Congestion Management in Delay Tolerant Networks *

Guohua Zhang, Jing Wang, Yonghe Liu
Department of Computer Science and Engineering
The University of Texas at Arlington
{gzhang,yonghe}@uta.edu

ABSTRACT

In delay tolerant networks, custody transfer can provide certain degree of reliability as a custodian node cannot discard a message unless its life time expires or the custody is transferred to another node after a commitment. This creates a challenging decision making problem at a node in determining whether to accept a custody transfer: on one hand, it is beneficial to accept a large number of messages as it can potentially advance the messages toward their ultimate destinations and network utilization can be maximized; on the other hand, if the receiving node over-commits itself by accepting too many messages, it may find itself setting aside an excessive amount of storage and thereby preventing itself from receiving further potentially important, high yield (in terms of network utilization) messages. To solve this problem, in this paper, we *apply the concept of revenue management, and employ dynamic programming to develop a congestion management strategy for delay tolerant networks.* For a class of network utility functions, we show that the optimal solution is completely distributed in nature where only the local information such as available storage space of a node is required. This is particularly important given the nature of delay tolerant networks where global information is often not available and the network is inherently dynamic. Our simulation results show that the proposed congestion management scheme is effective in avoiding congestion and balancing network load among the nodes.

Categories and Subject Descriptors

C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design - *Store and forward networks*

General Terms

Algorithm, Measurement, Performance

*This work is partially supported by NSF grants CNS-0721051 and CNS-0834493.

1. INTRODUCTION

Different from conventional networks exemplified by the Internet, delay tolerant networks often face long round trip delay, intermittent connectivity, and opportunistic contacts among nodes, which can be traced back to its origin of deep space communication [7, 5]. Correspondingly, store-and-forward, message-oriented architectures, in contrast with the dominating end-to-end architecture of current Internet, are often adopted to cope with the challenged environments [8]. In particular, to enhance end-to-end reliability, *custody transfer*, is proposed where the responsibility for reliable delivery of a message (*often termed a bundle*) is gradually moved toward its ultimate destination in hop-by-hop fashion [8].

Unfortunately, by accepting the custody of a bundle (in other words, the responsibility of reliable delivery of a bundle), a node may have to store the bundle for a significant period of time before being able to hand it over to another node during opportunistic contact, as discarding a bundle before its life time expires is generally prohibited. As a result, the precious storage space is committed by accepting the custody of a bundle which may likely hinder acceptance of future custody transfer requests. Therefore, it may not be entirely wise for a node to openly and widely declare oneself to be a willing storage device for any set of bundles requested. Rather, a carefully crafted congestion management strategy is desired in order to effectively manage the storage capacity on a node in delay tolerant networks so that the network utilization can be maximized. It is this problem that this paper aims to address.

While a node cannot discard the bundle unless its life time expires or the custody is transferred to another node after a commitment, the node indeed has the freedom in deciding whether to accept the custody in the first place. Consequently, two conflicting forces can be considered that are governing the receiving node's actions: on one hand, it is beneficial to accept a large number of messages as it can potentially advance the messages toward their ultimate destinations and network utilization can be maximized; on the other hand, if the receiving node over-commits itself by accepting too many messages, it may find itself setting aside an excessive amount of storage and thereby preventing itself from receiving further potentially important, high yield (in terms of network utilization) messages.

In this paper, we *apply the concept of revenue management, and employ dynamic programming to develop a congestion management strategy for delay tolerant networks.* For a class of network utility functions, we show that the

optimal solution is completely distributed in nature where only the local information such as available storage space of a node is required. This is particularly important given the nature of delay tolerant networks where global information is often not available and the network is inherently dynamic. Extensive simulation results show that the proposed congestion management scheme is effective in avoiding congestion and balancing network load among the nodes.

Although bundle handling protocols and routing schemes specific to delay tolerant networks have been extensively proposed [7, 19, 22, 20, 13, 9], few results, as to the authors' knowledge, exist on how to handle the above discussed congestion/resource allocation problem. Given the plethora of potential applications of delay tolerant networks in various domains, congestion will become imminent when the network technology is successfully applied and proper management schemes are demanded.

The paper is organized as follows. In Section 2, we investigate the related work for delay tolerant networks. Section 3 briefly presents some concepts of dynamic programming and delay tolerant networks. The congestion management problem is formulated in Section 4. In Section 5, we establish the dynamic programming formulation for the defined congestion management problem in Section 4. Section 6 studies the optimal strategy for congestion management in delay tolerant network with multiple nodes. In Section 7, we use several simulation scenarios to show the performance of the derived optimal control policies. Section 8 concludes the paper.

2. RELATED WORK

Delay tolerant networks in general have been the subject of a wide range of research [7, 8, 11, 15, 19, 22, 12, 9, 20, 13, 17]. Among those, a set of papers have focused on the routing problems [12, 20, 13, 17]. In these works, usually buffer space in each node is assumed to be unlimited or treated in an ad-hoc manner, as resource allocation is not the key focus there. For example, in [14], it is assumed that the send buffer and receive buffer have limited space in designing message ferry route, but the congestion issue is not addressed.

