# CodeTorrent: Content Distribution using Network Coding in VANET*

Uichin Lee, Joon-Sang Park, Joseph Yeh, Giovanni Pau, Mario Gerla
Department of Computer Science
University of California
Los Angeles, CA 90095
{uclee,jspark,jyeh,gpau,gerla}@cs.ucla.edu

## ABSTRACT

Mobile peer-to-peer systems have recently got in the limelight of the research community that is striving to build efficient and effective mobile content addressable networks. Along this line of research, we propose a network coding based file swarming protocol targeting vehicular ad hoc networks (VANET). We argue that file swarming protocols in VANET should deal with typical mobile network issues such as dynamic topology and intermittent connectivity as well as various other issues that have been disregarded in previous mobile peer-to-peer researches such as addressing, node/user density, non-cooperativeness, and unreliable channel. Through simulation, we show that the efficiency and effectiveness of our protocol allows shorter file downloading time compared to an existing VANET file swarming protocol.

## Categories and Subject Descriptors

C.2 [**Computer-Communication Networks**]: Network Architecture and Design, Routing protocols

## General Terms

Design, Performance

*This work was supported in part by the National Science Foundation under Grant No. 0520332 and the US Army under MURI award W911NF-05-1-0246. The research is continuing through participation in the International Technology Alliance sponsored by the U.S. Army Research Laboratory and the U.K. Ministry of Defence under Agreement Number W911NF-06-3-0001. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the US Army Research Laboratory, the U.S. Government, the UK Ministry of Defense, or the UK Government. The US and UK Governments are authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.


*MobiShare'06,* September 25, 2006, Los Angeles, California, USA.


## Keywords

VANET, ad hoc networks, content distribution, network coding

## 1. INTRODUCTION

Most peer-to-peer (P2P) file sharing systems (e.g., Gnutella, BitTorrent) are developed targeting wired IP networks and thus hardly work as intended in mobile ad hoc networks (MANETs) without modification. Recently, several P2P schemes targeting MANET, MANET-optimized version of existing P2P schemes as well as clean-slate designs, have been proposed. In this paper, we investigate the problem of running BitTorrent type P2P file sharing systems, i.e., *file swarming* protocols, in vehicular ad hoc networks (VANETs), MANETs consisting of cars, trucks, or any other types of vehicles on the road. We are particularly interested in VANETs since VANETs are becoming an imminent reality. In near future, every vehicle will be equipped with wireless connectivity devices [4] that enable communications with roadside objects and also with other vehicles. The development and deployment of VANETs are driven mainly by navigation safety requirements (e.g., VANET is used as reliable communication channel for *virtual* tail lamp signals between vehicles in supplement to the conventional *visual* tail lamp system) but emerging VANETs are expected to be operational over wireless LAN grade bandwidth (e.g., up to 27Mbps in DSRC standard [4]), thus allowing more applications spanning many fields from office-on-wheels to entertainment, P2P file sharing, Internet extension, etc.

MANET (as a generalized form of VANET) is characterized by highly dynamic topology. Thus, most MANET P2P protocols attempt to address problems caused by the topology dynamics through cross-layer optimization. In fact, P2P protocols are encumbered with various other characteristics of VANETs. Firstly, the wireless channel is error prone. If a protocol is designed without considering errors, the performance of the protocol in real deployment will be seriously degraded. For example, TCP connections usually die out in multihop networks with lossy channel but most P2P protocols simply assume that TCP offers reasonable bandwidth. User density should also be considered. In an urban scenario, VANETs can scale up to tens of thousands of nodes, and theoretically, all of the nodes can be users running P2P protocols. Even with any cross-layer optimization, no conventional MANET routing protocols is expected to support such big networks. Possible non-cooperative nodes are another concern. Most MANET protocols are designed

based on the assumption of node cooperativeness. Multihop routes can only be established when there are nodes willing to serve as relays for the sake of data sender. In a MANET built/maintained/owned by a single entity, such as a military tactical network or wireless mesh network, nodes can easily be forced to cooperate to achieve a common goal (e.g., providing a communication infrastructure.) But in VANETs, it is very likely that nodes are operated by different entities for their own good and thus it may not possible to force every node to cooperate each other. Lastly, IP addressing is non-trivial in VANETs. It is not clear how each node will be assigned an IP address in VANETs. Moreover, DSRC [4], a PHY/MAC standard expected to be used in future VANETs, defines the usages of random MAC address for privacy protection, which violates the static and unique MAC address assumption that every MANET routing protocol is built atop.

