Reputation system for User Generated wireless podcasting

Hu, Liang; Dittmann, Lars; Le Boudec, J-Y

Published in:
Australasian Telecommunication Networks and Applications Conference, 2008. ATNAC 2008.

Link to article, DOI:
10.1109/ATNAC.2008.4783353

Publication date:
2008

Document Version
Publisher's PDF, also known as Version of record

Reputation system for User Generated wireless podcasting

Liang Hu [#1], Lars Dittmann [#1], Jean-Yves Le Boudec [*2]

[#1] *Network Technology and Service Platform, Technical University of Denmark*
2800 Lyngby, Denmark
[1] liang.hu@fotonik.dtu.dk
[3] lars.dittmann@fotonik.dtu.dk

[*2] *I&C, Swiss Federal Institute of Technology, Lausanne*
1015 Lausanne, Switzerland
[2] jean-yves.leboudec@epfl.ch

***Abstract*—The user-generated podcasting service over mobile opportunistic networks can facilitate the user generated content dissemination while humans are on the move. However, in such a distributed and dynamic network environment, the design of efficient content forwarding and cache management schemes are challenging due to the lack of global podcast channel popularity information at each individual node. We design a distributed reputation system at each node for estimating the global channel popularity information which is significant for forwarding and cache management decision. Our simulation result shows that, compare to History-based Rank scheme, our reputation system can significantly improve system performance under Community-based Random Way-Point (C-RWP) mobility model and localized channel popularity distribution.**

I. INTRODUCTION

In recent years, mobile opportunistic network has become an attractive research area for networking small mobile devices carried by human being, vehicles and animals. Besides unicast content delivery, the broadcasting in opportunistic network has been proposed in Podnet project [1] to provide seamless content distribution beyond infrastructure network. In this paper, we focus on designing popularity-based content forwarding and public cache replacement schemes in User Generated Wireless Podcasting (UGWP) service over the system architecture of ad-hoc podcasting [1]. We mainly target at obsolete podcasting service where only the most recent content is of interests and old content is always obsolete by the latest one e.g. short news report distribution or software updates of mobile devices. In UGWP, obtaining popularity information of podcast channels is significant for the content forwarding and public cache replacement decisions. Unlike existing Internet-based user generate service such as YouTube [2] where the content popularity information is made centralized, in ad-hoc podcasting, the channel popularity information is fully distributed throughout the network and dynamic due to nodes' mobility. Thus it is much more difficult for each node to obtain and predict popularity information of global channels. With inaccurate channel popularity information, node may forward the content that future encounter nodes are not interested in. Ultimately, this would lead to low hit ratio of content retrieve, low utilization of both the node contact opportunities and cache storage.

In this paper, we design a distributed reputation system based on Bayesian framework through which each node can estimate the global channels popularities. The popularity of channel is represented by the reputation rating. The reputation system consist of three parts: Firstly, the reputation rating of channels at each node is built and updated by the number of requests to each channel from encounter nodes. This is called the first hand information of channel popularity by each node's direct observations. Secondly, reputation rating is also updated by integrating its encounter nodes' direct observations which is called the second hand information of channel popularities. By dong so, node can learn and adjust popularity information of channels from observations made by others even before having to learn by own experience. By nodes gossiping the channel reputations, the accurate channel popularity information can propagate much faster throughout the network, especially when the popularity distribution is non-uniform and localized. Moreover, to protect against rumor spread from liars, the second hand information is only accepted if a deviation test is passed. Thirdly, to adapt the channel popularity shifts, both the first hand information and the reputation ratings of each channel decays after each contact. The previous observations are gradually forgotten while more weight is put on recently observations.

To the best of our knowledge, our work is the first attempt to employ Bayesian Framework based reputation system for estimating the content popularity in the context of user-generated opportunistic content dissemination. Previous, the Bayesian framework based reputation system has been employed in coping with misbehaviors in mobile ad hoc networks [3]. The paper is organized as follows: in section 2, the concept of Bayesian Framework based Reputation is introduced. In section 3, the protocol specification and data structure of reputation system is described. We evaluate the

performance of reputation system by discrete event simulation in section 4. Section 5 concludes the paper.