While the bundle layer employs reliable transport layer protocols together with custody transfers to offer hop-by-hop reliability, no end-to-end reliability can be guaranteed. In [11], the authors have developed active receipt, passive receipt, and network-bridged receipt approaches to offer end-to-end reliability by delivering an active end-to-end acknowledgment over the delay tolerant network itself or another network.

It is worth noting that the storage congestion mitigation problem is handled in [16]. When custodian node becomes congested, the authors propose to migrate its stored messages to alternative storage (usually neighboring nodes) locations to avoid losses. The problem of storage congestion in a node is approached by migrating the stored data to neighbors. The proposed solution includes a set of algorithms to determine which messages should be migrated to which neighbors and when. However, this approach is a passive approach. In contrast, our approach is proactive in that each node actively makes decision on whether to accept a custody request in order to avoid congestion. In [5], a financial model based approach is adopted in addressing the congestion problem in delay tolerant networks. The concept introduced there such as conveyance fee paid by the sender and receiver of a bundle is different from the concepts of benefits of requests and opportunity cost proposed in our work. Furthermore, our work achieves distributed optimal solution when multiple nodes are present.

On the other hand, resource allocation has been the subject of extensive study in related overlay networks. In [2], resource allocation in overlay networks to protect the network from being overloaded is investigated. There, cost function and benefit function are defined. And subsequently, the decision whether to accept packets into the network is made by comparing the benefit of accepting a packet to cost on relevant paths. However, the approach is a centralized one since an oracle is needed to compute the cost of all links in a path. While the concepts of benefit and opportunity cost in our work are similar to the ones in [2], our approach is based on local information of each node which makes it applicable to the dynamic environment of delay tolerant networks.

There also exist a set of papers studying that network capacity can be significantly improved by exploring node mobility in wireless networks [10, 6, 21]. For example, autonomous agents is introduced in [6] as additional participants in delay tolerant networks. The agents can adapt their movements in response to variations in network capacity and demand to improve network performance. Delay-throughput tradeoff is investigated in [21] in mobile ad-hoc networks under hybrid random walk and one dimensional mobility models. In this paper, our congestion control strategy is a general approach independent of mobility model. Instead, it only depends on the remaining storage space in receiving nodes and the request reward.

3. PRELIMINARIES

Delay tolerant networks have attracted tremendous interests from the academia, military and industry recently. Interested applications include interplanetary networks, mobile tactical military networks, communication networks for remote rural areas, which often consist of wireless communications and user mobility [7, 5]. In these environments, often building a standard network with end-to-end connectivity is impractical, or transmission latencies are inherently high.

In delay tolerant networks, a new protocol layer, termed the *Bundle Layer*, is overlayed at the application layer or at least above the transport layer [7]. The bundle layer stores and forwards entire bundles (or bundle fragments) between nodes. Specifically, a node holding a bundle with custody is called custodians. Bundles are also called messages, which can consist of multiple pieces of applications data. Usually, a single bundle layer is employed across potentially heterogenous network domains in order to form a delay tolerant network.

The bundle layer can use reliable transport layer protocols together with custody transfers to move points of retransmission progressively forward toward the destination. This property minimizes the number of potential retransmission hops, and consequently reduces additional network load caused by retransmission, and the total time to convey a bundle reliably to its destination.

4. PROBLEM FORMULATION

While nodes in delay tolerant networks may offer custody

transfer, this decision is at their own discretion. Furthermore, a node that has been accepting messages and corresponding custodies can decide, on its own, to cease the accepting option if its local resources become substantially consumed. Similarly, the accepting operation can be resumed at the node when resources become more plentiful after, for example, when certain messages are successfully transferred to its downstream nodes. It is this decision as to whether to accept the custody of a message given the current resource constraint we are addressing in this paper. In doing so, our goal is to maximize the overall system benefit subject to the constraints of the system resources.

Before proceeding further on the formal model, we remark that routing algorithms can affect the performance of delay tolerant networks significantly. Unfortunately, complete knowledge of the delay tolerant network and routes can often not be known in advance nor are they static. In order to focus on the buffer management problem, we follow [16] and separate the management mechanism from route selection problem. Their joint design is our ongoing work and beyond the scope of this paper.

4.1 System Model

Without loss of generality, assume that node $i+1$ is any node to whom node i can forward messages by custody transfer. Similarly, node $i-1$ can be considered as any node that wants to forward messages to its downstream custodian i during contact opportunity. Node $i-1$ sends a request for a new custodian to fulfill the bundle transfer. It then waits for acknowledgement from its neighbor node. If node i receives a request for custodian from its neighbor node $i-1$, it will decide whether to accept or reject the request based on the current available storage space, and predefined optimal control strategy. We here assume that the storage space is the key resource constraint. Notice that we have not made any assumption regarding the contacts among nodes and hence can accommodate different mobilities.