To remedy the issues identified above we take a holistic approach. Put another way, instead of solving each issue separately as an independent problem, we design an entirely new protocol to address all these problems at once. As we can see, the use of MANET routing protocols in VANETs give rise to most of the issues. The main argument of this paper is that, in fact, file swarming protocols in VANETs can live without MANET routing protocols. In our design we resort to single-hop communication. Multihop routes are never used and thus are not required to be maintained explicitly by any layer in the protocol stack. Rather, multihop communication is implicit. We restrict logical peers, i.e., nodes exchanging file pieces, to physical neighbors, yet data is propagated through the (overlay) network of peers of common interest, which is the basic concept of operation of P2P file sharing systems. The main problem of restricting logical peers to physical neighbors in VANETs, however, is connectivity amongst peers. It might be difficult for a node to find peers of common interest. Even though some peers are found, there is no guarantee that those peers possess useful data. The main ingredients of our design are network coding and mobility assisted data propagation (e.g. [2, 14]). The two techniques enables our design to maintain enough connectivity among peers with low overhead such that users can download files in less time than the case with existing protocols.

By network coding, we refer to the notion of performing coding operations on the contents of packets throughout a network. This notion is generally attributed to Ahlswede et al. [1], who showed the utility of the network coding for multicast. The work of Ahlswede et al. was followed by other work by Koetter and Médard [8] that showed that codes with a simple, linear structure were sufficient to achieve the capacity of multicast connections in lossless, wireline networks. This result was augmented by Ho et al. [6], who showed that a random construction of the linear codes was sufficient. The utility of such random linear code for wired P2P file sharing systems was soon realized in [5]. Our contribution in this lineage is that we realize for the first time the utility of random linear code for P2P file sharing systems in mobile networks. Our work and [5] are similar in that both use the random linear code for P2P file sharing systems. However, the protocol proposed in [5] is not expected to work properly in VANET similarly to other wired P2P protocols because of the aforementioned reasons.

The rest of this paper is organized as follows. Section 2 illustrates our network coding based file swarming protocol and we evaluate the protocol through simulation in Section 3. Section 4 presents the related work and, finally, Section 5 concludes this paper.

## 2. NETWORK CODING BASED FILE SWARMING PROTOCOL

A node which intends to share a file, a *seed* node, creates and broadcasts to its 1-hop neighbor the *description* of the file. Similar to the torrent file in BitTorrent protocol, a description contains, for example, identification number, name, size, number of pieces, etc. We simply assume that each file can be uniquely identified with an identification number (*fileid*) during the time period in which every node interested in the file completes its downloading.

At the seed node, a file $F$ is divided into $n$ pieces $\mathbf{p}_1$, $\mathbf{p}_2$, ..., $\mathbf{p}_n$. In our protocol, nodes exchange *coded frames* instead of file pieces. We define a coded frame $\mathbf{c}$ to be a linear combination of file piece $\mathbf{p}_k$'s. That is, $\mathbf{c} = \sum_{k=1}^{n} e_k \mathbf{p}_k$ where $e_k$ is a certain element in a certain finite field $\mathbb{F}$ over which every arithmetic operation is. File piece $\mathbf{p}$'s and coded frame $\mathbf{c}$'s are also regarded as vectors over $\mathbb{F}$. Whenever the seed node is requested to exchange a coded frame, the node transmits a newly generated a coded frame $\mathbf{c}$ and when generating $\mathbf{c}$, each $e_k$ is drawn randomly from $\mathbb{F}$, hence the name of random linear coding. In the header of a coded frame, the *encoding vector* $\mathbf{e} = [e_1 \ ... \ e_n]$ is stored for the purpose of later *decoding*. Throughout this paper, we abuse lowercase boldface letters to denote vectors, frames, or packets, uppercase letters to denote matrices, italics to denote variables or fields in the packet header.