II. Bayesian Framework Based Reputation System

To implement Bayesian framework based reputation system, both first hand information and reputation ratings are needed. First hand information is the direct observations of channel popularity and can be passed to other nodes as second hand information. Reputation rating is the channel popularity information taking accounts both first hand information and second hand information by node's encounter nodes' direct observations. In this section, we introduce how both first hand information and reputation rating is built and updated.

A. First hand information by modified Bayesian approach

The first hand information for the popularity of channel j at node i is defined as

$$F_{i,j} = (A, B)$$

It represents the parameters of the Beta distribution assumed by node i in its Bayesian view of the popularity of channel j. Initially, it is set to (1, 1). The standard Bayesian method [3] gives the same weight to each observation regardless of its time of occurrence. However, the popularity of a given channel may change as the node may move between different communities of different content popularity characteristic. For this reason, we add a reputation fading mechanism to give less weight to the past observations, because the latest observations would be more important for estimating current and future popularity of the channel. Assume node i makes one individual observation of channel j during a contact with encounter node. Let s=1 if channel j is requested by the encounter node, and s=0 otherwise. The update is as follows:

$$A := uA + s\,; \quad B := uB + (1 - s)$$

The weight u is a discount factor for the past experiences, which serves as the fading mechanism.

B. Reputation Rating and Model Merge

The reputation rating $R_{i,j}$ is also defined by two numbers:

$$(\alpha\ ,\beta\ )$$

Initially, it is set to (1, 1). It is built and updated on two types of events: (1) when first-hand information is updated by own observations; (2) the second hand information from encounter nodes are accepted and copied. There are two variant of using second hand information from encounter nodes: direct observations (first hand information) from encounter nodes and reputation rating from encounter nodes. For event type (1), the update of reputation rating is the same for the first-hand information updating. Let $s \in \{0, 1\}$ is the observations:

$$\alpha\ := u\alpha\ + s, \qquad \beta := u\beta + (1 - s)$$

For the case (2), if we assume passing direct observations, the linear pool model is used to merge own reputation rating with direct observations passed from encounter nodes on the condition if the deviation test is passed. Deviation test is used to protect system against false rating from encounter nodes. The idea behind it is that humans only believe the opinions from others only if, to them, it seems likely i.e. it dose not differ too much from their own opinions. Moreover, even if they accepted opinions from others, they only attach less weight to other's opinions than their own opinions. We denote with E (Beta (A, B)) the expectation of the distribution Beta (A, B). Let the reputation rating of channel j at node i as follows:

$$R_{k,j} = (\alpha_j^k, \beta_j^k)$$

The first hand information of channel j at encounter node x:

$$F_{x,j} = (A_j^x, B_j^x)$$

The deviation test is as follows:

$$\text{If } \left| E(Beta(\alpha_j^k, \beta_j^k) - E(Beta(A_j^x, \beta_j^x)) \right| < \text{THS},$$

where THS is a positive constant (deviation threshold), then the deviation test is passed and we believe the report from node x is trustworthy. Then, α_j^k, β_j^k are updated by first hand observations of node x using the linear pool model merging:

$$\alpha_j^k = \alpha_j^k + w \bullet A_j^x\,; \ \beta_j^k = \beta_j^k + w \bullet B_j^x,\ 0<w<1.$$

III. Data Structure and Protocol Specification

The cache of each node consists of a private part for caching private interested channels and public part for caching public interested channels. Each node maintains a table of channel reputation ratings which is used for content forwarding and public cache replacement decisions. As an example, the reputation rating table of node A is as follows:

Reputation rating table at node A

Channel Feeds	First Hand Information	Reputation Rating	Latest Entry ID (entry name, time of publish)	Subscribed or Not (S/N)
1	A_1^A, B_1^A	α_1^A, β_1^A	-Weather forecast of Copenhagen -13th July at 10:00 am	S
3	A_3^A, B_3^A	α_3^A, β_3^A	-BBC news -10th July at 10:00 am	S
5	……	……	……	S
7	……	……	……	S
9	……..	……	……	S
……	……	……	……	N
M	A_M^A, B_M^A	α_M^A, β_M^A		N

Channel Feeds: Podcast channel identifications.
A_M^A, B_M^A: the first hand information of channel M at node A.
α_M^A, β_M^A: the reputation rating of channel M.
Latest Entry ID: the latest updates correspond to each podcast channel.
"S": the channel is subscribed by node A.
"N": the channel is not subscribed.

