Hikari: DTN message distribution system

Sergio Carrilho
The University of Tokyo
7-3-1 Hongo, Bunkyo-ku
Tokyo 113-8656, Japan
sergio@hongo.wide.ad.jp

Guillaume Valadon
Universite Pierre et Marie Curie
104, avenue du President Kennedy 75016
Paris, France
guillaume.valadon@lip6.fr

Hiroshi Esaki
The University of Tokyo
7-3-1 Hongo, Bunkyo-ku
Tokyo 113-8656, Japan
hiroshi@wide.ad.jp

Abstract—**Communication and information dissemination is taken for granted in urban areas covered by a robust communication infrastructure. Nevertheless, the majority of the world has limited or no access to information because most of the populated areas are poorly covered by a communication infrastructure. To transport information between these locations we could make use of travellers or nodes that transit in these areas. We propose a DTN publish and subscribe system called Hikari, that uses nodes' mobility in order to distribute messages without using a robust infrastructure. The research area of Delay Tolerant Networks (DTN) aims at providing connectivity between points with network partitions or large delays. The Hikari system uses information about nodes' movement paths, to forward information to remote locations. The path information is advertised by nodes or predicted by the system based on historical information of nodes' trips in the system. Hikari uses a publish and subscribe architecture in where the information flows by using topics of messages or queries from nodes. Therefore Hikari does not use node identifiers for message forwarding thus eliminating the complexity of routing. Information about nodes is not important for the forward decisions, since nodes are just carriers of information. Simulations in the Paris subway system showed us that the Hikari system and its message distribution algorithm achieve a superior deliver rate while keeping redundant messages in the system low, which is ideal when using devices with limited resources for message dissemination.**

I. Introduction

Delay Tolerant Networks (DTN) is the research area that deals with networks characterized by large partitions and delays and where end-to-end connectivity is not assured. Many types of networks fit into this category, for example, sensor networks for animal tracking [8], Developing regions message delivery [12], [7] , disaster stricken areas and military networks. In such networks, because paths between source and destination are not known in advance, routing is a challenging problem. There are several routing mechanisms proposed for such environments, and most of them use the node-to-node message exchange, in an opportunistic way, to deliver messages from source to destination. Information for routing is based on previous encounters and frequent routes travelled. In these systems, routing decisions are taken at each node independently. In DTNs, epidemic style algorithms are the ones that achieve the best delivery ratios but consume resources such as buffer size and add redundancy and noise to the system.

Publish and subscribe systems have been around since the early days of the internet. In such systems a subscriber who is interested in a topic, subscribes to a publisher. When topics are available or there are new updates, topic are sent to the subscriber, when she becomes available. The topics are delivered based on their content.

Most scenarios considered in the routing design process use information from previous encounters between nodes. In a scenario where some information about nodes' movement is known beforehand, complex routing is not needed. For example, in a metropolitan scenario, where stations are fixed, and trains movement is known beforehand, messages can be distributed between stations using trains or people traveling on them, given that peoples' destinations are also known. We can make use of this information for forwarding messages from/to hubs where many nodes meet (e.g. stations) and allow message exchange between nodes in different stations, using other nodes. Our approach, the Hikari system, works on such systems and avoids the complexity of routing issues by delivering topics based on content in a pub/sub way. The novelty of this approach is that information about nodes is not important and messages are forwarded to the destination based on their content. We make use of Location Masters to help with the process of discovering nodes' destinations. In order to validate the algorithm we have built the discrete modular Public Infrastructure Transport Simulator (PITS) for comparing the Hikari system with epidemic and random message distribution algorithms.

Our main contributions are:

The Hikari System: A DTN message distribution system that uses nodes' mobility and message content for message dissemination in a public transport scenario.

The modular PITS simulator: A modular generic simulator used for simulating message distribution in a DTN environment for public transport systems.

A real public transport infrastructure model for simulation: The Paris subway system was used for validation of the Hikari system.

This paper is organized as follows. Section 2 introduces the Hikari system design. Section 3 presents the message semantics. Section 4 introduces the model used for the simulations. The PITS simulator is described in Section 5. Section 6 describes the simulation, its requirements and the methodology used. Section 7 presents the preliminary results and some discussion around them. Section 8 summarizes the related works and the conclusion is presented in section 9.

