Co-operative Downloading in Vehicular Ad-hoc Wireless Networks

Alok Nandan, Shirshanka Das, Giovanni Pau, Mario Gerla and M.Y. Sanadidi
UCLA, Network Research Laboratory, Los Angeles, CA, USA
{alok,shanky,gpau,gerla,medy}@cs.ucla.edu

Abstract

Increasing need for people to be "connected"; while at the same time remain as mobile as ever poses several interesting issues in wireless networks. It is conceivable in the near-future that wireless "hotspots" experience flash crowds-like traffic arrival pattern. A common phenomena in the Internet today characterized by sudden and unpredicted increase in popularity of on-line content.

In this paper, we propose SPAWN, a cooperative strategy for content delivery and sharing in future vehicular networks. We study the issues involved in using such a strategy from the standpoint of Vehicular Ad-Hoc networks. In particular, we show that not only content server but also wireless access network load reduction is critical. We propose a "communication efficient" swarming protocol which uses a gossip mechanism that leverages the inherent broadcast nature of the wireless medium, and a piece-selection strategy that takes proximity into account in decisions to exchange pieces. We show through simulation that gossip incorporates location-awareness into peer selection, while incurring low messaging overhead, and consequently enhancing the swarming protocol performance. We develop an analytical model to characterize the performance of SPAWN.

1. Introduction

Future vehicular networks are expected to deploy short-range communication technology for inter-vehicle communications. In addition to vehicle-vehicle communication, users will be interested in accessing the multimedia-rich Internet from within the vehicular network. Conventional client-server approaches in the face of intermittent connectivity would experience degraded performance. A new paradigm in content delivery on the Internet using peer-peer swarming protocols is emerging. The goal of the Internet swarming protocols is to reduce the load on content servers.

Peer-to-peer networking gives ordinary users much more power than they can ordinarily obtain on their own. This power comes from users acting as content providers, rather than restricting themselves to their traditional role as consumers. Increasing popularity of open source projects and the consequent demand when a new release is out, (e.g.downloading the latest ISO images of Linux distributions in the first few weeks of its release) is overwhelming. The traditional solution is to add mirror sites or install a distributed hosting service like Akamai. However, there is a significant cost associated with establishing a content distribution network or establishing a global pool of mirrors. Thus the challenge to content providers is scaling their content delivery to large audiences. To address this problem, *swarming protocols* such as BitTorrent [5] and Slurpie [8] have been proposed. Their goals are scalable and economic content delivery by combining peer-to-peer networking with parallel downloads.

Swarming is a peer-peer content delivery mechanism that utilizes parallel download among a mesh of co-operating peers. Scalability is achieved since the system capacity increases with the number of peers participating in the system. The design space of swarming protocols is large and there are many dynamics involved. Recently, some models have been proposed to understand such dynamics and evaluate the scalability of these swarming protocols [1, 2].

Limited access to the Internet while driving [4] contrasts with the ever increasing dependence on Internet services. This motivates a compelling application of Co-operative Networking in the Vehicular Ad-Hoc network. The Ad Hoc network *extends* and *complements* the Internet. Wireless communication is inherently broadcast in nature, i.e. many nodes in the transmission range of a sender may receive a transmission from the sender. This drives us to explore different design paradigms from the ones used in typical wired settings.

In Ad-Hoc nets, every node acts as a router for other nodes in a multi-hop wireless scenario. Thus, cooperation at the network level can be resonated at the application level. For instance, a node that is forwarding some packet for its neighbor may find the data in the packet useful. It has been proposed in [6] to cache data for later use by other nodes in the vicinity using a cooperative mechanism. Consider a Vehicular Ad-Hoc Network with short-range communication technology. Given an average speed of 50 miles per hour and a gateway radio range of 500 meters, a simple calculation gives a car a transmission window to and from a fixed Internet access point of the order of a minute at the most. Taking into account contention from other cars, there may not be enough bandwidth to allow each user to download email, songs, as well as browse multimedia rich web-sites in the short time that they are connected to the gateway. It has been shown using measurements with IEEE $802.11b$ access points that the whole process of network attachment detection, IP auto-configuration(using DHCP) and the authentication takes 15 seconds. Another practical issue is that on intercity highways, the gateways will be hosted by gas stations and food concessions, and thus will be less frequent, say every 5-10 miles.

The idea of *Co-operative Networking* was first proposed to handle "flash crowds" on the Internet, where end-hosts cooperate to improve the overall network performance [9]. The gateway in our architecture would experience flash crowd arrival patterns. As we shall see, the high mobility of nodes in Vehicular Wireless Ad-Hoc networks coupled with the intermittent connectivity to the Internet provides an incentive for individual nodes to *cooperate* while accessing the Internet to achieve some level of seamless connectivity.

For the above reasons, an interesting problem is the design of cooperative protocols to improve client perceived performance of the vehicular network as a whole. The key contributions of our *SPAWN* protocol are as follows: (1) a gossip mechanism to propagate content availability information, (2) a proximity driven content selection strategy(which takes into account the fact that TCP throughput degrades over multi-hop wireless connection) and (3) leveraging the broadcast nature of wireless networks to reduce redundant message transmission. We prove the efficacy of our protocol using simple deterministic models for service capacity of the peer-peer network in an ad-hoc setting.

