Ad Hoc Networks 12 (2014) 291–304

Contents lists available at ScienceDirect

Ad Hoc Networks

journal homepage: www.elsevier.com/locate/adhoc

Reprint of "Prioritized gossip in vehicular networks" ☆

Alejandro Cornejo [a,*], Calvin Newport [a], Subha Gollakota [a], Jayanthi Rao [b], T.J. Giuli [b]

[a] Massachusetts Institute of Technology, Computer Science and Artificial Intelligence Laboratory, 32 Vassar St., Cambridge, MA 02319, USA
[b] Research and Advanced Engineering, Ford Motor Company, Dearborn, MI, USA

ARTICLE INFO

Article history:
Available online 9 August 2013

Keywords:
Radio networks
Gossip
Vehicular networking

ABSTRACT

We propose using real world mobility traces to identify tractable theoretical models for the study of distributed algorithms in mobile networks. Specifically, we derive a vehicular ad hoc network model from a large corpus of position data generated by San Francisco-area taxicabs. Unlike previous work, our model does not assume global connectivity or eventual stability. Instead, we assume only that some subset of processes might be connected through *transient paths* (e.g., paths that exist over time). We use this model to study the problem of prioritized gossip, in which processes attempt to disseminate messages of different priority. We present CabChat, a distributed prioritized gossip algorithm that leverages an interesting connection to the classic *Tower of Hanoi* problem to schedule the broadcast of packets of different priorities. Whereas previous studies of gossip leverage strong connectivity or stabilization assumptions to prove the time complexity of global termination, in our model, with its weak assumptions, we instead analyze CabChat with respect to its ability to deliver a high proportion of high priority messages over the transient paths that happen to exist in a given execution.

1. Introduction

A difficulty in studying distributed algorithms for mobile networks is defining realistic mobility. A common solution to this difficulty is to use position traces from real mobile network deployments. For example, Liu et al. [17] use traces of San Francisco-area taxicabs to study the performance of their *VMesh* strategy for local information storage, and Sarafijanovic-Djukic et al. [21] use traces from cabs in Warsaw to study an *island hopping* strategy for routing. In these two papers, as in most other data-driven analyses of mobile network algorithms, the position traces are used to support *simulation studies*. By contrast in this paper we propose using traces to derive models suitable for generating *theoretical results*.

1.1. Restricted dynamic graphs

Specifically, we begin with the *dynamic graph* model of [16], which describes the connectivity of processes in a mobile network as a graph in which the edge set can change arbitrarily from round to round. (An edge between two nodes in a given round indicates the ability for the associated processes to communicate in that round.) We then use position traces from real mobile networks to identify properties of these graphs that arise in practice. These properties define a restricted dynamic graph model. The goal is to identify properties that allow a theoretician to generate better algorithms and bounds, while at the same time maintaining the results' applicability in practice.

For example: imagine that the study of buses traveling on a fixed bus route reveals a small amount of connectivity at any one time step, but a high probability that a particular pair of buses will eventually be connected (e.g., as they

DOI of original article: http://dx.doi.org/10.1016/j.adhoc.2012.06.016

☆ This article is a reprint of a previously published article. A publishers' error resulted in this article appearing in the wrong issue. The article is reprinted here for the reader's convenience and for the continuity of the special issue. For citation purposes, please use the original publication details; A. Cornejo et al./ Ad Hoc Networks 11 (2013) 397–409.

* Corresponding author.
E-mail addresses: acornejo@csail.mit.edu (A. Cornejo), cnewport@csail.mit.edu (C. Newport), subha@mit.edu (S. Gollakota), jrao1@ford.com (J. Rao), tgiuli@ford.com (T.J. Giuli).

1570-8705/$ - see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.adhoc.2013.08.003

pass each other on the route). This might inspire the dynamic graph property that a specific pair of nodes are expected to be connected in the graph, for at least x rounds every T rounds, with high probability, where x is a small constant and T is a large constant, both derived from the bus traces. When analyzing a distributed algorithm to be deployed on buses, this data-derived property can be used to tame the otherwise arbitrary edge changes in the graph. A result proved in this model is likely to hold in the real world network from which the property was derived.