A node learns of a file from receiving the file's description transmitted from neighbors. If the node finds the file interesting, it broadcast a request containing *fileid* of the file. Upon receiving such a request, every node possessing any file piece or coded frame of the requested file responds with a newly generated coded frame. A node keeps requesting neighbors to send coded frames until it collects $n$ coded frames carrying encoding vectors that are linearly independent of each other.

Whenever requested, every node, not just the seed node, generates on-the-fly and transmits a new coded frame. The generation of a code frames on non-seed nodes is through basically the same process that the seed node has undergone to generate a coded frame, i.e., generating a random linear combination of coded frames available in local memory. Note that though the frames in local memory are coded ones thus the re-encoded frame $\acute{\mathbf{c}} = \sum_{k=1}^{rnk} \acute{e}_k \mathbf{c}_k$ is tagged with the encoding vector $\acute{\mathbf{e}} = \sum_{k=1}^{rnk} \acute{e}_k \mathbf{e}_k$ where $\mathbf{c}_k$ and $\mathbf{e}_k$ is a coded frame in local memory and the encoding vector prefixed to $\mathbf{c}_k$ respectively. $rnk$ is the number of $\mathbf{c}_k$'s found in local memory. When encoding, each $\acute{e}_k$ is drawn uniformly from $\mathbb{F}$.

To recover $n$ file pieces $\mathbf{p}_1$, $\mathbf{p}_2$, ..., $\mathbf{p}_n$, a node must collect more than $n$ coded frames carrying encoding vectors that are linearly independent of each other. Let $\mathbf{c}_k$ be a coded frame, $\mathbf{e}_k$ be the encoding vector prefixed to $\mathbf{c}_k$, and $\mathbf{p}_k$ be a file piece to be decoded and recovered where $k = 1$, ..., *blocksize*. Further, let $\mathbf{E}^\intercal = [\mathbf{e}_1^\intercal \ ... \ \mathbf{e}_{blocksize}^\intercal]$, $\mathbf{C}^\intercal = [\mathbf{c}_1^\intercal \ ... \ \mathbf{c}_{blocksize}^\intercal]$, and $\mathbf{P}^\intercal = [\mathbf{p}_1^\intercal \ ... \ \mathbf{p}_n^\intercal]$ where superscript $\intercal$ denotes the transpose operation, then conceptually $\mathbf{P} = \mathbf{E}^{-1}\mathbf{C}$, which obtains the original file pieces. Note that all

$\mathbf{e}_k$'s must be linearly independent to be able to invert $\mathbf{E}$.

Not only the seed node of a file also every node which possesses any coded frame of the file and willing to share them periodically broadcasts (at a very low rate) to its 1-hop neighbors the description of the file. If a node has multiple files to share, multiple descriptions are packed into the least number of packets that can carry all of them and then transmitted.

A request of coded frames may be accompanied by the *nullspace vector* which is a vector in the nullspace spanned by all encoding vectors of the frames stored in the local memory of the requesting node. On reception of such a request, a node transmits a coded frame only if there is in its local memory a frame with the encoding vector that is not orthogonal to the nullspace vector received with the request.

Every node promiscuously listens to packets, i.e., a node receives a specific packet even the node is not the designated receiver, so that it can use them if possible. A node always overhears the packets carrying coded frames and treats the overhead ones as the coded frames transmitted specifically to the node. If an overheard coded frame is linearly independent of the coded frames in local memory, then a node stores it.