The rest of the paper is organized as follows. Section 2 states the assumptions of the Vehicular communication architecture in which our swarming protocol works. We then describe the cooperative downloading mechanism used by *SPAWN* in Section 2.2. Section 3 provides a simple analytical model to prove the efficacy of our protocol. Section 4 details our evaluation of the performance of *SPAWN* using simulation. Section 5 gives a brief overview of BitTorrent, a popular swarming protocol in the Internet and outlines some related work. Finally, Section 6 concludes the paper.

2. SPAWN: Swarming Protocol for vehicular Ad-Hoc Networks

Before presenting the *SPAWN* protocol, we define more accurately our network model and specify the problem using that model.

2.1. Preliminaries

The network consists of a set of N nodes with same computation and transmission capabilities, communicating through bidirectional wireless links between each other, this is the infrastructure-less ad-hoc mode of operation. There are wireless gateways at regular intervals providing access to the rest of the Internet using infrastructure support(either wired or multi-hop wireless). A unicast routing protocol is available to support packet transmissions between the network nodes(we assume each node is running AODV in this paper). Nodes may or may not run the peer-peer application protocol. Hence, P the size of the peer-peer network is such that, $P \subset N$ and is established on top of this vehicular network.

In addition, we assume a CSMA/CA MAC layer protocol(IEEE 802.11a) that provides RTS/CTS-Data/ACK handshake sequence for each transmission. Nodes use TCP for reliable transfer of data and UDP for dissemination of *gossip* messages for content availability. The information unit for the swarming protocol is a *chunk*. It includes data packets as well as gossip messages. However, the packet, is the unit for network layer. We assume each node is reachable to every other node.

The problem is to design an application level protocol for vehicular ad hoc networks that disseminates data over this network in an efficient and scalable fashion and improves client perceived performance in the light of intermittent connectivity. Specifically we intend to, (a) Design a swarming protocol for network traffic that *scales* in a vehicular wireless and handles robustly the high churn rates of nodes;(b) Understand how the underlying network model affects the *service capacity* of the swarming protocols and analyze the feasibility of co-operative networking in vehicular network scenarios.

Our vehicular wireless architecture is composed of two kinds of communications, namely, vehicle-vehicle and vehicle-gateway. Dedicated Short-Range Communication(DSRC) [16] is a short to medium range communication technology operating in the 5.9 GHz range. The Standards Committee E17.51 endorses a variation of the IEEE 802.11a MAC called the IEEE 802.11a R/A(Roadside Applications) for the DSRC link [16]. DSRC allows two kinds of modes of operation: (1) Ad-hoc mode characterized by distributed multi-hop networking(vehicle-vehicle), (2) Infrastructure mode characterized by a centralized mobile one-hop network(vehicle-gateway). The two scenarios are shown in Figure 1. In order to access the infrastructure-network, vehicles would need a wireless connection to a gateway. This can be done using the DSRC system. These gateways would be installed on the freeways at regular intervals like 5-10 miles apart. We should point out that this connectivity is intermittent at best. The incentive for co-operation lies in this connectivity architecture.

This model of vehicular network is similar to the ”Infostation” model as proposed earlier in [12]. However there are some key differences in terms of data dissemination which we will enumerate in section 5.

The characteristics of the vehicular wireless architecture and its differences with the traditional wired always-connected model motivate the need to revisit the design of swarming protocols which were designed for the wired infrastructure. We propose *SPAWN* which builds on the fundamental mechanisms of partial downloading and sharing of content in *BitTorrent* but adapts to the wireless scenario by using different mechanisms for peer discovery, selection and content delivery.

2.2. The Protocol

SPAWN has the same generic structure of any swarming protocol. Peers downloading a file form a mesh and exchange pieces of the file amongst themselves. However the wireless setting of *SPAWN*, characterized by limited capacity, intermittent connectivity and high degree of churn in nodes requires it to adapt in specific ways. Figure 1 and the pseudo-code describe the basic operation of the *SPAWN* protocol.

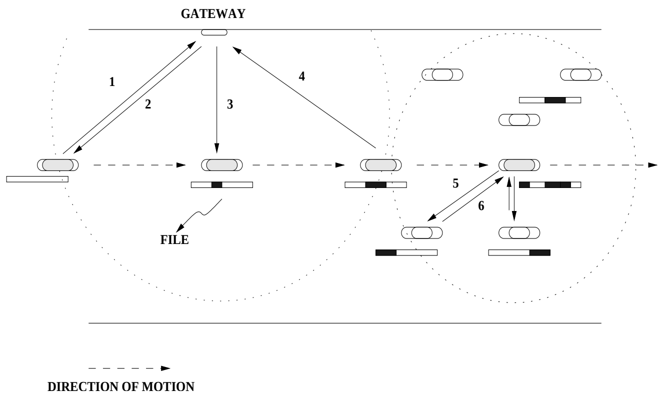

Figure 1. Evolution of a file in a node using the SPAWN protocol. (1) A car arrives in the range of a gateway,(2) initiates a download (3) downloads a piece of the file. (4) After getting out of range, (5) starts to gossip with its neighbors about content availability and (6) exchanges pieces of the file, thereby getting a larger portion of the file as opposed to waiting for the next gateway to resume the download

There are several components to the operation of the *SPAWN* protocol like Peer Discovery, Peer and Content Selection, and Content Discovery and Selection.

