Transmission Scheduling Algorithm in DTN

Wei Yu¹, Hongyan Li¹, Jiandong Li¹, Lin Jin¹, Kun Yuan²
¹State Key Laboratory of Integrated Service Networks¹, Xidian University, Xi’an, China
²Department of Automation², University of Science and Technology of China, Hefei, Anhui, China
Email: kunyuan@mail.ustc.edu.cn

Received June, 2013

ABSTRACT

Delay Tolerant Networks (DTNs) is a dynamic topology network, in which connection durations of each link are variable and paths between two nodes are intermittent. Most of protocols which are widely used in traditional wireless network are not suitable for DTNs. DTN adopts store-and-forward mechanism to cope with the problem of intermittent path. With limited storage of each node, it is a challenge for scheduling nodes’ transmission to avoid overflow of nodes’ buffers. In this paper we propose an optimal transmission scheduling algorithm for DTN with nodes’ buffer constraints. The object of the optimal algorithm is to get maximum throughput. We also present an algorithm for obtaining suboptimal transmission schedules. Our solution is certified through simulation, and it is observed that our solution can improve network performance in the aspects of avoiding overflow and increasing network throughput.

Keywords: DTN; Transmission Scheduling Algorithm; Throughput; Overflow

1. Introduction

Delay-Tolerant Networks (DTNs) [1] has characteristics of constrained storage, high delay and intermittent connections, which lead to no existence of end-to-end path between source and destination at most times. DTN approaches this problem with store-and-forward mechanism. The nodes, existing in the system operating on aforementioned mechanism, firstly store messages received from the upstream ones, and then wait for connection of outgoing links to send stored messages out. It’s very challenging for relay nodes with limited storage. Once the amount of stored messages meets the upper bound of storage, overflow potentially happens in the relay nodes due to a period of outgoing link disconnections. Since there is not enough storage space for new messages, relay nodes have to refuse new messages or discard old ones, causing energy waste and throughput decrease.

Although DTN networks are always mobile, in most cases, the node movements can be known in advance because of their predictable trajectories. There have been a lot of DTN studies based on prediction [2-4]. Consider an example deterministic network of five nodes {S₁, R, D₁, D₂} as shown in Figure 1, in which contact times are known or can be explicitly predicted. Contact times among the five nodes are shown in Figure 2(a), in which S₁ and R are connected from t = 2 to t = 5, and a link exists between R and D₁ from t = 10 to t = 13. Contact time between S₂ and R is [6, 8], and the one between R and D₂ is [7, 9]. Messages in S₁ need to be transmitted to D₁, but there isn’t an end to end path between them. So the messages should be firstly offloaded from S₁ to R in their contact time [2, 5] and then wait at R until t = 10, when the connection between R and D₁ begins. Similarly, S₂ sends messages to R at t = 6 and R relays messages to D₂ at t = 7. All the link rates are set to 1. Message forwarding sequences {S₁, R, D₁} and {S₂, R, D₂} are marked as path1 and path2 respectively. Contacts between nodes on both paths are periodic and the least common multiple of their periods is 14. Their periodicity isn’t shown in Figure 2 due to lack of space. The occupation of messages on R’s storage space has been depicted in Figure 3(a) with the storage of R unlimited. If the upper storage bound of R is set to 3, R is full at t = 5 and only until t = 10 able it is to free its storage as shown in Figure 3(b). So S₁ cannot send messages to R since their contact time begins at t = 6 and ends at t = 8. Under this situation, the contact times which can be utilized are shown in Figure 2(b) and the theoretical average network throughput is 0.214, i.e. 1×(5−2)/14=0.214. But if the contact time between S₁ and R is replaced with

![Figure 1. Network topology.](http://dx.doi.org/10.4236/cn.2013.53B2047)
Figure 2. Contact times among nodes.

Figure 3. Analytical results of occupations on storage space of relay node R. The black and blue lines represent the occupation on R of path1 and path2 with time varying respectively. The peripheral red line is the superposition of black and blue lines, which has overlaps with the black line in some positions. The green line of dashes represents the upper bound of R's storage.

[2,4], shown in Figure 2(c), to restrict the message transmission on path1, there will be enough storage space in R for message transmission on path2 as shown in Figure 3(c). After scheduling the contact times for transmission, the theoretical average network throughput increases to 0.286, i.e. \( [1 \times (4 - 2) + 1 \times (8 - 6)] \div 14 = 0.286 \), and no overflow happens.