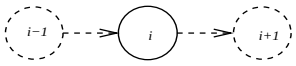


Figure 1: Simple DTN Scenario

By accepting a bundle, a node accumulate a certain amount of benefit denoted by B_j, where $B_j > 0$, $j \in \{1, \ldots, m\}$ is the index of the class of benefit. The benefit function can be of various forms and correspondingly different optimization goals can be achieved. For example, by properly adjusting the benefit function, performance issues such as throughput and fairness can be addressed [2].

Notably, the benefit can be a function of the bundle size with different weights based on their corresponding traffic priorities or traffic types. For simplicity, we assume that the bundle sizes of different priorities (or types) are homogeneous in volume and thus indistinguishable when filling the buffer if they are accepted. We also assume that arrivals of requests for custody occur at discrete points in time, which are called decision epochs. Notice that the arrival requests (events) drive the decision epochs, in other words, the decision epochs are not determined but rather by the arrival requests themselves. We also assume that the departure of a message can occur at anytime between decision epochs.

Over finitely many decision epochs, i.e., finite time horizon, our objective is to determine the optimal congestion management strategies that maximize the expected total benefits by accepting/forwarding the bundles, subject to the request and buffer constraints. Formally, the objective is to choose congestion management strategies that maximize the total expected reward over a time horizon of T decision epochs.

$$\mathbb{E}\left[\sum_{t=1}^{T} r_t u_t\right], \tag{1}$$

where $r_t \in \{B_1, B_2, \ldots, B_m\}$; $u_t = 1$ if the transfer request is accepted at decision epoch t, $u_t = 0$ otherwise; $\mathbb{E}(\cdot)$ is mathematical expectation.

We here remark that in this model, for each hop of custody transfer, the benefit is accumulated. And these benefits are summarized across the whole network over a finite time horizon. Additionally, unlike many economic models, our model does not try to reach an equilibrium state based on the rationality of participant nodes or influence noncooperative behavior. Rather, the goal is to optimize the overall revenue by accepting/forwarding bundle transfer requests under the assumption of minimally cooperative (nonrational) behaviors of nodes [2].

4.2 State Variable and Action Variables

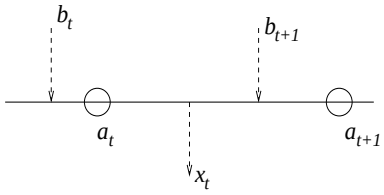


Figure 2: State transition relation

We define the state variable a_t to be the remaining capacity at decision epoch t. Since we assume that a request arrives at a node and drives the decision epoch t, message departure can occur between decision epoch t and $t+1$ if opportunistic contact exists between the current custodian and its future custodian. The remaining capacity in the current node at decision epoch $t+1$ may include the space released by bundles transferred to the next custodian during opportunistic contacts. Let the request size be x_t at decision epoch t, and the released space available for next decision epoch $t+1$ be b_{t+1}. The transfer relation shown in Fig. 2 can be described by the following transfer function.

$$a_{t+1} = a_t - u_t x_t + b_{t+1} \tag{2}$$

where u_t has been defined in (1); $b_{t+1} = 1$ if the bundle is successfully transferred to the next custodian, $b_{t+1} = 0$ otherwise; $x_t = 1$.

As we have said before, storage space may be released in time duration between decision epoch t and decision epoch $t+1$ due to message departure. Since the released space is only beneficial to arrivals at decision epoch $t+1$ and later, b_{t+1} is indexed with $t+1$.

4.3 Opportunity Cost and Benefit Function

Our goal is to choose the optimal strategies for maximizing the expected sum of benefits. Toward this, we need to consider two conflicting forces. First, it is wasteful to commit resources to requests that are not "desperate" for that resource, i.e., not enjoying the maximal possible benefit from occupying the resource. Second, it is equally dangerous to gamble that each resource can be occupied with maximal benefit gained without knowing the sequence of requests coming in the future. The key aspect of the above situations is that each decision can not be viewed in isolation since one must balance the desire for high benefit request with the undesirability of low future benefit request.

We use *opportunity cost* and *benefit function* to balance the above two conflicting forces. The opportunity cost measures the value of the storage capacity, which is the benefit that may be lost by higher benefit request as a result of consumption of the above resource by the lower benefit request. Theoretically, the opportunity cost can be captured by defining a *value function* $V_t(\cdot)$, which measures the optimal expected benefit as a function of the remaining capacity a_t at decision epoch t [18]. The opportunity cost is then the difference between the value function at a_t and the value function at $a_t - 1$, that is, $V_t(a_t) - V_t(a_t - 1)$. Obviously, the theoretical analysis of the optimal control strategies will heavily rely on the value function $V_t(\cdot)$.

On the contrary, the benefit function, as we have discussed before, denotes the gain of moving a bundle to the next hop. It can be defined according to users' specifications in the systems (for example as a function of the bundle size and type). While there is no strict limitation to choose benefit function, the choice of benefit function should guarantee that the value of benefit function and opportunity cost are comparable.