II. The Hikari System

The aim of the Hikari system is to provide connectivity to environments which are disconnected from the internet and/or any fixed network. The initial goal was to provide connectivity to remote villages in developing countries using minimal or no infrastructure. The idea has evolved to consider the case of a metropolitan scenario where nodes with devices can be used to distribute topics without the need of a "always connected" environment. The main characteristics of this system are:

- Distribute messages using minimal infrastructure.
- Use node mobility for message forwarding.
- DTN publish/subscribe system for messages delivery.

The main features of this system are:

- No complex routing needed.
- Use nodes' path information for selecting carriers for message distribution.

A. System Description

The Hikari system is aimed at a scenario with the following characteristics: (i) Nodes gather in "stations" or "hubs" which are areas with a high density of nodes, (ii) nodes board some sort of transportation which is regular and takes them to another location/station, (ii) They can stop at any station in the middle of their trip, (iv) every station has a LM. The Hikari system is divided in two parts: The node-to-node part and the main message distribution part.

Node-to-node: In small areas outside stations, nodes forward messages in a peer-to-peer way, using other nodes for ferrying messages.

Main distribution system: System with stations, LMs and special nodes that distribute messages to remote destinations.

In this paper we consider only the Main distribution part of the system, and as for now, we will call it simply, Hikari system.

B. System Components

This system has 3 main components:

Location Master (LM): Fixed infrastructure that stores, retrieves and distribute topics and queries. The LMs have full knowledge of the LM topology, that is, they know about the existence of the other LM's and know the virtual graph that represents the LMs.

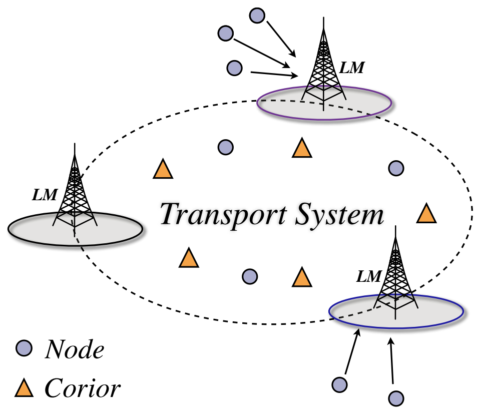

Fig. 1. Hikari Main Distribution System

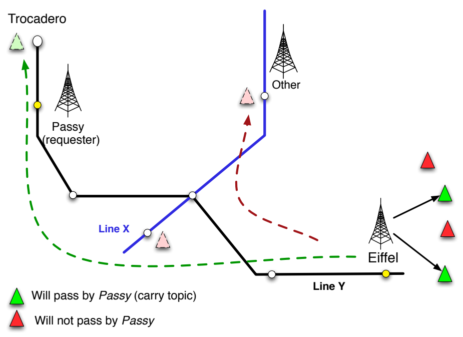

Fig. 2. Example scenario for MDS

Node: People or portable devices that request messages and provide content to LM's.

coreor: Special node that takes part in the dissemination of queries or data between LM's. Any node can choose to become a coreor. There are static coreors, which are coreors that have a predefined movement, like buses or trains, and the normal coreors which chose to be coreors and do not have fixed movement patterns, as for example, people or mobile devices carried by people.

The LM's are located in "stations" (e.g. train stations, disaster assembling areas) and they are simple access points with storage and a basic processor/controller power. Figure 1 shows the main distribution system. In such system there are the external nodes that aggregate in stations. Each station has at least one LM. The transport system is composed by nodes and coreors that transit between stations. You can think of this system as a metro system where metro stations are represented by ellipses in figure 1 and each station has LMs. Nodes are people, and coreors, which distribute messages in the system can be either people or trains.

C. The Hikari system in action

Suppose there are two metro lines and several stations, each one with one LM, as shown in figure 2.

There are two phases in this system:

1) The query phase:: Imagine that a node goes to station *Passy* and asks the LM *Passy* for the topic "slashdot", which is an electronic tech magazine. If *Passy* does not have the topic, it will query for the topic to a random number of coreors that use *Passy* to go to other stations. The query will have the topic name and the requester (e.g. slashdot:Passy). Imagine one of these coreors goes to station *Eiffel*. This coreor will deliver the query to *Eiffel*.

2) Message distribution phase:: If *Eiffel* does not have *Slashdot*, it will query nodes that are using the station and search for the topic. If the TTL of the message expires, it will be purged from the system. Otherwise, if the topic is encountered with one of the nodes using the system, *Eiffel* LM will try to find coreors going to *Passy* to deliver the message. Using the Hikari algorithm, for coreors that use *Eiffel*, *Eiffel* LM will have information about where coreors are going. *Eiffel* will choose a certain number of coreors that go directly to *Passy* or that pass by *Passy* in their paths (if they go for example to *Trocadero*). One of these coreors will deliver the message to *Passy*.