Since every transmission is MAC/link layer broadcasting, a small random amount of wait time before each transmission called broadcast jitter is applied to reduce collisions. Without broadcast jitter, MAC/link layer broadcasting suffers severely from the hidden terminal problem.

# 3. EVALUATION

In this section we evaluate CodeTorrent through simulations using Qualnet [13]. In the simulations, we use IEEE 802.11b PHY/MAC with 2Mbps data rate and Real-Track (RT) mobility model [11]. RT permits to model vehicle mobility in an urban environment more realistically than other simpler and more widely used mobility models such as Random Waypoint (RWP) by restricting the areas where nodes can appear (e.g., roads). The road map input to RT model is shown in Figure 1, a street map of 2,400$m\times$ 2,400$m$ $Westwood$ area in the vicinity of the UCLA campus.

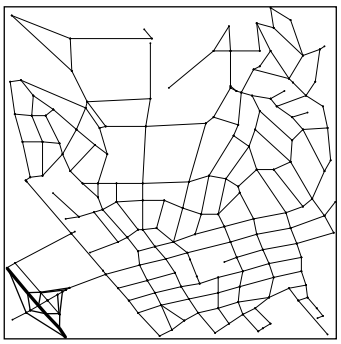

**Figure 1: Westwood area in the vicinity of UCLA**

A fraction of nodes (denoted as popularity) in the network is interested in downloading the same 1MB file. There is a special type of node called AP which possesses the complete file at the beginning of the simulation. Three static APs are randomly positioned on the roadside in the area. The 1MB file is divided into 4KB pieces. Thus, the total number of pieces is 250. A piece is transferred using four 1 KB packets. In CodeTorrent, a peer must acquire 250 linearly independent coded pieces to decode the file.

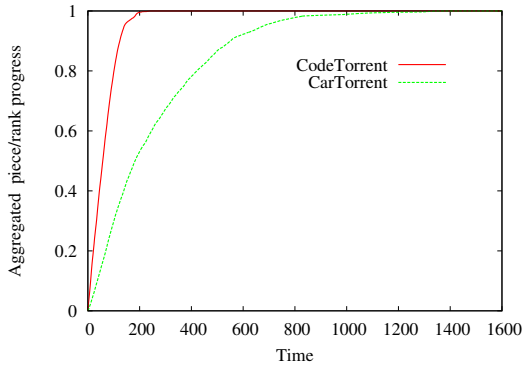


**Figure 2: Aggregated downloading progress (200 nodes moving with the maximum speed of 20 m/s. The popularity index is set to 40% meaning that the number of interested nodes is 80 nodes.)**

In CarTorrent, we use UDP to transfer data packets. As the underlying routing protocol, we use AODV. We limit the scope of the gossip packets to 3 hops as proposed in the original design [10]. We also limit the TTL value of RREQ in AODV to 3 hops. Each node initiates piece downloading either periodically (i.e., every 0.5 second) or upon receiving a new gossip packet. Successful downloading of a piece will also initiate downloading. Piece availability gossiping is carried out for every 5 seconds. We use a probabilistic gossiping: uninterested and interested nodes forwarded the gossiping packets with probability 0.1 and 0.8 respectively.

Similarly, CodeTorrent uses UDP to transfer packets to its neighbors. CodeTorrent does not use any underlying routing protocol, because it only relies on single hop unicast with overhearing. The $2^8$ field is used for coding. Thus, the size of encoding vector is 250B, 6% overhead over the payload.

We define the download "delay" to be the elapsed time for a node to collect all 250 pieces for CarTorrent or linearly independent coded pieces for CodeTorrent. The given metric is evaluated with various configurations, i.e., as a function of node density, maximum speed, and fraction of interested nodes.

## 3.1 Comparison of Download Delay

First, we contrast the download delay of CodeTorrent to that of CarTorrent in a specific setting to show the performance benefit of CodeTorrent over CarTorrent. The aggregated downloading progress (cumulative histogram with slot size of 2 seconds) is shown in Figure 2. For a given time slot (x-axis), the figure shows the average fraction of pieces collected by 80 nodes and the averaged fraction of linearly independent coded pieces collected by 80 nodes for CarTorrent and CodeTorrent respectively. The figure shows that CodeTorrent significantly expedites the overall progress compared to CarTorrent.

