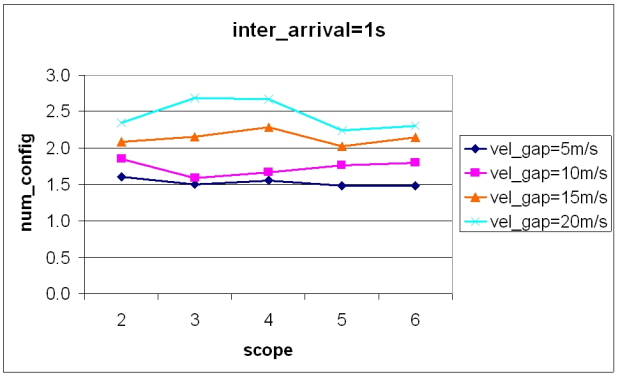

Fig. 3. Average number of address configurations of a node per $scope$ for several values of vel_gap and with constant $inter\_arrival$ time.

As shown in Figure 3, the number of configurations remains constant even if the size of the *SCOPE* changes. This happens because such size does not modify the Leader chain organization or the interaction between Normal nodes and Leader ones.

On the contrary, the num_config increases with the $inter\_arrival$ time (Figure 5). Increasing the $inter\_arrival$ period corresponds to spreading the nodes on a longer segment of terrain and, consequently, making a longer Leader chain as the chain building metric is the distance among Leaders. Therefore, higher $inter\_arrival$ times generate less dense networks with a higher ratio between Leaders and Normal nodes. Since Leaders reconfigure their addresses at least twice, the higher the percentage of Leaders, the higher the num_config value.

This is confirmed by Figure 4, which draws the total number of *Hello* packets sent with different simulative configurations. Since each Leader sends a *Hello* packet every $800ms$ and the thresholds for the Leader chain building were unchanged during the simulations, the only reason that causes an increase in the *Hello* traffic is a higher number of Leaders in the chain and hence a longer chain.

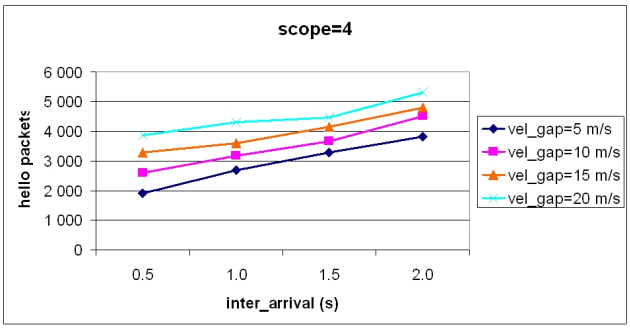

Fig. 4. Average number of *Hello* packets per $inter\_arrival$ time for several values of vel_gap and constant $scope$.

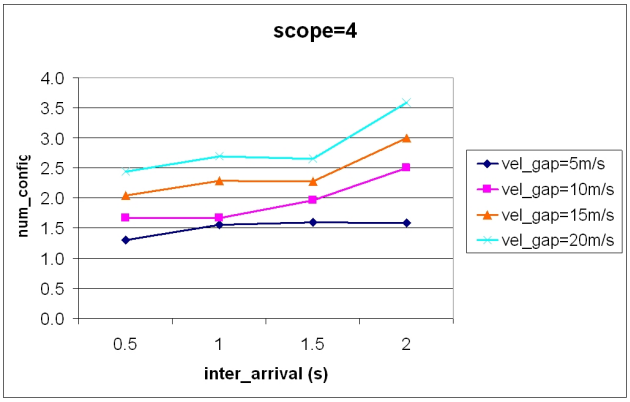

Fig. 5. Average number of address configurations of a node per $inter\_arrival$ time for several values of vel_gap and with constant $scope$.

Finally, the num_config value also increases with the vel_gap (Figure 3-5). This is due to a higher instability of the Leader chain configuration generated by higher node mobility. Indeed, with higher relative speeds among nodes, there is a higher probability that new Leaders enter the chain or current Leaders go away. Furthermore Leaders come close to each other and some of them may be forced to switch into Normal status.

VI. Conclusion

In this paper we have discussed a fundamental issue in VANETs: the IP address configuration of vehicles. The unique characteristics of vehicles preclude us from directly applying techniques developed for traditional ad-hoc networks. Consequently, we have developed VAC, an efficient protocol for the IP address configuration in VANETs. VAC is based on a network backbone in which leaders offer an enhanced DHCP service to all the other nodes in the network. Our approach guarantees a reliable configuration service that is characterized by low signaling overhead and low configuration time.

References

[1] Standard Specification for Telecommunications and Information Exchange Between Roadside and Vehicle Systems - 5GHz Band Dedicated Short Range Communications (DSRC) Medium Access Control (MAC) and Physical Layer (PHY) Specifications, 2003.

[2] EN 301. intelligent wide-band WLAN chipset. http://www.engim.com/.

[3] M Fazio, M Villari, and A. Puliafito. IP Address Autoconfiguration in Ad Hoc Networks: Design, implementation and Measurements. *Computer Networks Journal, Elsevier Science Publisher*, 50:898–920, 2006.

[4] D. B. Johnson, D. A. Maltz, Y-C. Hu, and J. C. Jetcheva. The dynamic source routing protocol for mobile ah hoc networks (dsr), July 2004. IETF Internet draft, MANET working group, draft-ietf-manet-dsr-10.txt.

[5] M. Mohsin and R. Prakash. Ip address assignment in mobile ad hoc networks. In *Proceedings of IEEE MILCOM*, September 2002.

[6] S. Nesargi and R. Prakash. MANETconf: Configuration of Hosts in a Mobile Ad Hoc Network. In *INFOCOM 2002*, New York, June 2002.

[7] C. E. Perkins, E. M. Belding-Royer, and S. R. Das. Ad hoc on-demand distance vector (aodv) routing, June 2002. IETF Internet draft, MANET working group, draft-ietf-manet-aodv-11.txt.

[8] C. E. Perkins and E. M. Royer. *Ad hoc Networking*, chapter Ad hoc On-Demand Distance Vector Routing. Addison-Wesley Publishers, 2000.

[9] A.K. Saha and D.B. Johnson. Modeling Mobility for Vehicular Ad Hoc Networks. In *poster at ACM VANET 2004*, October 2004.

[10] Y. Sun and E. M. Belding-Royer. A Study of Dynamic Addressing Techniques in Mobile Ad hoc Networks. In *Wireless Communications and Mobile Computing*, pages 315–329, April 2004.

[11] Scalable Network Technologies. Qualnet family of products. http://www.scalable-networks.com/products/qualnet.php.

[12] S. Toner and D. OMahony. Self-Organising Node Address Management in Ad-hoc Networks. In *Personal Wireless Communications (PWC 2003)*, Venice, Italy, September 23-25 2003.

[13] N. H. Vaidya. Weak duplicate address detection in mobile ad hoc networks. In *ACM International Symposium on Mobile Ad Hoc Networking and Computing*, Boston- Massachusetts, June 2002.

[14] K. Weniger. PACMAN: Passive Autoconfiguration for Mobile Ad Hoc Networks. *IEEE Journal On Selected Areas In Communications*, 23(3), March 2005.