5. SINGLE NODE CONGESTION CONTROL

In this section, we will focus on the decision for custody acceptance/rejection for a single node. The case for multiple nodes will be studied in the next section. Toward this end, we present a dynamic programming approach to the problem formulated in the previous section.

Dynamic programming can handle situations where decisions are made at decision epochs. The outcome of each decision may not be fully predictable but can be anticipated to certain extent before the next decision is made. The objective usually is to minimize a certain cost or maximize a certain reward. At each decision epoch, dynamic programming technique ranks decisions based on the sum of the present cost/reward and the expected future cost/reward, assuming optimal decision making for subsequent decision epochs [4]. Our technique will follow this storyline as well.

Recall that the value function $V_t(\cdot)$ denotes the value of the remaining capacity at decision epoch t, that is, the value of remaining capacity at decision epoch t is $V_t(a_t)$. We assume that the probability of an arrival of class j at decision epoch t is denoted by $p_j(t)$, and *at most one request arrives in one decision epoch*. Subsequently, we easily have $\sum_{j=1}^{m} p_j(t) \leq 1$.

Evidently, the arrival probability may vary with decision epoch t, and hence the mix of classes that arrive may vary over time. This varying probability will not affect the optimal control strategies, but it will incur extra computation.

For now, let us assume that a decision epoch will be driven by a request of one message, that is, at most one message arrives in one decision epoch (the case for a request with multiple message custody transfers will be discussed in 5.2).

Let r_t be a random variable, with $r_t = B_j$ if a request for class j arrives at decision epoch t, and $r_t = 0$ otherwise. Note that the probability $\mathbb{P}(r_t = B_j) = p_j(t)$. Our goal is to maximize the sum of the current reward and the reward to go, i.e.,

$$\max_{u_t \in \{0,1\}} \Big\{ r_t u_t + V_{t+1}(a_t - u_t x_t) \Big\} \tag{3}$$

Since $x_t = 1$, we can rewrite (3) as

$$\max_{u_t \in \{0,1\}} \Big\{ r_t u_t + V_{t+1}(a_t - u_t) \Big\} \tag{4}$$

Intuitively, the above objective function targets at achieving a balance between the current reward ($r_t = B_j$) for accepting a request and the potential value (V_{t+1}) of the remaining storage space in a node (possibly used for later bundle acceptance). For optimal value function $V_t(a_t)$ at decision epoch t, we have the following expression.

$$V_t(a_t) = \mathbb{E}\Big[\max_{u_t \in \{0,1\}} \{ r_t(t) u_t + V_{t+1}(a_t - u_t) \} \Big] \tag{5}$$

Given the finite horizon being considered, the boundary conditions are

$$V_t(0) = 0, \;\; t = 1, \ldots, T.$$

and

$$V_T(a_T) = r_T.$$

Here r_T stands for a salvage reward for the remaining amount of resource at the end of the time horizon. If $a_T = 0$ then $r_T = 0$; if $a_T \neq 0$, we assume that r_T is a concave function of the remaining capacity a_T such as piece-wise linear function. This assumption will guarantee that the optimal value function $V_t(\cdot)$ is also concave.

5.1 Optimal Strategy for Accepting Custody Transfer

In this subsection, our goal is to prove the following theorem regarding the optimal policy for the congestion control.

THEOREM 1. *For a class j request with reward $r_t = B_j$ arriving at decision epoch t, it is optimal to accept the request if and only if*

$$r_t \geq \Delta V_{t+1}(a_t) \tag{6}$$

This theorem actually denotes that for a request, if the benefit (r_t) is greater than its opportunity cost ($\Delta V_{t+1}(a_t)$), the message shall be accepted and the decision actually is optimal.

To do this, we first prove Lemma 1-3 that will be employed to deduce expression suitable for proving Theorem 1.

LEMMA 1. *If x, y are integers, and $x \geq y$, we have*

$$V_t(x) = V_t(x - y) + \sum_{k=x-y+1}^{x} \Delta V_t(k)$$

where $\Delta V_t(k) = V_t(k) - V_t(k-1)$.

PROOF.

$$\begin{aligned} V_t(x) &= V_t(x) - V_t(x-1) + V_t(x-1) \\ &= V_t(x) - V_t(x-1) + V_t(x-1) - V_t(x-2) \\ &\quad + \cdots - V_t(x-y) + V_t(x-y) \\ &= V_t(x-y) + \sum_{k=x-y+1}^{x} \Delta V_t(k) \end{aligned} \tag{7}$$

□

LEMMA 2.

$$\max_{u\in\{0,1\}}\{r_t u + V_{t+1}(x-u)\} = V_{t+1}(x) + \max_{u\in\{0,1\}}\{(r_t - \Delta V_{t+1}(x))u\}$$

where $\Delta V_{t+1}(x) = V_{t+1}(x) - V_{t+1}(x-1)$ *is the expected marginal value of remaining capacity in decision epoch* $t+1$.