Figure 3 shows the histogram of download delays for both protocols with a slot size of 20 seconds. In CodeTorrent, nodes collectively help each other to distribute coded pieces using network coding (i.e., algebraic mixing). At the second time slot, i.e., [20,40), we see that around 6 nodes become

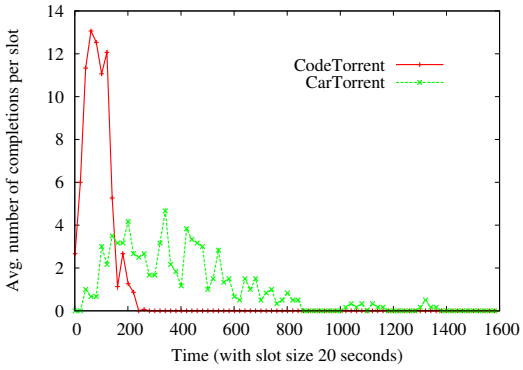


**Figure 3: Histogram of download delays**

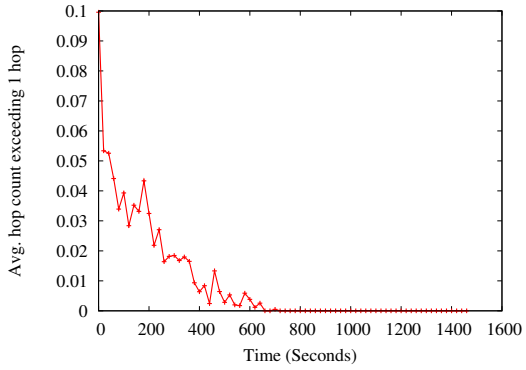


**Figure 4: Average hop count histogram for multi-hop pulling in CarTorrent**

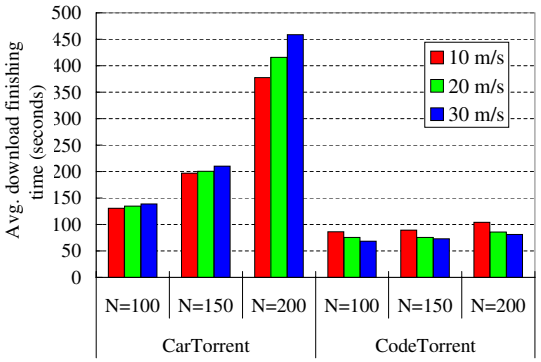


**Figure 5: Impact of mobility on average download delay**

seeds (a node becomes a seed when it completes downloading of the shared contents), which is followed by a burst of about 11 new seeds in the next slot. As the number of seeds increases in the network, the usefulness of random coded packet increases and thus, this further shortens downloading time. This confirms the benefits of network coding, which was also observed in Avalanche in the wired environment [5].

On the other hand, CarTorrent does not show such burst births of seeds, but seeds are born rather gradually due to the competition among nodes to secure downloading bandwidth. For example, after receiving a gossip packet from nodes or APs, in the worst case, 80 interested nodes start requesting pieces all at the same time. Nodes located far away from the source needs to go for muti-hop. As a result, a large number of nodes start to setting up connections toward the originator. This crowd effect causes severe channel contention, thus resulting in performance degradation as shown in the Figure 3. The first download completion happened at the third slot, [41-50), and the maximum birth rate of seeds[1] was always less than 5. To show the behavior of multi-hop pulling, in Figure 4 we show the histogram of average hop count exceeding 1 hop with a slot size of 20 seconds. The figure clearly states that since the availability of a random piece increases as time passes, the average hop count gradually decreases. Multi-hop pulling was continued till 700 seconds by which about 93% of interested nodes became seeds. As of the 700 second mark, nodes fetched a random piece only from their neighbors. Note that CarTorrent uses a closest-rarest first strategy for piece selection.