1.2. Data-derived dynamic graph properties

To validate the usefulness of this modeling approach, we study vehicular ad hoc networks (VANETs) comprised of taxicabs in an urban setting. The source of our experimental observations is a large corpus of position traces gathered from GPS-equipped embedded computers deployed in San Francisco-area taxicabs [1]. We study networks of 100 and 200 vehicles. We next examine the properties of the connectivity graphs induced by these networks, first showing that there is never global connectivity. For example, in our 200 vehicle networks the largest connected component observed in any round contained no more than 65 vehicles. What we instead observe is a large amount of *transient connectivity* between vehicles—e.g., paths over time—with half of the vehicles in our 200 vehicle networks having transient connections to at least 150 other vehicles. We also observe moderately stable pairwise links, with 25% of links lasting at least 10 s and 10% lasting at least 30. We combine these observations of transient connectivity and pairwise link stability into what we call the *ℓ-stable transient path* property, which describes a transient path between two vehicles such that each hop in the path exists for at least ℓ consecutive rounds. [1] In this paper, when we analyze the performance of distributed algorithms, we do so in the dynamic graph model under the assumption that such paths exist; e.g., *given an ℓ-stable path between nodes u and v in the dynamic graph, starting at round r, we prove the following performance result, etc.*

1.3. The prioritized gossip problem

With our data-driven model defined, we turn our attention to solving a specific problem. A commonly-cited use for vehicular networks is the dissemination of timely information between vehicles [8,27], for example: an observation about traffic, the location of a road-side access point, or an accident alert. This problem can be cast as a form of gossip, in which vehicles occasionally generate *messages* of different priorities that need to be disseminated. Presumably, an accident alert would have higher priority than an observation of a traffic jam. We refer to this problem as *prioritized gossip*, and we study it in the context of the dynamic graph model with ℓ-stable transient paths.

A challenge for gossip in our setting is the lack of strong connectivity assumptions. In contrast to previous work [16], we do not assume global connectivity (or even that every pair has transient connectivity), and this prevents us from proving the time complexity of global termination (e.g., *the gossip problem terminates in $O(n^2)$ rounds*). Instead, we can only expect that *some* pairs may be connected by an ℓ-stable transient path, for varying ℓ values. This leaves the designer of prioritized gossip algorithms the task of proving their algorithms leverage such paths, when and if they arise, to deliver as much high priority information as possible. Such results are weaker than those guaranteeing global termination, but because they make no connectivity assumptions they are applicable in a wider variety of practical settings.

Another challenge of solving gossip in our model is the presence of priorities. Without priorities, it is sufficient for processes to work through their message queue in round robin order, broadcasting a new message in each round: this behavior guarantees that over any ℓ-stable transient path, ℓ different messages are delivered (given a sufficient number of messages existing in the system). Priorities, however, complicate this approach, as we not only desire to send *unique* messages, but we also want to send *high priority* messages. Imagine, for example, a process u with an ℓ-stable transient path to v, and a message queue of size much larger than ℓ. The round robin approach might lead u to deliver ℓ low priority messages during the ℓ rounds it participates in the path. A good prioritized gossip algorithm, therefore, must be careful in how it schedules its messages for broadcast.

1.4. The t-latency metric

To capture the effectiveness of a given gossip algorithm's priority scheduling scheme, we introduce the *t-latency* metric, which upper bounds the number of rounds required for a process to broadcast its t highest priority messages, over all rounds in which it has at least t messages, over all executions. An algorithm that guarantees a small t-latency with respect to t, for all t values, will deliver a high proportion of high priority messages at each hop of an ℓ-stable transient path. Our main performance theorems, summarized below, will leverage t-latency results proved with respect to our algorithm, CabChat, to lower bound the amount of high priority information the algorithm guarantees to be sent over a given ℓ-stable transient path.

1.5. Our results

In particular we consider the prioritized gossip problem with exponentially distributed priorities (i.e., messages with priority 1 are twice as important as messages with priority 2, which are twice as important as messages with priority 3, and so on). We propose the distributed algorithm CabChat that leverages properties of a slight variation of the *binary carry sequence* [3], which also describes an optimal solution the classic *Towers of Hanoi* problem [23,15].[2]

[1] Notice that capturing the stability of the hops is important as it bounds the total amount of information that can flow through the path. Assuming a rate of one message per round, an ℓ-stable path between u and v allows u to transmit ℓ messages to v.

[2] This versatile sequence has also been used to identify *Hamiltonian paths* in hypercube graphs and generate *binary reflected codes*, also known as Gray codes.