PROOF. From Lemma 1, it is apparent that

$$V_{t+1}(x) = V_{t+1}(x-y) + \sum_{k=x-y+1}^{x} \Delta V_{t+1}(k) \tag{8}$$

In equation (8), note that if $y = 0$, the last summation disappears. Let $y = u$, then we have

$$V_{t+1}(x-u) = V_{t+1}(x) - \sum_{k=x-u+1}^{x} \Delta V_{t+1}(k) \tag{9}$$

As u can be 0 or 1, (9) can be changed to

$$V_{t+1}(x-u) = V_{t+1}(x) - \Delta V_{t+1}(x)u \tag{10}$$

By (10), the expression

$$\max_{u\in\{0,1\}}\{r_t u + V_{t+1}(x-u)\} \tag{11}$$

can be rewritten as

$$\begin{aligned} &\max_{u\in\{0,1\}}\{r_t u + V_{t+1}(x-u)\} \\ &= \max_{u\in\{0,1\}}\{r_t u + V_{t+1}(x) - \Delta V_{t+1}(x)u\} \\ &= V_{t+1}(x) + \max_{u\in\{0,1\}}\{(r_t - \Delta V_{t+1}(x))u\} \end{aligned}$$

□

By Lemma 2, the Bellman equation (5) can be rewritten as

$$\begin{aligned} V_t(a_t) &= \mathbb{E}\left[\max_{u_t\in\{0,1\}}\{r_t(t)u_t + V_{t+1}(a_t - u_t)\}\right] \\ &= V_{t+1}(a_t) + \mathbb{E}\left[\max_{u_t\in\{0,1\}}\{(r_t - \Delta V_{t+1}(a_t))u_t\}\right] \end{aligned} \tag{12}$$

And as a result, we can now prove Theorem 1. We will use mathematical induction to prove Theorem 1. In order to prove the above theorem, it is necessary to investigate the concavity of the value function $V_t(\cdot)$ and $\Delta V_t(\cdot)$. To do so, we need the Definition 1 and Lemma 3.

DEFINITION 1. *A function defined on the set of nonnegative integers* $g : \mathcal{Z}_+ \to \mathcal{R}$ *is concave if it has nonincreasing differences, that is,* $g(x+1) - g(x)$ *is nonincreasing in* $x \geq 0$.

The above definition can be considered as a discrete version of concave definition of a continuous function defined on $\mathcal{R}$. Additionally, we also present the following lemma from [18].

LEMMA 3. *Suppose* $g : \mathcal{Z}_+ \to \mathcal{R}$ *is concave. Let* $f : \mathcal{Z}_+ \to \mathcal{R}$ *be defined by*

$$f(x) = \max_{a=0,1,\ldots,m}\{ap + g(x-a)\}$$

for any given $p \geq 0$ *and nonnegative integer* $m \leq x$. *Then* $f(x)$ *is concave in* $x \geq 0$ *as well.*

Interested readers are referred to [18] for details on the proof of Lemma 3. Using Lemma 3, we will prove Theorem 1 below.

PROOF OF THEOREM 1. We first prove that the function $V_t(x)$ is concave in x. The proof is by induction at decision epochs. Note that in the terminal decision epoch T, there are two possibilities: the first one is that $a_T = 0$, then $V_T = 0$; the second one is that $a_T \neq 0$ and then $r_T \neq 0$. For the second case, we assume that the remaining capacity receives a salvage reward that is concave in a_T. For the above two cases, $V_T(a_T)$ is concave in a_T.

Assume $V_{t+1}(x)$ is concave in x at decision epoch $t+1$. By Lemma 3, we can easily know that

$$\mathbb{E}\left[\max_{u_t\in\{0,1\}}\{r_t(t)u_t + V_{t+1}(a_t - u_t)\}\right]$$

is concave. The reason is that it is simply the expectation of $\max_{u_t\in\{0,1\}}\{r_t(t)u_t + V_{t+1}(a_t - u_t)\}$ based on the probability $\mathbb{P}(r_t = B_j) = p_j(t)$. Therefore, we conclude that $V_t(x)$ is concave.

We can also prove that $\Delta V_t(x) \geq \Delta V_{t+1}(x)$ by way of induction. The intuition of this inequality is that the marginal value at any given remaining capacity x decreases with time.

From the above argument, we can know that the optimal value $V_t(a_t)$ at decision epoch t can be achieved if and only if $r_t \geq \Delta V_{t+1}(a_t)$. □

5.2 Discussion

In this subsection, we first analyze the properties of setting a fixed opportunity cost, then discuss the case of a request with multiple messgaes custody transfers.