This system does not aim at delivering the message to the original requester node. Instead it aims at delivering it to any node that request for the same information (including the original one, if it returns to the same station).

D. How does the system know where coreors are going

One of the most important parts of the Hikari system is to know where coreors are going. Messages are given to coreors for distribution in the system, according to their paths and destinations. Depending on the type of node the coreor is, there are 3 ways for the system to know where the coreors are going, namely: Static lists, coreors' announcements and prediction systems.

1) Static lists: If coreors are nodes that have a fixed traveling routes that are know beforehand, the paths these coreors will take can be fed to the system as static list. Examples of such coreors are buses and trains, that have a fixed and known path between the stations.

2) coreors' announcements: This method can be divided into 2 areas: The ability to learn and user configured destinations.

- Ability to learn: Devices can have the ability to learn about their own paths. For example, a device carried by a person, can monitor which stations this person use on a specific day of the week , for example, Monday, for 4 weeks. After this monitoring period, if the person usually follows the usual path, the device can predict with a high probability where the person will go on Mondays. It can then try compare with the real movement on that day and adjust accordingly the probability value for that path on that specific day.
- User configures destination: In here is a user decides to become a coreor, or it is a regular coreor, she can pre-configure the path she is taking on a specific time. This can be tricky tough, because the system needs to trust the user, and there is a possibility that the user lies or misconfigures her path.

3) Prediction systems: Similar to the device's ability to learn, the system can keep track of nodes and their destinations. Otyi [1] and Routing in a Cyclic Mobispace [10] proved that nodes follow a cyclic movement path. An agenda based system like Otiy which predicts where nodes are located on a given time, can be used to predict coreors' paths based on an agenda recorded by the system.

III. Message semantics

Messages are information requested by nodes that is sent in the network. Examples of messages are: electronic newspapers, weather information, prices information for remote villages and mailing lists messages. We consider two main features that contribute to eliminate redundancy and loops in the system: Time to live and Popularity rate.

Time to live (TTL): Messages have a TTL counter that represents the lifetime of the message. For example, weather information is usually important before a given day. Therefore, when topics are created, they can be assigned a TTL.

Popularity rate(PR): This is a counter for allowing only popular messages to exist in the system or on a given LM. If a message is requested by many nodes, the popularity rate will increase and it will be kept on top of a queue. For a number of total requests, if the topic is not requested, its popularity rate will decrease. If the PR of the message is low, it will be eligible for purging even if its TTL is high.

IV. Scenario

We have built a tool that created a graph of the full Paris subway system [13] and used this graph as a scenario for our simulations. There are two reasons why we chose to evaluate our assumptions in such system: The first one is that we wanted to use a scenario very close to the real world, for testing our assumptions. The second reason is that although deterministic, this is a complex scenario that we can use to test various conditions that can arise in a message distribution system. The Paris metro system has more than 300 stations, 6000000 nodes per day and thousands of trips between stations. If our assumptions work in this scenario, they will hold for simpler scenarios, like developing countries and disaster stricken areas where traffic is low, there are fewer stations, the frequency of buses/trasnport between stations is low and there are not so many decisions to be taken in the stations' LMs. The subway scenario also allows to test the system, varying the number of coreors and nodes, and in the case of having access to the real traces of people going in and out of the system, these traces can be used for generating results that are close to the day-to-day reality.

For modeling the Paris subway system we considered 3 main components:

Stations: We have built a complete weighted graph of the Paris subway metro system, where the nodes are the stations and the edges of the graph represent the lines between the stations. The edges' weight is the time between stations. Each station has at least one LM. If there are more lines passing by the station, we consider the number of LMs being the same as the number of lines passing by the station.

Commuting stations: For stations that have more than one metro line, we have divided the station node into several nodes corresponding to different points in the same station. That is, for a station which has L number of lines passing by it, it will have N number of nodes in the the graph and $E = (\sum_{i=1}^{N} i - 1) * 2$ edges between the nodes. These edges' weights are the commuting time between areas in the same station, that is, the time it takes to commute from one line to the other.

Trains: This represents the trains that exist in each line. There is a delay between train arrival, and each train is given a name according to the time it reaches the first station in the line.