In brief, the reputation system based podcasting protocol works as follows:

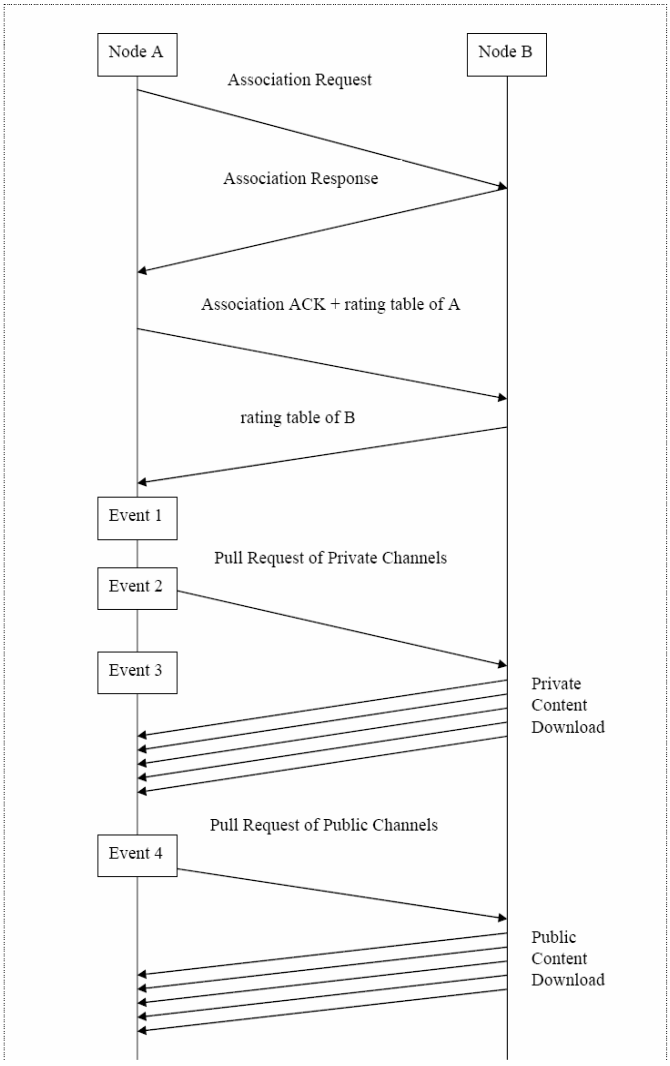
1. Idle node periodically broadcast association requests to its neighbors. If it discovers several neighboring nodes, it randomly selects one node to associate and establish pair-wise connection.

2. Node updates its reputation ratings of all channels by merging the second hand information from peer if deviation test is passed.

3. Node firstly pulls content of private interested channels.

4. Node updates both first hand information and reputation rating of channels by peer's requests of privately interested channels.

5. Node pull content of public interested channels based on estimated channel popularities using popularity-based forwarding and public cache replacement schemes. Various forwarding and public cache replacement schemes are described in [6]. For detailed description of protocol specification, see the message sequence chart below: (suppose node A and node B establish pair-wise association.)



Protocol Walkthrough

List of notations:

γ : set of channels in rating table of node A

λ : set of channels in the rating table of node B.

u: discounting factor for reputation rating and first hand information, 0<u<1

w: factor for linear merge of first hand and second hand information, 0<w<1

THS: threshold for deviation test (0<THS<1)

(α_i^A, β_i^A): reputation ratings of channel i of node A

(α_i^B, β_i^B): reputation ratings of channel i of node B

(A_i^A, B_i^A): first hand information of channel i of node A

(A_i^B, B_i^B): first hand information of channel i of node B

1) Node Association Phase:

Each node (In this example, we denote node A.) periodically broadcasts "**association request**" for neighbour discovery. Then one or several neighbouring nodes replies with "**association responses**" to nodes A. Upon receiving responses from neighbours, node A randomly select one of the neighbouring nodes (In this example, we denote node B) and send "**association ACK**" which is piggybacked the rating table of node A to the selected node (node B). Now node A and node B have established a pair-wise association. Reputation rating fading: α_i^A =u* α_i^A , β_i^B =u* β_i^B , A_i^A =u* A_i^A , $B_i^A = B_i^A * B_i^A$.