[5,6] provide routing algorithm to realize optimal routing for pairs of nodes. But the transmission on one path may sacrifice throughput on other paths which share the same relay node with it since the relay's storage is limited, causing decrease of network throughput. The above-mentioned scenario includes more than one path. Such multi-flow problem has been studies well in connected works [7], but the algorithm is unsuitable in DTN due to its intermittent connection. [8-10] propose buffer management algorithm, which schedule message discarding and forwarding in relay node. [8] chooses discarding old messages first to catch up with network update. But outgoing link which belongs to the path for new messages transmission probably begins late, so relay node still can't free its storage and such solution does no help in throughput increase. Although [9,10] adopt other drop mechanisms, they still cannot avoid the same problem.

This paper proposed a transmission scheduling algorithm, which is based on the contact time features of links on different paths sharing the same relay node. This algorithm deals with overflow by restricting transmission on some paths from their upstream nodes. We only describe DTN networks of periodic pattern in order to get convenience when compare average throughput, but the transmission scheduling algorithm is also suitable for aperiodic pattern. Our solution not only optimizes the network throughput, but also cuts energy waste. It also can be used as a reference for routing algorithm.

The rest of the paper is organized as follows. We start with a model of network and transmission scheduling algorithms in Section II, and in Section III we analyze the model. Simulation results are presented in Section IV and followed by conclusion in Section V.

2. Model

In this section, we describe our model of network and how the network throughput is optimized through transmission scheduling algorithm.

2.1. Network Connectivity

Consider a network including \( n \) upstream nodes and \( n \) downstream nodes represented by the sets

\[
S = \{S_1, S_2, \ldots, S_n\} \quad \text{and} \quad D = \{D_1, D_2, \ldots, D_n\}
\]

respectively. \( S_i \)'s messages need to be transmitted to \( D_i \), which is \( S_i \)'s corresponding node in set \( D \). Since \( S_i \) and \( D_i \) aren't connected directly, messages transmission between them must pass \( R \), which is the only one
relay node in this network. So there are \( n \) ingoing links and \( n \) outgoing links. These links can be represented by link function which can show whether the link exists or not. Link function \( L_{in}^i(t) \) is defined for ingoing link between \( S_i \) and \( R \), and \( L_{out}^j(t) \) is for outgoing link between \( R \) and \( D_j \). Let \([t_{in}^i,t_{out}^i] \) denote the contact time between \( S_i \) and \( R \) in a period of \( T \). \( S_i \) and \( R \) can communicate to each other ingoing link exists during \([t_{in}^i,t_{out}^i] \). Similarly, \([t_{out}^j,t_{out}^j] \) denotes the contact time between \( R \) and \( D_j \). We can define \( L_{in}^i(t) \) and \( L_{out}^j(t) \) as:

\[
L_{in}^i(t) = \begin{cases} 1 & t_{in}^i \leq t \leq t_{out}^i \, \quad i = 1, 2, \ldots, n \\ 0 & \text{others in } [0,T] \end{cases}
\]

\[
L_{out}^j(t) = \begin{cases} 1 & t_{out}^j \leq t \leq t_{out}^j \, \quad i = 1, 2, \ldots, n \\ 0 & \text{others in } [0,T] \end{cases}
\]

Let \( v_{in}^i(t) \) denote rate function for ingoing link \( S_i R \), and \( v_{out}^j(t) \) is for outgoing link \( R D_j \). Rate function \( v_{in}^i(t) \) and \( v_{out}^j(t) \) are defined as:

\[
v_{in}^i(t) = v_{in}^i L_{in}^i(t) \, \quad i = 1, 2, \ldots, n
\]

\[
v_{out}^j(t) = v_{out}^j L_{out}^j(t) \, \quad i = 1, 2, \ldots, n
\]

where \( v_{in}^i \) is the rate of connected ingoing link \( S_i R \), and \( v_{out}^j \) is the rate of connected outgoing link \( R D_j \).

It is noted that there are \( n \) forwarding sequences, which can be represented by the set

\[
P = \{ S_1 R, D_1 \}, S_2 R, D_2 \}, \ldots, \{ S_n R, D_n \}
\]

in this network. Sequence \( \{ S_p R, D_q \} \) is marked as path \( p \), so we can also present \( P \) as \( P = \{ p_1, p_2, \ldots, p_n \} \).

### 2.2. Optimal Transmission Scheduling Algorithm

In this paper, our algorithm is able to avoid overflow in relay node and increase network throughput by altering lengths of contact times between nodes. We assume that all the messages received by \( R \) can be sent out in the period of \( T \), i.e., each outgoing link can undertake transmission task from its corresponding ingoing link. Let \( t_i \) denote the length of ingoing link \( S_i R \)'s contact time is shortened by. Accordingly, the length of outgoing link \( R D_j \)'s contact time is shortened by \( t_i v_{in}^j / v_{out}^j \) to meet the aforementioned assumption. Therefore the original contact times of links \( S_i R \) and \( R D_j \) in (1) are changed into \([t_{in}^i - t_i, t_{in}^i - t_i - t_i v_{in}^j / v_{out}^j] \) and \([t_{out}^j - t_i v_{in}^j / v_{out}^j, t_{out}^j - t_i v_{in}^j / v_{out}^j] \) respectively. The most important constraint here is that the sum of each path's occupation on \( R \) must be less than \( R \)'s maximal storage at any time. Let \( f_j(t) \) denote \( p_j \)'s occupation function on \( R \). Expression of \( f_j(t) \) is as follow:

\[
f_j(t) = \int_0^t [v_{in}^j(t) - v_{out}^j(t)] \, dt \quad 0 \leq t \leq T
\]