Nodes: Node arrival in stations is a Poisson distribution. Nodes that are in stations board first train that will take them to their destination using the shortest path in the graph, calculated using Dijktra's algorithm. Nodes that arrive in their final destination are removed from the system.

Messages: Messages are generated and placed in station LMs for distribution.

V. The PITS simulator

We have built a modular discrete simulator called Public Infrastructure Transport Simulator for simulating DTN publish/subscribe systems using support infrastructure. This is a simulator for general use that can handle multiple scenarios and different types of conditions. It runs in any machine that has a Python [5] interpreter. In order for it to be general purpose we decided to separate the simulator's core from the scenario and node generation. Therefore the simulator core is simple and only aimed at running the simulation itself once fed with lists created beforehand. We have included with PITS, tools for generating the "simulation world" including the graph for the desired scenario, nodes' lists, trains' lists and messages' lists.

A. *PITS components:*

PITS is composed by several modules that interact to create the world and to run the simulation:

Node factory: A list with nodes' arriving at different times, according to a Poisson distribution is created beforehand. The list contains a 5-tuple with the elements (initial station, coreor's name, commuting and final stations, path to destination, time of creation).

Train factory: A list with trains separated by an arrival delay in each line. The list is composed by 3-tuple elements with the format (current train, current station, current time).

Message factory: A list containing messages for delivery to the requester LM(s). The list contain 5-tuple elements with the format (Message ID, Requester station/LM, time of creation, TTL, station of origin)

Object factory: Analyses the lists of trains, nodes, lines and messages and creates and destroys objects as required be the core.

Core: Takes the lists created previously by the train factory and runs the simulations on a discrete way. PITS analyses the list and calls the appropriate modules to add/remove nodes to the trains and stations, deliver messages, perform housekeeping and management of LMs, and simulate movement in the system.

VI. Simulation

We have used the PITS simulator to analyze the Hikari algorithm and to compare with common DTN algorithms for message distribution. The simulation results show that Hikari way of message distribution has the same efficiency as an epidemic style message distribution in stations and it creates less replicas.

A. *Algorithms*

We compared the Hikari algorithm used for message distribution to coreors to two algorithms that are broadly used to test message delivery metrics: epidemic message distribution and random message distribution. These algorithms are used for in the distribution of messages to coreors, to be transported to requester LMs. Therefore, when referring to the use of these algorithms, the aim is not to use every node, in an epidemic way for example, for message dissemination, but the distribution of messages to coreors in the station that has the messages. Many of the algorithms proposed in the field of DTN, for example [2], [15], are aimed at node-to-nodes exchange of messages. Since our system does not require complex routing, we have used the basic algorithms that used in DTNs for analysis and for building other algorithms.

Epidemic message distribution: Based on the epidemic algorithm [16] , messages are distributed to coreors in an epidemic way. LMs that have a message requested by other LM's, distribute the messages to every coreor in range (both using the station and inside trains).

Random message distribution: Given an integer constant *N*, LMs with messages for distribution, select a

N number of coreors randomly and distribute messages to these coreors.

Hikari algorithm: Uses coreors' path information for message distribution. For coreors using the station, the LM will ask for their paths and if the coreor passes for a station where a message has to be delivered, a copy of the message will be pushed to this coreor for delivery. For the other coreors that go to different directions, no replica will be pushed.

B. Methodology

We compared the Hikari message distribution algorithm with the epidemic and random message distribution algorithms. Our main goals were to compare the performance between these algorithms to validate Hikari. Since epidemic will achieve the best message delivery rate in such system, we used it as a reference for comparing with the Hikari algorithm. The drawback of epidemic though, is that it creates a lot of replicas in the network and we wanted to overcome this problem with Hikari, since when using portable devices, buffer space is an issue. Random works better for many cases where the nodes' destination is not known. In this system we want to analyze the effect of random algorithm and compare it to Hikari to verify it's performance.

We have used 3 metrics for comparisons: Number of replicas in the system, number of replicas delivered to a requester LM and the efficiency.