## 3.2 Impact of mobility

Next, we investigate the impact of mobility on the download delay. The average download delay as a function of node speed with various node densities is illustrated in Figure 5. We only present the results for the popularity 40% case since the results for other popularity indices show similar trends. The figure shows that in CarTorrent, as the number of nodes increases, the performance gradually degrades. For a given popularity index, the increased number of nodes means that we have increased number of interested nodes; e.g., N=100 and 200 have 40 and 80 interested nodes respectively. As the number of interested nodes increases, the overhead of an underlying routing protocol and gossiping becomes problematic. Fast mobility will induce more route errors especially when the number of interested nodes is large. For instance, when N=200, the average number of route error messages (RERRs) increases from 83.1 to 134.6 when the maximum speed increases from 10 to 30 m/s. Such routes errors will re-initiate route discovery (i.e., RREQ) and will consequently worsen the network congestion. Another important factor attributing to such congestion is the periodical gossiping. Our simulations constrain that a gossiping packet can travel up to 3 hops, and the gossiping period is fixed to 5 seconds. The network congestion is inevitable as the number of nodes participating gossiping increases. Moreover, gossiping period must be adjusted according to mobility in order to accurately choose the closest node (i.e., closest-rarest first selection):, i.e., the higher the mobility, the more the frequent the advertisements. However, this will further exacerbate the network congestion, thus resulting performance degradation.

In contrast, the average download delay of CodeTorrent decreases as mobility increases. Since CodeTorrent is based on single hop data pulling and overhearing, mobility plays an important role such that data dissemination latency could be reduced with increased mobility. For ease of an explanation, let us imagine two nodes traveling along the same path without any other contacts until they reach the end of the path. After exchanging useful information at the beginning, the remaining contact period will be useless to each other. We realize that the useless period can only be shortened when we increase their mobility. As shown in Figure

[1]Number of newly born seeds for a given slot.

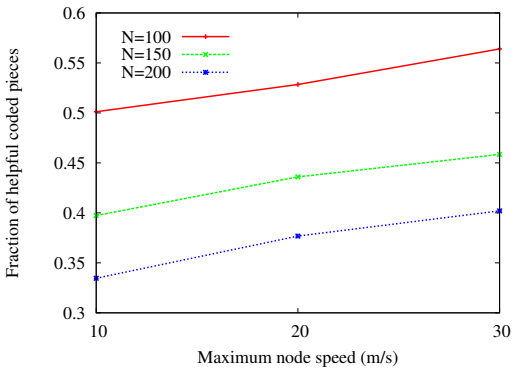


**Figure 6: Helpfulness as a function of node speed**

5, this "mobility"-based mixing on top of algebraic mixing through network coding can further reduces the delay. To support this observation, we present the average fraction of helpful coded pieces (i.e., having a linearly independent code vector) that are pulled or overheard from one's neighbors in Figure 6. As the average number of neighbors increases, it is more probable that a node overhears unhelpful coded pieces from its neighbors. For example, in a static scenario a set of nodes located in between two groups as forwarders, will receive more linearly dependent coded pieces when the size of the target group increases.[2] If nodes are mobile, this can be alleviated, and thus, the helpfulness improves with increased mobility. One caveat is that if the mobility is too high, the contact period will be too short to exchange a piece, and thus, this will adversely affect the performance. We are currently developing a mathematical model to quantify the impact of mobility.

## 4. RELATED WORK

Content sharing in MANETs are roughly categorized based on whether a protocol uses a mobility-assist or cross-layer technique. A mobility-assist protocol basically utilizes node mobility to disseminate/retrieve content or index. 7DS [12] aims at sharing web content among nodes based on a high locality of information access within a geographic area, even without Internet connectivity. A node can pull and carry content of interest from its neighbors, thus diffusing content into the network. In Passive Distributed Index (PDI) [9], mobility is exploited for disseminating and maintaining a distributed index of shared content. Basic operations of CodeTorrent are quite similar to previous approaches. However, CodeTorrent is designed to provide a BitTorrent style content distribution with network coding as proposed in [5]. Our focus in this paper is to analyze the impact of mobility on the performance of content sharing with network coding.