2.2.1 Peer Discovery

When a new node enters the vehicular network(such as entering a freeway or a section of freeway with access points), it requests the Gateway for the particular file. If the Gateway has the file in its cache, it starts uploading a piece to the node. Decision policies with respect to piece choice are discussed later. The node starts downloading pieces from the Gateway while it is in range. The Gateway also bootstraps it with a list of the last known peers who requested for the same file. This sort of *centralized* Peer Discovery phase is similar to other swarming protocols such as BitTorrent [5] and Slurpie [8].

The centralized approach to peer discovery in BitTorrent [5] and Slurpie [8] has several disadvantages (beyond the most obvious disadvantage of having a central point of failure). In our scenario, the Gateway can only bootstrap an incoming peer with the last few peers that passed by and were interested in the same file. This set is too small for efficient sharing/downloading. We propose a *decentralized* mechanism for peer discovery to be carried out en route. We utilize the broadcast medium of the wireless channel to *gossip* information about the content availability at neighbors. In a mobile environment, gossiping provides a way to incorporate location awareness into the peer discovery scheme [7]. Since TCP over multiple-hops suffers quite dramatically in the ad-hoc wireless scenario [11], a node is better off using unicast TCP connections with peers who are just 4 or 5 hops away.

Hence in *SPAWN*, the centralized approach and the gossiping mechanism can be used in conjunction to construct the mesh of peers and update connectivity information. *We introduced the gossiping mechanism to increase the robustness of peer list maintenance and discovery, in the presence of high degree of node churn, as well as the intermittent presence of gateways.*

Gossip is the mechanism used to advertise the piece list that a particular peer possesses. It leverages the broadcast medium of the network and also lets peers discover each other. A Gossip Message contains the TorrentID to identify the file being distributed by the Gateway, a bitfield representing the list of pieces that the originator possesses, a timestamp indicating when it was originated, and a list of *node-id*s indicating which nodes processed it along the route. A gossip message is sent using mac-layer broadcast. All nodes within range will hear it and process it depending on their type. We consider the presence of three kinds of nodes: (1)**Interested**: Nodes who are interested in this file, (2)**Uninterested**: Nodes downloading a file other that this one (3)**Relays**: nodes that don’t understand the *SPAWN* protocol.

Relay nodes simply drop Gossip Packets since they don't understand them. However they are running some routing protocol (ostensibly to support the general vehicular network and other safety applications that maybe running on top) and therefore will relay IP packets which are unicast through them, even those associated with SPAWN file transfers. Peers generate Gossip messages from time to time to advertise their presence and content. A naive gossiping scheme has a potential of generating a large number of gossip messages as well as the problem of ping-pong of messages, where two peers keep exchanging stale data.

(a)**Probabilistic Spawn** *Uninterested Spawn* nodes listen to gossip messages and forward them with a low probability. *Interested Spawn nodes* listen to gossip messages and forward them with a higher probability after stamping the route-list of the packet with their own id. An *Interested spawn* node who is currently downloading a file will generate Gossip messages on completion of downloading a new piece. (b)**Rate-Limited Spawn** Each Spawn node maintains two caches. Entries in the first cache contain the last gossip message heard from each node id with regard to any file that the node is interested in. The second cache contains the last gossip message heard from each node id with regard to files that the node is not interested in. Thus if a node is interested in 5 files, it will keep 5 Interested Caches and 1 Non-interested Cache. Thus the soft state requirement is linear in the number of files that a particular node is interested in and does not depend on the total number of files that are being shared on the network. Each node can decide to keep the individual caches as large as it wants to. In our simulations we observed that the maximum number of good peers(where a good peer is a peer interested in the same file and within 4 hops) was never more than 15 for realistic car densities, so a cache size of 20 is sufficient for the scenarios that we investigate. Thus Rate-limiting Spawn by itself does not impose any strict overhead on the soft state at each node. A newer gossip message from a node will overwrite the earlier cached message from the node. Thus the caches maintain the last gossip heard from as many nodes as possible.

Each node maintains two rates, a Low_Fwd_Rate at which it periodically selects a recent message from its Non-Interested cache and forwards a gossip message, and a High_Fwd_Rate at which it periodically selects a message from its Interested cache and forwards a gossip message. The decision about which message to select from each cache can be made in different ways. To ensure that stale messages are not re-forwarded back to the source, we cache the source addresses of the gossip messages as well. We experiment with two decision policies in this paper.

(b1)**Rate-Limited-Recent Spawn** The gossip message with the most recent time-stamp is forwarded. (b2) **Rate-Limited-Random Spawn** The gossip message is selected at random from the relevant table.

2.2.2 Peer & Content Selection

TCP connections spanning fewer hops perform better in multi-hop wireless networks. To that end, *SPAWN* does some intelligent Peer Selection based on the distance of the peer possessing a certain piece it intends to download. This information is gathered from the gossip messages. One could also gather this information from GPS enabled traffic-safety messages that are likely to become ”standard” applications running on vehicles in the future. However we decided to keep our ”inference” methodology independent of other applications that may co-exist on the same node.

We introduce a *proximity-driven piece selection* strategy. It uses several distinct strategies to choose which piece to download, based on how much has been downloaded already. Selecting pieces to download in an order that reduces contention at the peer serving the piece has a definite impact on performance as observed earlier [5]. We employ several strategies that might perform better in the wireless setting. We estimate proximity based on hop-count. Other approaches to estimate proximity can be using ping messages to measure round-trip times, however this approach inadvertently introduces more delay and message overhead. Our hop-count based estimate performs well in a mobile wireless scenario.(1)*Rarest-Closest First:* Choose the rarest piece among all peers in the peer list, break ties based on which peer is closest. (2)*Rarer-Closer:* Weighted piece selection based on both the rarity and the distance of the peer sharing the piece. (3)*Closest-Rarest First:* Select the rarest piece among all closest peers. This strategy induces osmosis, whereby the piece propagates one application-hop at a time until it pervades the network.