2) Exchange Rating Table Phase:

Node B sends rating table of node B to node A.

According to the Bayesian Framework based Reputation System, Node A updates its rating table by merging Node B's rating table.

If first hand information is passed on

for all channel $(i \in \lambda) \cap (i \notin \gamma)$ **do**

Add new channel feeds i to the rating table of node A

Initialize $\alpha_i^A=1, \beta_i^A=1, A_i^A=1, B_i^A=1$ (Prior distribution Beta (1,1))

Linear merge (α_i^A, β_i^A) with (A_i^B, B_i^B) if the deviation test is passed

if $(|\frac{\alpha_i^A}{(\alpha_i^A+\beta_i^A)} - \frac{A_i^B}{(A_i^B+B_i^B)}| <= \text{THS})$

$\alpha_i^A = \alpha_i^A + w * A_i^B$, $\beta_i^A = \beta_i^A + w * B_i^B$. (0<u, w<1)

for all channel $i \in (\lambda \cap \gamma)$ **do**

Linear merge (α_i^A, β_i^A) with (A_i^B, B_i^B) if the deviation test is passed

if $(|\frac{\alpha_i^A}{(\alpha_i^A+\beta_i^A)} - \frac{A_i^B}{(A_i^B+B_i^B)}| <= \text{THS})$

$\alpha_i^A = \alpha_i^A + w * A_i^B$, $\beta_i^A = \beta_i^A + w * B_i^B$. (0<u, w<1)

3) Pull updates of privately interested channels: node A pull new updates of privately interested channels which may include new channels learned from node B.

- Node A can request any privately interested channels from node B even when B dose not have the new updates of those channels. In this way, the channel popularity information can propagate faster to the network.

4) Download private content from node B.

Rearrange the private cache according to the cache management scheme below;

Update the "entry name and publish time" in the rating table of node A;

5) node A updates rating table according to the request of private channels of node B.

for all channel i ∈ {channels requested by node B}

$A_i^A = A_i^A + 1; \alpha_i^A = \alpha_i^A + 1;$

for all channel i ∉ {channels requested by node B}

$B_i^A = B_i^A + 1; \beta_i^A = \beta_i^A + 1;$

6) Pull updates of public interested channels

- The channels to be forwarded are selected according the public content forwarding scheme based on the reputation rating table.
- Only channels or entries that B has are requested

7) Download forwarding content

Download public content from node B.

Rearrange the public cache according to the cache management scheme

Update the "entry name and publish time" in the rating table of node A;

IV. PERFORMANCE EVALUATION

Based on simulation, we evaluate the performance of reputation system under "Community-based Random Way-Point" (C-RWP) mobility model and localized channel popularity distribution.

C-RWP captures the "clustering" effect of realistic human mobility: The mobility of nodes tends to be localized in certain geographical area where they frequently meet other nodes with similar social roles e.g. workmate, classmate; On the other hand, nodes only occasionally meet nodes with dissimilar social roles in other geographical areas. In C-RWP, nodes are divided into different communities. One community is a group of nodes with the similar mobility patterns. Nodes of one community move within the same square in a random way-point (RWP) model. Nodes of the same square have equal chance of meeting each other regularly while nodes of different squares can seldom meet each other or only occasionally meet near the borders of two squares.

Secondly, we assume channel popularity distribution. Based on the measurement results of YouTube, a recent paper [5] shows that: video clips of local interests only have a high local popularity; there is no correlation observed between global and local popularity. Along the line of their observations, we assume: firstly, one community of nodes have one group of interesting channels which is a subset of total global available channels. Among one community, the popularity of its group of channels follows Zipf-like distribution. Secondly, different communities have different groups of interested channels. One example could be one community is interested in the channels of English language while other is interested in channels of German language.

Thirdly, the location of the channel publishing nodes and its subscribing nodes could be as follows: (1) the publishing node and its subscribing are in the same community; (2) they are in two different communities which are partially or totally physically separated; (3) publishing node and some of its subscribing node are in the same community while other subscribing nodes are in other community. We focus on the scenario (2): due to physical separation of communities, nodes of one community may have difficulty of learning popularities of channels published from other communities.