For the simulation related with performance and message delivery, we have used the full Paris subway system, 25000 coreors which are people, and not static coreors like trains, with a simulation time of 160 min, and an arrival rate of coreors in the stations with a Poisson mean of $\lambda = 4$ for each station which corresponds to around 6% of the theoretical number of nodes that arrive in stations in each 5 minutes in the Paris subway system. The delay between trains is the average delay in the Paris subway system, which is 5 minutes, and we considered a constant commuting time between lines in a station of 5 min. We used 10 unique messages that appear on time t=0 and have a infinite TTL, so they will not expire during this simulation. The messages' request and originator were generated randomly. We do not consider the query phase in this simulation, especially because the spreading of the query is trivial (random nodes that go to random destinations). Therefore, we consider that the requester of the message is the LM that initiated the query and the originators of the messages are the LMs which received the query, found the message and are now trying to forward it to the requester LM. We consider that the information of nodes are going is 100% accurate, that is, no coreors change their minds during their way and no message is lost. We did not consider the effect of traffic in different times of the day in the present simulation and we did not use node-to-node exchange of information.

We also decided to test the system for unpopular stations, that is, stations that are not used often, or stations situated in far extremities of a line. If you take a long line for example, the probability that a coreor will go from one extremity to the other is very small, since usually nodes commute in intermediary stations. In the case of static coreors, like trains or buses, this is not a problem since there is a guarantee that there will always be traffic between any station. But if humans or mobile devices are used the delivery rate is expected to be low. For this experiment, we have used 10 unique messages originated in extremities of station in different lines, and destined to requester LM's in the other extremity of the same line.

Both of these algorithms expect that coreors with messages will eventually go or pass for the stations where the message was requested.

We ran a second set of simulations for analyzing the effect of number of coreors in the system. Since one of the main components of the system are the coreors, we analysed the effect of number of coreors in the system. For this, we have varied the arrival rate of coreors in the stations. Higher arrival rate (larger Poisson λ) means that more coreors will be in the system.

For the evaluation of we have used 50000 coreors, 100 unique messages, randomly distributed in the stations and we have used λ values of 2, 3, 4, 5, 6, 7, 8, 9 and 10. The simulation ran for 180 minutes which is the value that we have chosen for all the simulations. We have measured the replica delivery rate and number of replicas in the system.

VII. Preliminary results

A. Algorithm comparison and message dissemination analysis

Figure 3 shows the number of replicas delivered to a requester LM. The total number of messages to be delivered, calculated as the number of coreors that transit between message creator and requester LM, is 177. The figure shows that Hikari has the same delivery rate as Epidemic, which is 175, representing 98.9% of the total number replicas that are possible to deliver. This is an expected result since Hikari delivers the messages to coreors that it knows will go to the requester LM. In the case of loss in the system, or of coreors changing their routes in the system, it is expected that epidemic might have a slight better performance that Hikari, since some nodes that change routes may go to requester LM's that they did not expect to find in the path announcement/discovery phase. Random has a poor delivery rate, since only 16 replicas are delivered, which corresponds to around 9% of total replicas delivered. Therefore, for the relation of replicas created vs