If a client is not downloading anything yet, it uses the Closest-Rarest-First strategy. A piece is rarer if fewer neighbors have it. In the case of a tie for rarity, the selection is not made randomly as was done in BitTorrent instead we bring proximity awareness in the content selection. We evaluate the performance impact of these distinct content selection strategies using simulations as well as with analytical models in later sections.

2.2.3 Content Discovery & Delivery

When two peers become neighbors, they exchange bitfields that describe which pieces they possess. When a peer fully receives a piece, it immediately notifies all of its neighbors. Thus, every peer always knows exactly which pieces its neighbors possesses. We evaluate the robustness and performance of the protocol with varying piece size and varying channel characteristics using simulation. *SPAWN* uses

UDP for delivery of gossip messages and TCP for content delivery.

2.3. Design Rationale

It has been argued by Jinyang *et al* [10] that the key deciding factor to whether a large ad hoc network is feasible is the locality of traffic. The effect of traffic locality determines to a large extent the scalability of per node capacity.

SPAWN tries to minimize the peer-side wall clock time taken to download a large file. *SPAWN* like all swarming protocols is motivated by the fact that for popular files, the content distributor becomes the bottleneck as far as bandwidth and processing is concerned while the downloaders have ample spare capacity. In vehicular networks this problem is further exacerbated due to the intermittent and short-lived connectivity to the infrastructure. This form of co-operative data transfer encourages locality in network traffic and consequently *scales* while at the same time providing extended perceived connectivity. We prove it more formally using simulation and analysis.

In BitTorrent protocol, a peer contacts the tracker every 30 seconds to refresh its peer list. In a mobile scenario such as vehicular networks, the timeliness of refreshes is paramount in the presence of high churn. By sending *gossip* messages a peer can maintain a list of peers which are more likely to share pieces and this way handle the churn in the peer list in a distributed manner. The exchange of gossip messages with neighbors increases the probability of a peer finding a piece in its locality. The inherent broadcast nature of the wireless medium is utilized by *SPAWN* since we perform a MAC layer broadcast of gossip messages instead of sending unicast messages to each peer individually.

3. Models

Our model for file-sharing is influenced by the model in Veciana*et al* [1]. The inherent capacity limitations of wireless networks add another dimension of complexity to the analysis. The primary goal is to show the efficacy of co-operative data dissemination in vehicular networks.

3.1. A Simple Deterministic Model

We first present a simple model which provides some basic intuition of the benefits of using co-operative downloading in vehicular networks. We consider a deterministic model in which there are n users interested in downloading a large file that is initially available to one peer.In our case the peer is the gateway (i.e the gateway acts as an intermediary proxy).

It was shown in previous models for peer-peer networks such as [1] using a simple deterministic arguments that the **service capacity** defined as *the number of peers available for serving the file*, of the peer-peer network grows exponentially with the number of peers and consequently the delay averaged over *all* the peer scales as $O(log\ n)$ instead of $O(n)$ for a client-server model with a fixed set of servers.

Theorem 1 *The average delay experienced by all the peers in downloading a multi-part file is as follows,*

$$\overline{d}^{(m)} \approx \frac{\tau_0}{m}\, log_2\, n$$

where m is the number of identical sized chunks that the file is divided, n is the number of peers and τ_0 is the amount of time taken to download the whole file, if downloaded from a single peer.

There are certain assumptions made in the derivation of the Thm(1) in Veciana et al [1], that do not hold true in the case of ad hoc networks. We address each of the assumptions and derive relevant bounds for service capacity and average delay for peers participating in *SPAWN* protocol for vehicular networks.

The first assumption, was the *unconstrained* capacity of the underlying physical network (service capacity was however limited by the upload capacity of a peer). Hence by the virtue of this fact, an exponential increase in number of peers resulted in an increase in service capacity. However in the case of ad hoc networks this will not hold true. We will derive constraints on the capacity based on the traffic patterns of the network.

The seminal work of Gupta et al [13] showed that the *per-node* capacity in an n-node random ad hoc network is $\Theta(1/\sqrt{nlogn})$. A looser bound that can intuitively be explained is $O(1/\sqrt{n})$ as was shown in [10]. We use this *per-node* capacity bound to derive the *service capacity* of the peer-peer network to provide some basic intuition.

We derive the average delay experienced by a peer with the constrained network capacity which varies with network size as $O(1/\sqrt{n})$. We consider an n-node network, with the file available at one peer at the start. Without loss of generality we assume $n = 2^k$. Let b bps be the maximum per node upload capacity (i.e. when there is just 2 nodes in the network so they can send at the maximum capacity). Let s be the size of the file to be co-operatively downloaded. Define $\tau_0 = \frac{s}{b}$ to be the time taken to download the file from a single peer. Under this idealized setting, we can divide the evolution of service capacity as shown in the figure into "epochs". At each epoch, the service capacity *doubles*. However the capacity and the current state of the network(i.e. the number of nodes) determine the duration of this epoch. To get an intuitive understanding, how the service capacity is affected, we consider a looser bounds, that the *per-node* capacity decreases as $1/\sqrt{n}$.