Note that $\Delta V_{t+1}(a_t)$ is a function of a_t, that is, the opportunity cost dynamically varies with remaining capacity of a_t. The opportunity cost increases as the remaining capacity decreases. While setting a fixed opportunity cost to accept or reject a request is simple, to be effective, the opportunity cost must be updated after accepting a request or forwarding a message. Without the ability to make opportunity cost a function of available storage space, however, a simple static opportunity cost is indeed a dangerous form of control [18]. If the static opportunity cost is too low, then there is no difference between the request with low reward and the request with high reward. If the static opportunity cost is too high, then most of the requests will be rejected even if the utilization of resources in nodes is very low.

In the above discussion, for the sake of simplicity, we have also assumed that only a request of one message arrives at a decision epoch. Indeed, the approach can be easily generalized to the scenario where a request with multiple messages can arrive at a decision epoch. If a request contains multiple messages, the receiving node may decide to partially accept any quantity q in the range $0 \leq q \leq Q$ given the number of messages $Q \geq 1$. In this case the analysis in 5.1 still holds by serializing the messages and applying the above steps. In other word, a request drives a decision epoch, accepting node then serializes all messages contained in a request

with certain rules. The optimal control strategies can decide whether each message can be accepted or not one by one based on the dynamic opportunity costs.

However, a request may contain custodian transfer for multiple messages and the request must be satisfied in an *all-or-none basis*. In other words, given a request containing $Q > 1$ messages, all Q messages or none must be accepted. In this case, the value function may not be concave. The marginal value of capacity may actually increase [18], and the congestion management strategies developed in 5.1 may not be optimal. To address this issue, we first must specify the distribution of group sizes to model how much demand we have from groups of various sizes. This in fact does not increase the theoretical difficulty. The difficulty lies in that the value function may not be concave which can make the optimality issue intractable [18, 14]. We leave this as the subject of our future study.

We also remark that the benefit function itself does not address the issue about how the request is forwarded. On the contrary, the route itself shall be decided by the routing algorithms. In other words, we rely on the routing algorithm to prevent the loops or tossing back of messages for artificial benefit inflation.

6. NETWORK CONGESTION CONTROL

In the above section, we have studied the optimal congestion control policy for a single node case. In this section, we first extend the above dynamic programming based approach to network capacity control with multiple resources, then develop the congestion control policy for delay tolerant networks with multiple nodes.

6.1 Optimal Policy with Global Information

Suppose that the network has n nodes and there are m requests. Each request may need a combination of resources on the n nodes. Define an *incident matrix* $\mathbf{A} = [a_{lh}]_{n\times m}$, where $a_{lh} = 1$ if resource on node l is used by request h and $a_{lh} = 0$ otherwise. As a result, the h^{th} column of $\mathbf{A}$, denoted by $\mathbf{A}_h$, is the incidence vector for request h; the l^{th} row, denoted by $\mathbf{A}^l$, has an entry of one in column h corresponding to a request h that uses resource on node l. If the network topology is fixed and routing path is predefined, we can easily construct incidence matrix $\mathbf{A}$ with appropriate dimensions based on schemes that the requests use the resources.

Demand in period t can be modeled as the realization of a single random vector $\mathbf{B}(t) = (B_1(t), \ldots, B_m(t))$. If $B_h(t) = B_h > 0$, a request h arrives and the benefit to accept it is B_h; if $B_h(t) = 0$, then there is no request with type of h. A realization $\mathbf{B}(t) = 0$ means that there is no request at time period t.

The state of the network can be described by $\mathbf{x}(t) = (x_1(t), \ldots, x_n(t))$, where $x_l(t)$ is the remaining buffer capacity of resource on node l at time period t. Let $u_h(t)$ be decision variable for request at time period t. $u_h(t) = 1$ if request h is accepted in time period t, $u_h(t) = 0$ otherwise. The decision to accept, $u_h(t)$, is a function of the remaining capacity vector $\mathbf{x}(t)$ and benefit B_h of request h, that is, $u_h(t) = u_h(t, \mathbf{x}, B_h)$. Since we can accept at most one request in any period, if the current remaining capacity is $\mathbf{x}(t)$, then the following condition should be satisfied: $\mathbf{x}(t) \geq \mathbf{A}_h$.

Similar to Section 5, let $V_t(\mathbf{x})$ denote the optimal expected revenue to go. Then $V_t(\mathbf{x})$ satisfies the following Bellman equation.

$$V_t(\mathbf{x}) = \mathbb{E}\left[\max_{u_h \in \{0,1\}} \left\{ B_h(t)u_h + V_{t+1}(\mathbf{x} - \mathbf{A}_h u_h) \right\}\right] \quad (13)$$

Appropriate boundary conditions for the above Bellman equation can be set at decision epoch T provided that the salvage benefit at decision epoch T is concave. Subsequently, as in 5.1, we can obtain the optimal control for the above equation as

$$u_h(t, \mathbf{x}, B_h) = \begin{cases} 1 & \text{if } B_h \geq V_{t+1}(\mathbf{x}) - V_{t+1}(\mathbf{x} - \mathbf{A}_h) \\ & \text{and} \quad \mathbf{A}_h \leq \mathbf{x}(t) \\ 0 & \text{otherwise.} \end{cases} \quad (14)$$

The idea behind the above optimal control policy is that accepting a request h with benefit B_h if and only if there are sufficient remaining capacities in relevant resources and benefit is greater or equal to the opportunity cost to occupy the storage spaces.