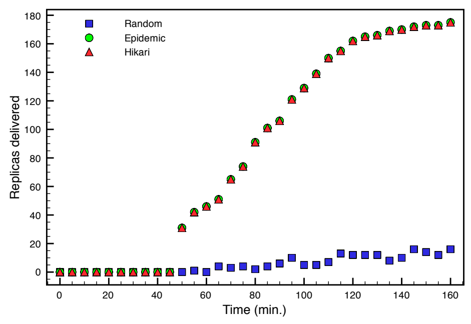

Fig. 3. Replicas delivered vs time

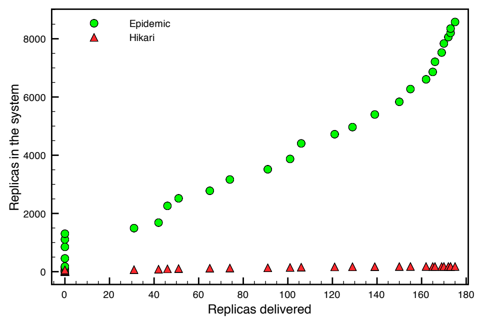

Fig. 4. Relation replicas in the system vs replicas delivered

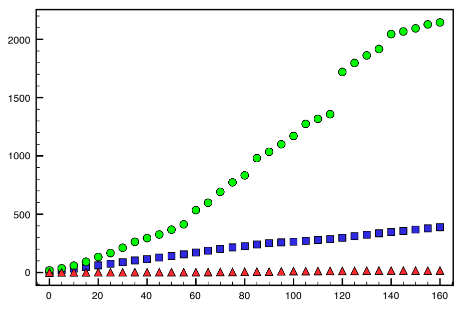

Fig. 5. Relation replicas in the system vs replicas delivered

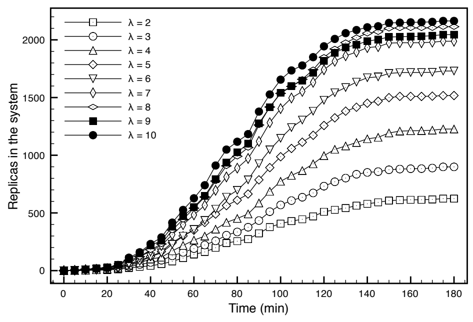

Fig. 6. Replicas in the system for different arrival rates

replicas delivered we did not consider the effect of the Random message delivery. Figure 4 shows this relation, and we can see that for 175 replicas delivered, Hikari creates around 177 replicas, while Epidemic creates 8577 replicas. This is an encouraging result because it shows that Hikari can achieve a result close to Epidemic for message delivery, while creating fewer redundant replicas.

In Figure 5 we analyze the delivery rate in unpopular station. The number of replicas created in the system is: Hikari: 17, Epidemic: 2145, and Random: 388. The Figure shows that the the delivery rate is quite low (only 10 replicas delivered), which means that when mobile nodes or persons are used as coreors, some improvement should be introduced.

B. Arrival rate in stations

For 50000 coreors we have varied the Poisson λ arrival rate in the stations, from $\lambda = 2$ to $\lambda = 10$. In the plots we have used lines to connect the dots for better visualization.

Figure 6 shows the effect of varying the arrival rate in the stations for the replicas in the system. As we can see, having a higher arrival rate, which means more coreors in the system, creates more replicas in the system. Using 50000, for small numbers of arrivals, the number of replicas increases in large steps, while, for higher number of arrivals, the number of replicas increases in a more or less constant manner. In fact, this is due to the fact that for smaller arrival rates, the number of coreors in the station increases slowly, and therefore even using 50000 nodes, for a simulation time of 180 minutes, not all 50000 coreors are in the system if the arrival rate is low. For $\lambda = 2$ and $\lambda = 3$ the number of coreors in the system after 180 minutes of simulation is lower than 50000 coreors.

Figure 7 shows the relation between the replicas delivered and the arrival rate in the stations. From the plot, we can see that the delivery rate of replicas increased substantially when more coreors are in the system. There is also the observation that more replicas are delivered in the beginning. With this result, we can say that if the number of coreors is high, replicas can

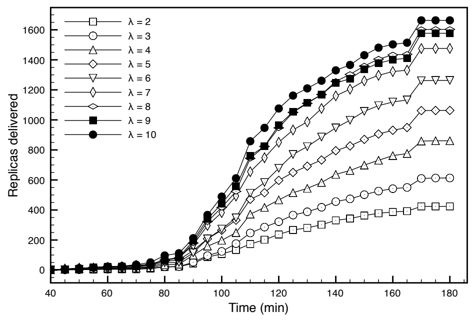

Fig. 7. Replicas delivered in the system for different arrival rates

be delivered for a short time in the beginning, since more replicas will be created, and there will be enough redundancy in case some replicas are lost during the transport. In other words, the plots show that after 30 into the simulation there are around 23 replicas for a λ of 2 while there are around 112 replicas for a λ of 10. That means that with a lower λ , in order to compensate for any loss or to have enough redundant messages, we have to keep distributing messages in the system for some time. With a higher λ more replicas are distributed in a short time, and therefore we only need to distribute the replicas for a short time, thus releasing resources in the LMs and creating less chattiness in the system.

The final intuition is that having a higher Poisson λ contributes to the creation of more replicas in the system and faster, thus increasing the delivery rate of replicas. Nevertheless although having redundancy is good to assure the delivery of messages in the case of loss, having too much redundancy in the system in not good, since it consumes resources in the LMs side and in coreors, since they have to carry more messages. In order to address this problem, we can use one of characteristics notices in this experiment to design a better algorithm. Since replicas are delivered in a short time, to more coreors when λ is higher, when setting up a minimum number of coreors to carry messages in the system, having a high λ will make the delivery of messages faster to the requesting LMs. In other words, having a high λ only a short time will be needed for the LMs to distribute messages to an optimal number of coreors, and then stop the distribution, this saving resources and preventing the creation of too much redundancy. When having a small λ, more time will be needed to distribute messages to an optimal number of coreors, thus holding resources in the LM while not creating enough redundancy in the system (enough redundancy can compensate for losses in the system). These experiments also rose our interest on how a different number of coreors in different stations affects the system. For example, the question of if having a larger number of λ in popular stations helps with the information dissemination. We plan to analyse these issues in future work in order to make the system better and faster.