Definition 1 *Define $\Delta\tau_i$ as the duration of the i^{th} epoch. Hence,*

$$\tau_k = \sum_{i=0}^{k-1} \Delta\tau_i$$

Also, let $n(\tau_i)$, denote the number of peers uploading the file at time τ_i.

Hence we have,

Observation 1

$$\Delta\tau_i = c\ \tau_0 \sqrt{n(\tau_i)}$$

where c is a proportionality constant.

This observation is based on the fact that, each epoch duration is determined by the number of nodes that are present at the start of the epoch, since the throughput degrades as $O(1/\sqrt{n})$ hence, the epoch duration which is the time taken for a peer to upload the file to another peer increases as $O(\sqrt{n})$ in the idealized deterministic model above. Also by the definition above, $n(\tau_k) = n$ and $n(\tau_0) = 1$. Where τ_k is the total time in which the file is replicated to all the n peers.

The average delay experienced by a peer can be computed using the above observations. Let d_j denote the delay of the j^{th} peer and also observe that 2^i peers finish downloading in i^{th} epoch. Hence,

Lemma 1

$$\overline{d} = \frac{1}{n}\sum_{j=1}^{n} d_j = \frac{1}{n}\sum_{i=0}^{k-1} 2^i\ \tau_i \approx \tau_0\ \sqrt{n}$$

Observe that $n = 2^k$ and τ_i can be obtained from the observation 1. Hence plugging in these values we obtain the expression. This result provides basic intuition of the impact of the underlying physical capacity on the service capacity of the peer-peer network. *The average delay seen by a peer scales as $O(\sqrt{n})$ compared to $O(log\ n)$ as proved in [1]. However this result is still favorable compared to the $O(n)$ scaling expected in a client-server model. This result supports our claim that co-operative downloading indeed scales inspite of the constraints of the ad-hoc network physical capacity.*

We can derive similarly for the multi-part file downloads. The m multi-part factor reduces the average delay to $\frac{\tau_0\ \sqrt{n}}{m}$.

3.1.1 Traffic Pattern Model

Earlier studies such as Morris *et al* [10] claimed that the per-node capacity of $\Theta(1/\sqrt{nlog\ n})$ [13] were pessimistic since they assumed random nodes for choice of source and destinations that establish a connection. However in reality, an ad-hoc network demonstrates locality in traffic patterns. *The primary result of [10] was that the expected path length of a particular traffic pattern determines capacity scaling. In short, the less local the traffic pattern, the faster per node capacity degrades with network size.*The most obviously scalable traffic patterns are *exactly* local. That is, each node sends only to nodes within a fixed radius, independent of the network size.

One of the design goals of the *SPAWN* protocol was to achieve scalability. One way to achieve it is to make the traffic as "local" as possible. Our peer selection and content selection strategies such as closest rarest first are designed with this scalability in mind. In this section we model the locality of traffic imposed by our protocol. To determine the how "local" the traffic patterns are, in other words, what is the expected path length between sources and destinations. We use the result due to [15] derived with a log-normal shadowing radio propagation model for studying the connectivity of wireless ad-hoc networks. The main result that we use is the probability that a link exists between two nodes. While their results were derived for a general source and destination at a distance r, we derive the probability of having a link between two nodes at a distance r from each other. We take into account the vehicular network scenario to relate the car density, the velocity(v_{ave}) and the number of lanes(L) in the freeway with the average distance between two nodes as described in [17] as follows:

$$r = v_{ave}\tau/L \tag{1}$$

here τ is the average time lag between two vehicles. Using this expression we derive the probability that a link exists between two arbitrary nodes, using previous expression for

$$p_{link} = \frac{1}{2} + \frac{1}{2}\text{erf}\Big[\frac{\beta_{th} - \eta\text{log}(r)}{\sqrt{2}\sigma}\Big] \tag{2}$$

where, β_{th} the threshold attenuation is set to 40dB. Also σ, the standard deviation of shadowing is set to 4dB. η the path-loss exponent which depends on the environment and the terrain structure and can vary from 2 in free space to 6 in urban areas.

Taking into account the popularity of a file, we denote the probability that a random node is interested in a file to be p_i and is independent of the other nodes. Hence when we say a file 60% popular then each node has a $p_i = 0.6$ of being interested in that file independent of the other nodes.

Lemma 2

$$l_{ave} = \frac{1}{1 + \frac{ln\ (p_{link}p_i)}{ln\ (\rho_a)}}$$

if $\rho_a >> 1$ where, p_{link} is the probability of a link between two nodes at a distance 'r' apart, ρ_a is node density and p_i is the popularity of the file.

We skip the proof for brevity, the interested reader can see the proof in a more complete version of the paper [14]. To understand the impact of these parameters on the average hop count and consequently the protocol performance, consider the graphs as shown in figure. We observe in figure 2 that as the velocity of nodes increases the expected application hops increases as the probability of a link between two nodes depends on the velocity as expressed in Eq.(2). The impact of the node density is also clear from the figure. At very low node densities, the Expected number of application hops drops, which is intuitive since you have fewer number of nodes in general so you have to travel further to find a node and consequently an interested peer. Similar pattern is depicted in the expected path length for different values of popularity as shown in figure 3. It has been observed that the one-hop capacity of an ad-hoc network is determined by the amount of spatial reuse possible [10]. Our analysis of expected path length (in terms of application hops) will give us an upper-bound on the per node capacity since $\lambda < c\, l$, where λ denotes the per-node capacity and l the expected path length(in physical hops). However, note that the expected path length(in application hops) l_{ave} will always be upper-bounded by l the path length in physical hops.