The structure of the value function for network case (13) is similar to the value function for the single node (5). However, this is partially resulted from the assumption that global information is available and centralized calculation can be performed. This framework used in Revenue Management is most likely impractical for delay tolerant networks where the network is potentially highly dynamic in almost every aspect.

6.2 Distributed Optimal Policy

The above analysis follows the classic revenue management with an unrealistic assumption on the availability of global information. Furthermore, even if the global information were available, the potential large dimension of the state space would often render the solution hopeless in practice. Fortunately, the unique setup of delay tolerant networks naturally provides an approach toward distributed solution based on system decomposition.

The key for distributed solution is the independence of the resource requirement at each node in the network. In conventional resource management strategy, the resources are requested simultaneously at multiple nodes in order to pave the end-to-end path. This is actually reflected in the above analysis when global information is available. In such a scenario, if one of the node could not fulfill the request, the request actually will be rejected. Fortunately for delay tolerant networks, end-to-end path is often not present and hop-by-hop forwarding and control is employed. Once the bundle is transferred to another hop, the resource originally occupied will become immediately available. Whether this transaction will be executed will solely depend on the receiving node.

Based on above discussion, a request in delay tolerant networks can be expressed as $A_j = (0, \ldots, 0, 1, 0, \ldots, 0)$, where A_j is a request vector that custody transfer is requested on node j. Without losing generality, we assume that the value function $V_{t+1}(\mathbf{x})$ has a gradient $\nabla V_{t+1}(\mathbf{x})$. In other words, the value function $V_{t+1}(\mathbf{x})$ is differential with respect to vector $\mathbf{x}$. Formally,

$$\begin{aligned} & V_{t+1}(\mathbf{x}) - V_{t+1}(\mathbf{x} - A_j) \\ & \approx \nabla V_{t+1}^T(\mathbf{x}) A_j \\ & = \sum_{i \in A_j} \pi_i(t, \mathbf{x}) = \pi_j(t, \mathbf{x}) \end{aligned} \quad (15)$$

where $\pi_j(t, \mathbf{x}) = \frac{\partial}{\partial x_j} V_{t+1}(\mathbf{x})$ is the *opportunity cost* of node j at decision epoch $t+1$.

From the above equation, we can see that the opportunity cost evidently depends on the format of the utility function. Assume that $V_{t+1}(\mathbf{x})$ has the following format

$$V_{t+1}(\mathbf{x}) = \sum_{i=1}^{n} V_{t+1}^{i}(x_i) \tag{16}$$

where $V_{t+1}^{i}(x_i)$ is the value function of node i at decision epoch $t+1$. Then, from (15) and (16), we have

$$\begin{aligned} &\pi_j(t, \mathbf{x}) \\ &= \frac{\partial}{\partial x_j} V_{t+1}(\mathbf{x}) = \frac{\partial}{\partial x_j} \sum_{i=1}^{n} V_{t+1}^{i}(x_i) \\ &\approx V_{t+1}^{j}(x_j) - V_{t+1}^{j}(x_j - 1) \end{aligned} \tag{17}$$

From (17), the opportunity cost of node j at decision epoch $t+1$ only depends its remaining storage space.

In the above discussion, we assume that the value function $V_{t+1}(\mathbf{x})$ is differential with respect to vector $\mathbf{x}$. If the value function $V_{t+1}(\mathbf{x})$ is not differential with respect to vector $\mathbf{x}$, gradient $\nabla V_{t+1}(\mathbf{x})$ can be replaced by subgradient, it will not affect the outcome except introducing extra computation [3].

From Theorem 1 and (17), the congestion control policy in node j is as follows.

$$r_t \geq \Delta V_{t+1}^{j}(x_j) \tag{18}$$

where $\Delta V_{t+1}^{j}(x_j) = V_{t+1}^{j}(x_j) - V_{t+1}^{j}(x_j - 1)$. This policy can achieve network level optimality given the value function presented in (16).

One reason to choose the value function such as (16) is that we consider delay tolerant networks where nodes maximize a common additive value. Each node has information only about its value component, and maximizes that component while exchanging information between any two nodes only during their opportunistic contact.

If we choose general value function other than the format of (16), the key point is still on how to compute the opportunity cost. From (15), we can know that the opportunity cost of node j at decision epoch $t+1$ depends on the remaining storage spaces in other nodes. Since it is not practical for each node to have information about other nodes in delay tolerant networks, this is another reason that we empoly the value function as in (16).

For general value function, we can use some decomposition approach to decompose the general value function into the format as in (16). The decomposition is at the expense of losing some network information. We will address this issue in the future.