VIII. Related work

The area of Delay Tolerant Networks has been gaining a lot of attention lately, specially in areas like sensor networks, disaster communications and communication for developing regions. In DTNs communication between two ends in not assured and nodes are used to forward packets. Our approach falls in the area of DTNs. he DTN Research Group [6] has been working on an architecture that implements store-and-forward message switching by overlaying a layer, the bundle layer, on top of heterogeneous region-specific lower layers [dtntutorial]. This architecture takes in account aspects like routing in DTNs, addressing and bundle management. This architecture uses a unicast communication model, where information about source and destination needs to be defined before the information is transmitted. The architecture uses endpoint identifiers [3] [4] for routing, and it aims at reliable routing of packets between a source and destination. Our systems focus is distributing messages among mobile nodes using some infrastructure for support. Also, our use of identifiers is loose, since coreors themselves are anonymous and the messages themselves provide the information for the forwarding mechanism. Therefore, our aims is different from the works with the bundle layer, since these works are aimed at a broader, general scenario.

The Daknet [12] and Kiosknet [7] projects use a mechanical backhaul to ferry messages between villages and to/from internet gateways/hubs. [12] pioneered the use of transportation infrastructure to carry packet between villages. It uses buses, motorcycles and ox-carts that carry mobile access points (MAP) and exchange data with kiosks in villages using a proprietary protocol. [7] addresses the problem of low cost reliable communication for kiosks in rural villages. The system is a comprehensive solution that encompasses naming, addressing, forwarding, routing, identity management, application support and security. In both [7] and [12] , Kiosks are central points that maintain information about users, and it is the point where users make and receive requests. Our system differs from [7] and [12] in terms of goals and the methods used. In our system we are not aiming at taking internet to disconnected regions, but instead we are aiming at a publish/subscribe system that works on DTN where persons can generate content or subscribe to it. These systems depend heavily on addresses and maintaing identifiers because of the complex requirements, while in our system, identifiers other that the few ones used by LM's, are not used. In our system, the most important point for forwarding a message is the content/topic of that message.

The work on Delay Tolerant Broadcasting [9] proposes an open, receiver-driven broadcast system for information dissemination using one-hop data transfers. [9] is purely a broadcast system, while Hikari is a message distribution system that supports queries and storage.

Peoplenet [11] aims at creating a distributed system to distribute and match queries originated by mobile nodes. The system uses the concept of bazaars which are pre-determined geographic regions that handle a specific type of query (e.g. sports). When a user sends a query, it propagates to the appropriate bazaar, via a cellular infrastructure, and it is distributed to a k number of nodes associated with that bazaar. This nodes in turn spread the query to other nodes. [11] is a query based service where queries are sent to people, based on individual interests, that uses the cellphone infrastructure. While [11] is aimed at cities with a good cellular infrastructure, our system is a more generic system that uses the free wi-fi band and the devices used can be of different types, for example, PDAs, Cellphones with short range communication radio and computers. Also, in our system, nodes do not have identifiers (like cellphone numbers) since the messages are destined to anyone interested in a topic and LMs retrieve topics from anonymous nodes.

TACO-DTN [14] aims at a time-aware approach to DTN content based dissemination. The main idea is to use temporal functions for subscriptions and events, and temporal functions for topics. The main contribution point of this work is to prove that by using the temporal functions, relevant information can be correctly distributed to infostations and the delivery rates improve compared to random mechanisms. Topics that are of interest of a group in a certain infostation are sent to that infostation. In Hikari the way the content is distributed to the LM's is not considered, since LMs ask for topics when there are requests from nodes. Also in [14] the scenario is that of infostations with some sort of direct connectivity with a content server or/and other infostations, while in Hikari, although there is connectivity between LM's this connectivity is "virtual", and coreors are the links between the LMs.