Therefore, the linear program computed the maximal network throughput \( Th \) is resulted as follow:

\[
\text{max } Th = \frac{1}{T} \sum_{i=1}^n \int_0^T v_{in}^i(t) \, dt
\]

\[
= \frac{1}{T} \left[ v_{in}^1 (t_{in}^1 - t_{in}^1 - t_i) + v_{in}^2 (t_{in}^2 - t_{in}^2 - t_i) + \cdots + v_{in}^n (t_{in}^n - t_{in}^n - t_i) \right]
\]

\[
s.t. \quad f(t) = \sum_{i=1}^n \int_0^t [v_{in}^i(t) - v_{out}^j(t)] \, dt \leq C_{max}^R
\]

\[
0 \leq t_i \leq t_{in}^i - t_{in}^i - t_i v_{in}^j / v_{out}^j
\]

where \( f(t) \) is a function representing the sum of all paths' occupation functions on \( R \), and \( C_{max}^R \) is \( R \)'s maximal storage.

It is important to note that this linear program is hard to solve due to its complex constraint, which must guarantee no overflow at anytime. So we also propose a simplified algorithm called transmission scheduling algorithm (TSA).

### 2.3. Transmission Scheduling Algorithm

We simplify this problem by assuming that \( f(t) \) only has one maximum beyond \( C_{max}^R \) during the period of \( T \) under the assumption of unlimited storage of \( R \), that is \( t_i \) for \( i = 1, 2, \ldots, n \) in (2). Let \( t \) denote the time when \( f(t) \) reaches its maximum, which is also the maximal value of \( f(t) \). So \( t \) must satisfy the following inequality:

\[
f(t') \geq f(t) \quad 0 \leq t \leq T, 0 \leq t' \leq T
\]

Therefore when the storage of \( R \) is set to \( C_{max}^R \), the overflow \( C_{over} \) can be figured out as follow:

\[
C_{over} = f(t') - C_{max}^R
\]

We shorten the contact times to decrease the sum of message transmission of all ingoing links by \( C_{over} \). We certainly make sure the real-time value of \( f(t) \) is no more than \( C_{max}^R \) in this way. The shortening process is realized by wiping a part off each contact time in (1).

The length of contact time shortened by is also represented as \( t_i ^* \) which has the same meaning as in the linear program of (2). It is important to notice that the end time of the wiped part must be earlier than \( t_i \), otherwise such shortening makes no contribution to overflow avoiding. Therefore the wiped part of ingoing link's contact time can be presented as \( \min(t_i ^*, t_{in}^i - t_i) \). That is if \( t_i ^* \geq t_{in}^i - t_i \), change the contact time of link \( S_i R \) in (1) into \([t_{in}^i - t_i, t_{in}^i - t_i - t_i v_{in}^j / v_{out}^j] \), otherwise change it into two parts, \([t_{in}^i, t_{in}^i - t_i] \) and \([t_{in}^i - t_i, t_{in}^i - t_i - t_i v_{in}^j / v_{out}^j] \). Accordingly, the contact time of outgoing link \( R D_j \) in (1) is changed into \([t_{out}^j - t_i v_{in}^j / v_{out}^j, t_{out}^j - t_i v_{in}^j / v_{out}^j] \). Therefore the linear program of (2) can be simplified as follow:
\[ \text{max } Th = \frac{1}{T} \sum_{i=0}^{n} \int_{0}^{T} \nu_{\text{in}}^i(t)dt \]
\[ = \frac{1}{T} \left[ \nu_{\text{in}}^1(t_1^{\text{in}} - t_1^{\text{in}} - t_1) + \nu_{\text{in}}^2(t_2^{\text{in}} - t_2^{\text{in}} - t_2) \right. \]
\[ \left. + \cdots + \nu_{\text{in}}^n(t_n^{\text{in}} - t_n^{\text{in}} - t_n) \right] \]
\[ \text{s.t. } \sum_{i=1}^{n} \left[ \nu_{\text{in}}^i(t) \right] dt = C_{\text{over}} \]
\[ 0 \leq t_i \leq \min(t_{\text{out}}^i - t_{\text{in}}^i) \]

3. Model Analysis

In this section, we analyze the factor which can influence the performance of TSA.