4. Simulation

In this section we describe the simulations we performed to evaluate the gossip schemes proposed. We implemented the gossip schemes in *Nab* a network simulator written in *Ocaml*. *Nab* [3] is a fast (For example, a 100 node simulation run for 300 simulated seconds completes in 4 minutes), flexible and scalable simulator for ad-hoc networks. We incorporated our mobility model, and a simple traffic model into the simulator. The car arrival process at the access point follows a poisson distribution with the average interarrival time varying from 0.5 to 4 seconds. We consider only one direction of vehicle motion in the highway. The peer group is maintained among cars driving in the same direction. When a car comes within range of the gateway, it starts downloading random pieces of the file. The tracker running on the gateway bootstraps the car with a set of 6 peers who last crossed that gateway and were interested in the same file. Each car possesses an initial speed which is varied at random by a small amount every 5 seconds. Cars maintain the same direction throughout and are not affected by the speeds of cars around them. The simulation parameters are as follows: File Size is 5MB, the piece size is 64KB and the velocity varies from 40-80mph.

We used a simplified version of the 802.11 DCF protocol implemented in the NAB simulator. In particular, the gossip messages are broadcast in the CSMA mode of 802.11. At the network layer we used AODV(Ad-Hoc On-Demand Distance Vector Routing). There are other on-demand routing protocols such as DSR(Dynamic Source Routing) which can be potentially used in Vehicular Ad-Hoc Networks. Moreover, proactive routing protocols(eg OLSR) could also be used. The optimal choice of routing scheme is clearly an important issue. However since the focus of this paper is to evaluate application layer strategies, we will keep our study routing protocol agnostic. Leveraging routing protocol specific messages(for instance coupling our gossip messages with RREQ messages for efficiency purposes is part of continuing research effort). For chanel data transfer rate we assume the typical 802.11a data transfer rate. This is a conservative assumption given that DSRC has a rate varying from 6-27Mbps. We are interested in the efficiency of the gossip schemes, the message overhead each scheme introduces. We analyze the impact of each of the simulation and traffic model parameters on the performance of the gossip schemes.

4.1. Analysis of Gossip Schemes

There are essentially three characteristics which we observe while evaluating the gossip mechanisms. (a) *Good Peer* Set Length: ”Good Peer” is defined as the set of peers that are within k hops of a particular node. In all our simulations we set k to be 3. (b) Local File Downloaded (c) Peer-Space File Downloaded: the total fraction of the file that is present at a node and its Peer-List nodes.

We are interested in the Peer-Space File Evolution since this is the upper bound for the achievable fraction of the file for a particular car at a particular instant. The number of *Good Peers* in the Peer-List is a measure of the locality awareness of the peer discovery scheme.

Figure 4 shows the evolution of the good peer list with the different gossip schemes at a typical node. The performance of a swarming protocol without any gossip clearly falls off as the peer starts moving away from the gateway. The various gossip schemes perform the same as far as the good peer set is concerned. The local File Evolution shown in Figure 5 for different schemes supports the intuition that gossip will help in retrieving more pieces of the file. The Peer-Space File Evolution in Figure 7 depicts that the gossip does enable robust peer discovery in the presence of high churn of peers.

4.2. Message Overhead

The advantages of gossip are clearly visible in the simulation results we presented. A natural question to ask is what is the cost of this robustness and location awareness? We ran simulations to analyze the Message Overhead of each of the gossip schemes. One of the simulation parameters that would have an impact on the overhead is

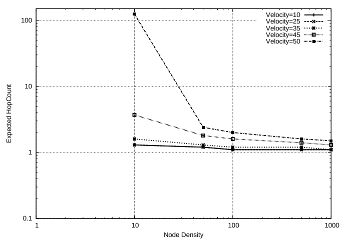

Figure 2. Impact of Average Velocity on the Expected # of Application Hops needed to find a peer, with varying node densities

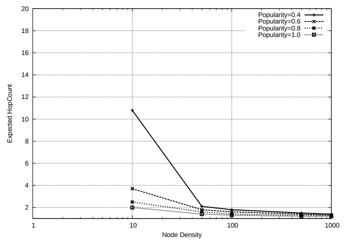

Figure 3. Impact of Popularity of the File on the Expected # of Application Hops needed to find a peer with the file, with varying node densities