A. Performance Metrics

To quantify the user satisfaction of user generated podcasting, the Recall and Delay are employed as the performance metrics of reputation system. Recall indicates the fraction of the chunks that are relevant to the node interets that are successfully received. It is borrowed from the area of Information Retrieve (IR). Delay: indicates the latency between the time when chunk is published and the time when it is received. Since we target at news-related content distribution, the average delay is curial to the end user satisfaction.

Average Recall:
The Recall of node i is defined:

$$R^{i}(t) = \frac{X^{i}_{R}(t)}{X^{i}_{P}(t)},\ i=0, 1, 2\ldots..N-1.$$

N: the total number of nodes; i: the node ID.

$X^{i}_{R}(t)$: total number of chunks that have been received by node i by time t.

$X^{i}_{P}(t)$: total number of chunks that have been published from all interested channels of node i, by time t. The average recall of all nodes N is defined:

$$\bar{R}(t) = \sum_{i} R(t) / N,\ i=0, 1, 2\ldots..N-1.$$

In this work, we are only interested in the average recall at the end of the simulation.

Average Delay: The chunk delivery delay is defined as $\Delta t = T_{publish} - T_{receive}$. $T_{publish}$ is the chunk publish time while $T_{receive}$ is the time when it is received. The average chunk delivery delay is calculated:

$$\frac{\sum_{i} \Delta T_i}{M} \quad i=1, 2, 3\ldots.M;$$

M is the total number of chunks received by all nodes at the end of simulation.

B. Simulation Settings

We compare the performance of Reputation System via passing direct observations with History-based Rank [1] under the scenario: two separated communities of nodes with two groups of interesting channels. The history-based rank method [3] is a method which estimate channel popularity only by first hand information in the form of number of encounter requests per channel. It works as follows: node keeps track of the channels that were requested by past nodes and maintains a history-based ranking. Only the requests for the channels that the requester is actually subscribed to are counted.

The performance evaluation is done with our own simulator which is based on a simple communication model: two modes can communicate with a nominal bit-rate if their geometric distance is smaller than a threshold value (that models the radio range of mobile device). The simulation model does not incorporate link layer issue such as collision or interference, since we simulate a sparsely connected network where the collisions or interference among different associations are very rare. For the simulation, we further assume that the setup time for nodes associations is negligible.

Figure 1: two separated communities of nodes

As indicated in figure 1, 100 nodes are grouped into two communities: A (blue) and B (yellow). The nodes are human beings who carry WiFi-enabled mobile device. Each community is interested in one group of popular channels among total 100 channels. Nodes of ID 0-49 belong to community A while nodes of ID 50-99 belong to community B. Both nodes of community A and B move within a square of the same side length 500 meters in Random Way-Point (RWP) model. The moving speed is constant 1 m/s with pause time 1 s. Each node publishes one channel, with the channel ID identical to the node ID, e.g. node 0 publish channel 0, node 1 publish channel 1. Community A publish channels from 0-49 while community B publish channel from 50-99. The content publish interval is identical for all channels which is 600 s. The nodes of community A are only interested in the channels published from community B (channel ID 50-99) while nodes of community B are only interested in the channel published from community A (channel ID 0-49). Each node is interested two channels. Among community B, the popularity distribution of channels 0-49 follows Zipf-like distribution with a=1.5, where the channel 0 is the highest popular channel, channel 1 is the second popular and so on. Define the popularity of channel 0-49 in community B:

$$P_i \sim \frac{1}{(i+1)^a}, \text{ i = 0, 1, 2….49}$$

Likewise, among community A, the popularity distribution of channels 50-99 follows the same Zipf-like distribution with a=1.5. Assume the channel 50 is the highest popular channel, channel 51 is the second popular and so on: Define the popularity of channel 50-99 in community A:

$$Q_i \sim \frac{1}{(i-49)^a}, \text{ i = 50, 51, 52….99.}$$

Nodes only associated pair-wise, even if more than two are within reach of one another. The reason is that the contact duration may be short and it is better to get high throughput by only sharing the transmission capacity between two parties than to get high connectivity. We assume the forwarding scheme is “Most” and public cache replacement scheme is also “Most”, since this combination works best under the perfect knowledge of channel popularity [6]. The channel popularity is represented by local reputation ratings. With Most forwarding scheme, node forward the content from the most popular channels to least popular channels until two nodes get disconnected through their mobility. With Most public cache replacement scheme, when public cache is full, the content of less popular channel is always replaced with content of more popular one. Other simulation parameters are summarized in table 1.