To investigate what factor the increase of throughput is dependent on, we define following variables. Let \( F_i \) denote the total data flow of \( p_i \) in the period of \( T \) under the assumption of unlimited storage of \( R \), i.e. without consideration of \( R \)'s storage when computing \( F_i \). So we use original \( L_{\text{in}}^i(t) \) of (1) to calculate \( F_i \). Expression of \( F_i \) is as follow:

\[ F_i = \int_{0}^{T} \nu_{\text{in}}^i(t)dt \]

Let \( C_{\text{max}} \) denote maximal occupation of \( p_i \) on \( R \)'s storage under the same assumption, which is represented as follow:

\[ C_{\text{max}}^i = \text{max } f_j(t), 0 \leq t \leq T \]

We define the ratio of data flow \( C_{\text{max}}^i \) to maximal occupation \( C_{\text{max}}^i \) as \( \rho_i \). A bigger \( \rho_i \) means attaining the same data flow of \( p_i \) at a lower cost of occupation on \( R \), or gaining a better data flow at the same cost of occupation. In the same period of \( T \), better data flow means better throughput. \( \rho_i \) also can show the relationship of contact times between an ingoing link and its corresponding outgoing link. A bigger \( \rho_i \) means longer overlap between their contact times along the time axis. It is important to notice the minimal value of \( \rho_i \) is 1, which denotes no overlap between contact times.

Our solution in this paper is actually restricting transmission on the path with a smaller \( \rho_i \) to make room for the transmission on the path with a bigger \( \rho_i \). In this way, the network can get a better data flow by utilizing the storage of relay node properly, avoiding some messages staying in the relay node for a long while. If \( \rho_i = 1 \) for \( i = 1, 2, \cdots n \), our transmission scheduling algorithm will be useless since there is no possibility to increase throughput without overlaps.

4. Simulation Results

We use MATLAB to do the simulation of a network with five nodes as the example shown in Section I, i.e. there are two paths in all. Contact times and link rates are generated randomly in our simulation, satisfying the following two conditions: i) guarantee at least one path’s \( \rho \) equals 1; ii) the outgoing link can undertake the transmission task of its corresponding ingoing link.

Contact times are generated a hundred times. For lack of space, we have chosen ten times of them to show their average throughput in Figure 4. We set relay node storage \( C_{\text{max}}^R \) as 10. Except in the 1st, 4th and 10th times, throughput has been increased with the use of TSA. The same throughput in the 1st time is due to the fact that both \( \rho_1 \) and \( \rho_2 \) equal 1, and so is it in the 10th time. The reason for the same throughput in the 4th time is that there is no overflow before scheduling, i.e. the relay node storage is enough for the whole transmission. In the 2nd, 3rd, 6th, 7th, 8th and 9th times, the throughput is 10 before scheduling, that is because there is only message transmission on the ingoing link which belong to a path with \( \rho = 1 \) before the storage is fully occupied.

Figure 5 illustrates how the average network throughput varies with \( \rho_2 \), the ratio of data flow to occupation of path2. In this simulation, \( \rho_1 \) is kept as 1. It is observed from Figure 5 that the network throughput increases with \( \rho_2 \) after scheduling. A bigger \( \rho_2 \) means longer overlap between contact times, but the potential of overlap can not be exploited without transmission scheduling algorithm. That's why the network throughput is constant before scheduling.

Figure 6 illustrates how the average throughput on each path varies with \( C_{\text{max}}^R \), the maximal storage of relay node. Before scheduling, the throughput of path1 gets to its maximum first while the throughput maximum of path2 comes earlier after scheduling. Before scheduling, the throughput of path2 has been restricted to 0 when \( C_{\text{max}}^R \) is relatively small, while the throughput of path1 increases with \( C_{\text{max}}^R \) until path1 completes its whole transmission. The reason is that the start of \( S_i^R \)'s contact time is relatively late, leading to the storage of \( R \) fully occupied by transmission on path1. When \( C_{\text{max}}^R \) is big enough to undertake the transmission on all the paths,
This algorithm takes intermittent connection and constrained storage of DTN into consideration. In the future, we want to analyze how high delay will affect this transmission scheduling algorithm, and optimize the algorithm in order to apply it in more complex DTN scenarios.

6. Acknowledgements

This work is supported by the National Science Foundation (60972047, 61231008), National S&T Major Project (2011ZX03005-004, 2011ZX03004-003, 2011ZX-03005-003-03, 2013ZX03004007-003), Shaanxi 13115 Project(2010ZDKG-26), National Basic Research Program of China (2009CB320404), Program for Changjiang Scholars and Innovative Research Team in University(IRT0852), the 111 Project (B08038) and State Key Laboratory Foundation (ISN1002005, ISN090305).

REFERENCES