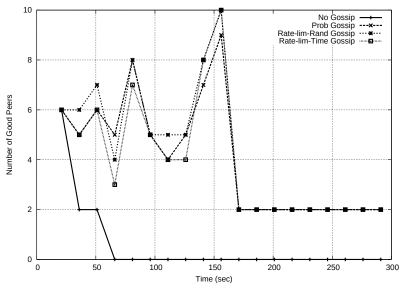

Figure 4. Number of Good Peers

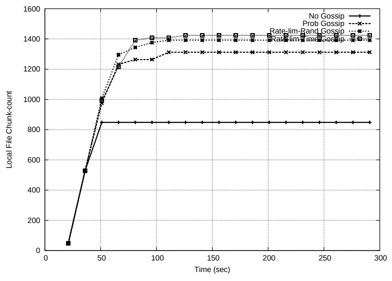

Figure 5. Local File-Chunk Evolution

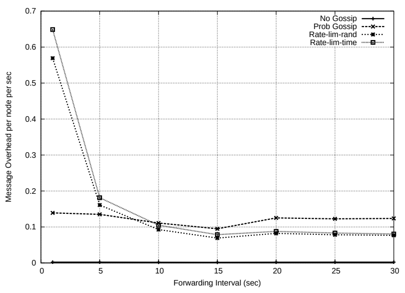

Figure 6. Message Overhead with Forwarding Interval

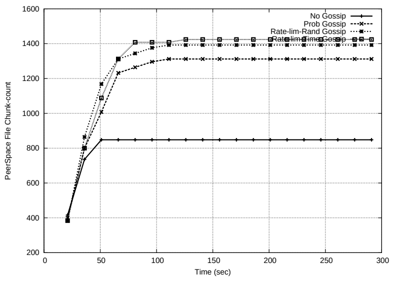

Figure 7. Peer-Space File-Chunk Evolution

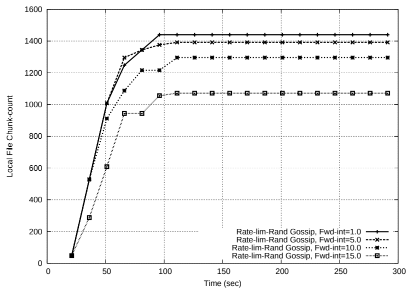

Figure 8. Effect of Forwarding Interval on Chunk Evolution

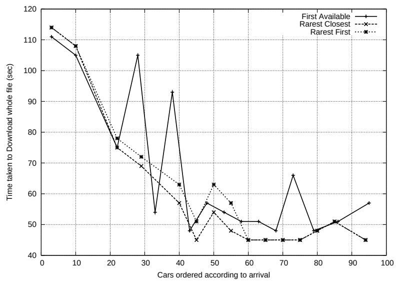

Figure 9. Different PieceSelection Strategies

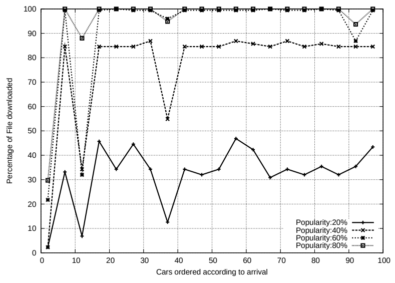

Figure 10. Popularity works !

the Forwarding Interval of the gossip messages. Figures 6 and 8 show that by varying the forwarding interval the overhead reduces considerably while still keeping the local File-Chunk evolution relatively stable. For our simulation runs, a forwarding interval of 1 second provided low message overhead and decent evolution rate.

4.3. Piece Selection Strategy

We experimented with three different piece selection strategies: *First Available*, *Rarest First* and *Rarest Closest*. First Available tries to fill the first empty chunk in the bitfield that can be filled. The search procedure is from low index to high, so lower index bitfields get filled up faster. Such a strategy is useful for files which have partial content usefulness. Some Mpeg files will play parts of the file if you have the partial file, so in these cases it would be advantageous to assemble the initial parts of the file first. Rarest First is the BitTorrent policy of searching for the rarest bitfield in your peerlist and downloading it. In wireless networks this could suffer from problems such as trying to download a rare piece from someone quite far away, while a slightly less rare piece is located very close to you. Connections to far away hosts are likely be unstable and lossy so we experiment with a variation of the rarest first scheme called Rarest Closest which weighs the rare pieces based on the distance to the closest peer who has that piece. Rare pieces which are situated closer to the node are preferred. A node can guess the distance of a particular peer by looking at the gossip message of the peer, and calculating the number of nodes which have stamped the packet from the relevant field. Figure 9 shows the experienced download times for the three strategies; it is clear that Rarest Closest consistently gives shorter download times than Rarest First. First Available does the worst since it encourages determinism and reduces the entropy of the system.

4.4. Popularity Index

One of the critical factors in determining the download time of a file is its popularity. We varied the popularity index of the file (the percentage of cars that are interested in this file) from 20% to 80%. Figure 10 shows the percentage of the file that is downloaded by the cars in the allotted 300 seconds time. It is clear that low popularity files are slow to download, however the speed ramps up pretty fast and gets bottlenecked by the capacity of the wireless channel at around 60% popularity. From there on, there are always “enough” new chunks for cars to keep downloading until they finish.

5. Related Work

BitTorrent is a popular file-sharing tool, accounting for a significant proportion of Internet traffic. BitTorrent builds its overlays by randomly selecting peers, a fact that can be potentially wasteful and affect performance. The topology of the peer-peer network determines the scalability as well as the capacity of the ad-hoc content delivery mechanism via swarming. While [2] tries to answer basic questions for BitTorrent and Slurpie like co-operative swarming assisted downloads. The primary purpose of the protocols are is two-fold: from the conventional *server perspective*: reduce the load of the origin server or the content publisher and secondly the *client perspective* reduce the download time. Veciana *et al* [1] define the *service capacity* of peer-peer networks and analyze two regimes in the evolution of a file distribution using branching process models and Markovian models. The benefits of gossiping techniques have been exploited in ad-hoc routing as well as in wired networks.