Table 1

Parameters of Reputation System	
THS	0.4
u	0.99
w	0.2
Other Parameters	
Cache size	2 GB
Public Cache size	60 MB
Chunk size	2 MB
Simulated time	12 hours

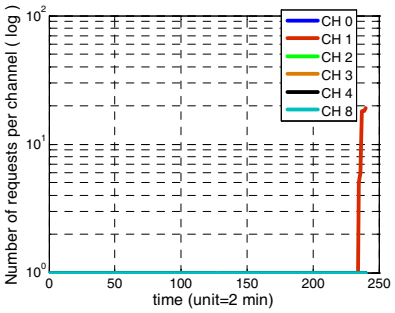


Figure 2: Number of requests per channel at node 60

From the figures 2 and 3, it is obvious that the history-based rank poorly estimates the popularity of channel 0,1,2,3,4,8. With history-based rank, node 60 cannot get any popularity information of channel 0,1,2,3,4,8 until 460 minutes. The reason is that node 60 cannot have enough first-hand information about channel popularity. In contrast, by using both direct observation and second hand information, reputation system can always perfectly estimate the popularity of channel 0,1,2,3,4,8 since the very beginning of the simulation, as showed in figure 3.

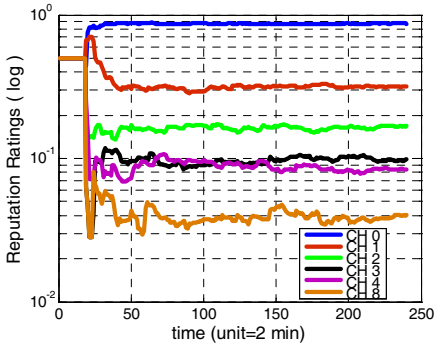


Figure 3: reputation ratings per channel at node 60

Without enough and accurate channel popularity information, nodes are not able to forward the channels of content which are interested by its future encounter nodes. History-based rank method achieves much lower average recall compare to Reputation System, as showed in the table 2. History-based Rank only achieves average recall 0.015 while Reputation System achieves 0.250. The performance gain of using

reputation system over history-based rank is more than 20 times. In terms of average delay, Reputation system also performs better than History-based rank, where reputation system achieves almost 400s of average delay less than history-based rank.

Table 2

	History-based Rank	Reputation System
Average Recall	0.015	0.250
Average Delay	1510 s	1112 s

V. Conclusion and Future Work

We design a distributed reputation system for estimating podcast channel popularity information in user generated wireless podcasting service. With reputation system, by nodes sharing their direct observations of channel popularities, the accurate channel popularity information can propagate much faster throughout the network, especially when the node mobility is community based and channel popularity distribution is localized. Our simulation results shows reputation system overwhelmingly outperforms history-based rank scheme in terms of average recall and average delay under a two-community C-RWP model and localized channel popularity distribution.

As the next step, we envision studying the performance of reputation system under a more realistic mobility model such as [4] which captures node moving both within the communities and between communities. Also, we intend to study the impact of liars on the performance of reputation system in user generated ad hoc podcasting.

References

[1] Vincent Lenders, Martin May, and Gunnar Karlsson. Wireless Ad Hoc Podcasting. In Proceedings of IEEE SECON, San Diego, CA, June 2007.

[2] http://www.youtube.com/

[3] Sonja Buchegger, Jean-Yves Le Boudec. A Robust ReputationSystem for P2P and Mobile Ad-hoc Networks. In Proceedings of Second Workshop on the Economics of Peer-to-Peer Systems, Harward University, June 2004.

[4] Mirco Musolesi, Cecillia Mascolo, Design Mobility Models based on Social Network Theory. Journal of Mobile Computing and Communication Review, Volume 11, Number 3.

[5] Michael Zink, Kyoungwon Suh, Yu Gu and Jim Kurose, Watch Global, Cache Local: YouTube Network Traces at a Campus Network - Measurements and Implications," In Proceedings of *2008* IEEE MMCN, San Jose ,CA, January 2008

[6] Liang Hu, Jean-Yves Le Boudec, Reputation System for wireless ad-hoc podcasting. Technical Report, Technical University of Denmark, July 2008.