Cross-layer techniques incorporate routing layer for content sharing and indexing. Most protocols have been focused on overcoming the discrepancy between a logical overlay and a physical topology of mobile nodes. For example, XL-Gnutella [3] maps the logical overlay neighbors to physical neighbors. CarTorrent [10], a BitTorrent style content sharing protocol in wireless networks, uses the proximity-driven piece selection which is known to perform better than the rarest first piece selection. Similarly, ORION [7] builds an on-demand content-based overlay, closely matching the topology of an underlying network. Unlike these approaches, CodeTorrent mainly considers a dynamically changing topology and intermittent connectivity due to high mobility in VANET as well as various other issues that have been disregarded in previous mobile peer-to-peer researches such as addressing, node/user density, non-cooperativeness, and unreliable channel.

[2]Assuming that they continue to forward pieces to the next group.

## 5. CONCLUSION

In this paper, we proposed a network coding based file swarming protocol. We note that this paper presented a work in progress and preliminary results. We have not explored all the parameters and optimization opportunities that the proposed protocol allows. Rather, we kept the protocol in the simplest form for clearer presentation of the main idea. Our approach does not completely resolve the issues arising in VANET P2P systems but is a simple way to mitigate the problems.

## 6. REFERENCES

[1] R. Ahlswede, N. Cai, S.-Y. R. Li, and R. W. Yeung. Network information flow. 46(4):1204–1216, July 2000.

[2] F. Bai and A. Helmy. Impact of Mobility on Mobility-Assisted Information Diffusion (MAID) Protocols. Technical report, USC, July 2005.

[3] M. Conti, E. Gregori, and G. Turi. A Cross-Layer Optimization of Gnutella for Mobile Ad hoc Networks. In *MobiHoc'05*, Illinois, USA, May 2005.

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

[5] C. Gkantsidis and P. Rodriguez. Network coding for large scale content distribution. In *Prof. IEEE INFOCOM 2005*.

[6] T. Ho, M. Médard, R. Koetter, D. R. Karger, M. Effros, J. Shi, and B. Leong. A random linear network coding approach to multicast. Submitted to *IEEE Trans. Inform. Theory*.

[7] A. Klemm, C. Lindemann, and O. Waldhors. A special-purpose peer-to-peer file sharing system for mobile ad hoc networks. In *Proc. IEEE Vehicular Technology Conference, 2003. VTC 2003-Fall.*

[8] R. Koetter and M. Médard. An algebraic approach to network coding. 11(5):782–795, Oct. 2003.

[9] C. Lindemann and O. Waldhors. A distributed search service for peerto-peer file sharing in mobile applications. In *Proc. 2nd IEEE Conf. on Peer-to-Peer Computing (P2P 2002)*.

[10] A. Nandan, S. Das, G. Pau, M. Sanadidi, and M. Gerla. Cooperative downloading in vehicular ad hoc wireless networks. In *Proc. International Conference on Wireless On demand Network Systems and Services*, 2005.

[11] A. Nandan, S. Das, S. Tewari, M. Gerla, and L. Klienrock. AdTorrent: Delivering Location Cognizant Advertisements to Car Networks. In *WONS'06*, Les Menuires, France, Jan. 2006.

[12] M. Papadopouli and H. Schulzrinne. Effects of power conservation, wireless coverage and cooperation on data dissemination among mobile devices. In *Proc. ACM Symp. on Mobile Ad Hoc Networking and Computing (MOBIHOC 2001)*.

[13] Scalable Networks. `http://www.scalable-networks.com`.

[14] A. Vahdat and D. Becker. Epidemic Routing for Partially-Connected Ad Hoc Networks. Technical Report CS-200006, Duke University, Apr. 2000.