The *Infostations* model [12] of wireless ad-hoc networks aims to provide trade-offs between delay and capacity of these networks by providing geographically intermittent connectivity. It has been observed that the *Infostations* model trade *connectivity* for *capacity*, by exploiting the mobility of the nodes. Our vehicular network model is similar to this model. However one of the key differences as far as the data dissemination is concerned is that in all the previous applications these *Infostations* are viewed as information gatherers rather than information disseminators(i.e. they use the *pull* model rather than the *push* model of data dissemination). While the application and the data itself is delay-tolerant, its the service capacity *scaling* that is the key performance gain using the co-operative swarming protocol.

The Drive-thru Internet project [4] investigates the usability of providing network connectivity and, ultimately, Internet access to mobile users in vehicles. Several measurement experiments analyzing the feasibility of the connectivity architecture at high speeds were conducted [4]. One or more locally interconnected access points form a so-called *connectivity island* that may provide local services as well as Internet access. Several of these connectivity islands along a road or in the same geographic area may be interconnected and *cooperate* to provide network access with *intermittent connectivity* for a larger area.

6. Conclusions

In this paper, we analyzed the scalability and performance of bulk data dissemination in vehicular ad hoc networks. We gave a brief overview of the *SPAWN* protocol which tries to achieve the design goals of scalability in wireless networks, improved perceived performance for individual clients using co-operation in highly mobile scenario. Our results show that client performance improves as the popularity of the file increases. This is intuitive since the probability of finding peers in the vicinity serving some chunk of the file increases. We have shown newer chunk selection strategies that are particularly optimized for the wireless setting including the *rarest-closest first* which performs much better than default strategy used in BitTorrent and Slurpie of *rarest-first*. Our simulations result show that this strategy indeed introduces locality awareness and improves the scalability and hence the performance of peer-peer network.

We presented detailed analytical models for the protocol performance. We incorporated a more realistic log normal shadowing radio model both in our simulations and analysis for vehicular networks and analyzed their impact on the protocol performance. We first presented a simple deterministic model of the SPAWN performance improvement over the conventional client-server architecture. We observed that in spite of the capacity limitation in wireless networks, using a swarming protocol still makes sense and provides better scalability that the traditional client-server approaches.

One of the interesting open issues with all swarming protocols in general and consequently *SPAWN* as well is the incentive for co-operation. We have seen that as more peers participate in the protocol, the performance improves. Our protocol uses some of the BitTorrent-like [5] incentives in the protocol such as the tit-for-tat policy and choking algorithms for encouraging co-operation; however the high churn of nodes in vehicular networks make the policies too strict. A credit based system that is accrued across different file dissemination not just one file can be an approach that will encourage co-operative behavior in vehicular networks.

References

[1] X. Yang and G. de Veciana, *Service Capacity of Peer to Peer Networks* In Proc. of INFOCOM 2004

[2] Dongyu Qiu and R. Srikant, *Modeling and Performance Analysis of BitTorrent-Like Peer-Peer Networks,* To appear in SIGCOMM 2004

[3] Henri-Dubios Ferriere, *http://nab.epfh.ch*

[4] J. Ott and D. Kutscher, *Drive-thru Internet: IEEE 802.11b for Automobile Users,* In Proc. of IEEE INFOCOM 2004.

[5] B. Cohen, *Incentives Build Robustness in BitTorrent,* IPTPS 2003.

[6] L. Yin and G. Cao, *Supporting Co-operative Caching in Ad Hoc Networks,* In Proc. of IEEE INFOCOM 2004.

[7] D. Kempe, J. Kleinberg, A. Demers, *Spatial gossip and resource location protocols,* In Proc. of ACM STOC, pp163-172, 2001.

[8] R. Sherwood, R. Braud and B. Bhattacharjee, *Slurpie: A Co-operative Bulk Data Transfer Protocol,* In Proc. of IEEE INFOCOM 2004.

[9] V. Padmanabhan and K. Sripanidkulchai, *The case for cooperative networking,* IPTPS 2002.

[10] J. Li, C. Blake, D.S.J. De Couto, H.I. Lee and R. Morris, *Capacity of Ad-Hoc Wireless Networks,* In Proc. of ACM MOBICOM 2001

[11] K.Tang, M. Gerla and R. Bagrodia, *TCP Performance in Wireless Multi-hop networks,* Proceedings of the Second IEEE Workshop on Mobile Computer Systems and Applications, 1999.

[12] T. Small and Z.J. Haas, *The Shared Infostation Model - A New Ad Hoc Networking Paradigm (or Where there's a Whale, there's a Way),* In the Proc. of ACM MOBIHOC 2003

[13] P. Gupta and P. R. Kumar, *The Capacity of Wireless Networks,* IEEE Trans. on Info. Theory, March 200, pp388-404

[14] Alok Nandan, Shirshanka Das, Giovanni Pau, Mario Gerla, M.Y. Sanadidi, Co-operative Downloading in Vehicular Ad-Hoc Wireless Networks, UCLA CS Technical Report # 040035, October, 2004.

[15] R. Hekmat and P. Van Meighem, *Study of Connectivity in Wireless Ad-Hoc Networks with an Improved Radio Model,* In Workshop on Wireless Optimization, WiOpt 2004

[16] *Dedicated Short Range Communication Architecture,* www.astm.org/SNEWS/MAY_2004/dsrc_may04.html

[17] W. Enkelmann, *FleetNet Applications for Inter-Vehicle Communication,* IEEE Intelligent Vehicles Symposium (IV 2003), pp. 162-167, Columbus, OH, USA, June 2003