IX. Conclusions

We proposed and described the Hikari system, a DTN publish and subscribe message distribution system. The Hikari system is a solution for providing connectivity to regions that are not always connected to a communication infrastructure. We use the mobility of user nodes for message distribution/dissemination. Unlike other DTN systems, the proposed Hikari system does not use the identifiers of nodes for message forwarding decisions. By means of this approach, we can eliminate the technical complexity related to the identifier maintenance and its distribution. In the Hikari system, the nodes do not need to know any information about other nodes. Simulation results showed the Hikari system achieves a message delivery rate, which is close to that of epidemic message delivery system (when there are no loss of messages in the system), while generating less unnecessary redundant replicas in the system. We also noticed from the simulations that a higher number of coreors in the system contributes to a faster dissemination of information. Although more replicas are created in the system in a short time, tuning the Hikari algorithm to only distribute a certain number of replicas, can help decrease the chattiness of the system while saving resources. There are some unresolved technical issues that we have been identified in the Hikari system. The Hikari system does not have an optimal performance when requesting stations are unpopular stations. If there is some information in an unpopular station it will be difficult to disseminate this information, especially to other small stations. There is also the issue of number of coreors in the system. The simulations showed that, for a small number of coreors, the system will not achieve a good delivery rate. We plan, as future work, to analyze the effect of loss in the system and to address the unpopular stations issue by used large stations or direct node communication inside trains for helping disseminate the information.

References

[1] M. Boc, A. Fladenmuller, and M. D. de Amorim. Otiy: locators tracking nodes. In *CoNEXT '07: Proceedings of the 2007 ACM CoNEXT conference*, pages 1–12, New York, NY, USA, 2007. ACM.

[2] J. Burgess, B. Gallagher, D. Jensen, and B. N. Levine. Maxprop: Routing for vehicle-based disruption-tolerant networks. *INFOCOM 2006. 25th IEEE International Conference on Computer Communications. Proceedings*, pages 1–11, April 2006.

[3] V. Cerf, S. Burleigh, A. Hooke, L. Torgerson, R. Durst, K. Scott, K. Fall, and H. Weiss. Rfc 4838- delay-tolerant networking architecture. RFC, April 2007.

[4] K. Fall. A delay-tolerant network architecture for challenged internets. In *SIGCOMM '03: Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 27–34, New York, NY, USA, 2003. ACM.

[5] P. S. Foundation. The python programming language website. http://www.python.org/.

[6] D. R. Group. Dtn research group web site. http://www.dtnrg.org.

[7] S. Guo, M. H. Falaki, E. A. Oliver, S. U. Rahman, A. Seth, M. A. Zaharia, and S. Keshav. Very low-cost internet access using kiosknet. *SIGCOMM Comput. Commun. Rev.*, 37(5):95–100, 2007.

[8] P. Juang, H. Oki, Y. Wang, M. Martonosi, L. S. Peh, and D. Rubenstein. Energy-efficient computing for wildlife tracking: design tradeoffs and early experiences with zebranet. *SIGARCH Comput. Archit. News*, 30(5):96–107, 2002.

[9] G. Karlsson, V. Lenders, and M. May. Delay-tolerant broadcasting. In *CHANTS '06: Proceedings of the 2006 SIGCOMM workshop on Challenged networks*, pages 197–204, New York, NY, USA, 2006. ACM.

[10] C. Liu and J. Wu. Routing in a cyclic mobispace. In *MobiHoc '08: Proceedings of the 9th ACM international symposium on Mobile ad hoc networking and computing*, pages 351–360, New York, NY, USA, 2008. ACM.

[11] M. Motani, V. Srinivasan, and P. S. Nuggehalli. Peoplenet: engineering a wireless virtual social network. In *MobiCom '05: Proceedings of the 11th annual international conference on Mobile computing and networking*, pages 243–257, New York, NY, USA, 2005. ACM.

[12] A. Pentland, R. Fletcher, and A. Hasson. Daknet: rethinking connectivity in developing nations. *Computer*, 37(1):78–83, Jan. 2004.

[13] RATP. Ratp website. http://www.ratp.com/.

[14] G. Sollazzo, M. Musolesi, and C. Mascolo. Taco-dtn: a time-aware content-based dissemination system for delay tolerant networks. In *MobiOpp '07: Proceedings of the 1st international MobiSys workshop on Mobile opportunistic networking*, pages 83–90, New York, NY, USA, 2007. ACM.

[15] T. Spyropoulos, K. Psounis, and C. S. Raghavendra. Spray and wait: an efficient routing scheme for intermittently connected mobile networks. In *WDTN '05: Proceedings of the 2005 ACM SIGCOMM workshop on Delay-tolerant networking*, pages 252–259, New York, NY, USA, 2005. ACM.

[16] A. Vahdat and D. Becker. Epidemic routing for partially connected ad hoc networks. Technical report, Duke University, 2002.