7. SIMULATION

We developed a discrete event-driven simulator based on DTN simulator to evaluate our congestion management strategy [1]. The simulator implements congestion management strategy as proposed in the previous section. To isolate the effect of link bandwidth on congestion management strategy, we assume that each link has infinite bandwidth.

7.1 Simulation Settings

Table 1: Simulation Parameters

Parameter	Value
Simulation field size	3000×3000 (m^2)
Transmission range	150 m
Number of static nodes	6/8
Number of mobile nodes	40/50

The simulated network consists of static nodes, destination, and mobile nodes distributed in a 3000 by 3000 $\times$ 3000 m^2 field. Static nodes are randomly distributed in the field and generate messages following poisson processes. Each static node can generate five classes of request messages, each with average generation probability of 0.2 message/second. Mobile nodes function as relay nodes and their mobility follows the random-way-point model with random initial location as well. The random way point model employed for mobile nodes has a moving speed uniformly distributed in [0.2, 0.5] meters/s and the pause time of a stop is uniformly distributed in [1, 2] seconds. The destination node has unlimited storage capacity and is randomly located in the field and ready to accept messages during opportunistic contacts. The storage capacity in each mobile node has a size of 50 messages. Since messages have to traverse lower layers of the network, they are ultimately subject to the restrictions there in term of maximum packet size. For example, on most IP networks it is safest to assume that single packet should be less than 1500 bytes long. Therefore, we assume that each message has a size of 1500 bytes [9]. We consider two scenarios with different mobile/static node mix: scenario 1 with 40/6 mix and scenario 2 with 50/8 mobile/static mix respectively. The parameters are summarized in Table 1.

We assume that there is an oracle for message routing. The oracle knows everything and can distribute routing information around the network [9]. Notice that the oracle is only responsible for message routing. Congestion control in each node is addressed by congestion management strategy.

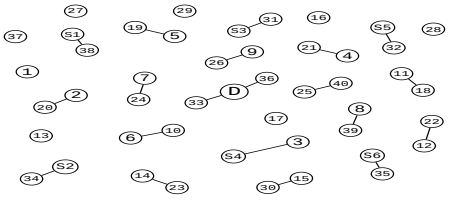


Figure 3: Nodes' position snapshot at 600s for 40/6 node mix for case listed in Fig. 5.

Fig. 3 shows a snapshot of the network. Here, destination is marked as D, static nodes are marked $S1 - S6$, mobile nodes are indexed 1-40, and the lines stand for existing links between nodes.

The main performance metrics of congestion management strategies in the simulation are throughput of the simulated network and the buffer utilization in each node. In order to evaluate the performance of our proposed scheme, we compare it (with dynamic opportunity cost based on remaining storage space) with a static congestion management strategy, where the opportunity cost is fixed in each node, under

the same routing scheme described above. We employ the function $w \log x$, where w is an adjustable weight, x is the remaining capacity of a node, to compute the salvage reward of remaining amount of capacity, the reason to choose such function is to guarantee that the salvage reward is concave in the remaining amount of capacity at the end of time horizon.

7.2 Simulation Results and Discussion

As we have discussed, we separate the congestion management mechanism from the route selection problem. Routing algorithm is based on the oracle in the system. Fig. 4 shows the distribution of hop-count of messages of two different mobile node/static node mixes under different congestion policies (scheme with dynamic opportunity cost and scheme with static opportunity cost). From Fig. 4, we can see that there is no significant difference in hop count among several simulation scenarios.

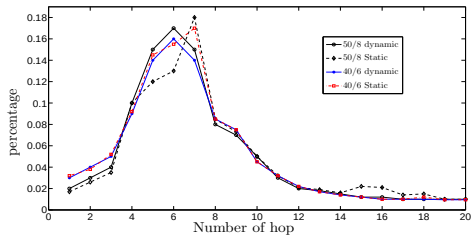


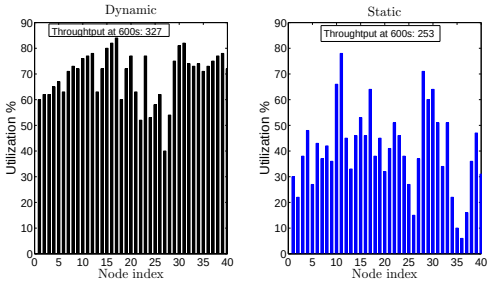
Figure 4: Hop count distribution

Fig. 5 show snapshots of buffer utilization under the two control policies with 40/6 node mix at 600s. Fig 5a is the snapshot of buffer utilization with arrival density $\lambda_1 = 1/2$ of 6 poisson processes. Fig. 5b is the snapshot with message arrival density $\lambda_1 = 5/9$ of 6 poisson processes. Fig. 6 shows the snapshots of buffer utilization with 50/8 node mix at 600s. Fig. 6a is the snapshot of buffer utilization with arrival density $\lambda_1 = 1/2$ of 8 poisson processes. Fig. 6b is the snapshot with message arrival density $\lambda_1 = 5/9$ of 8 poisson processes.

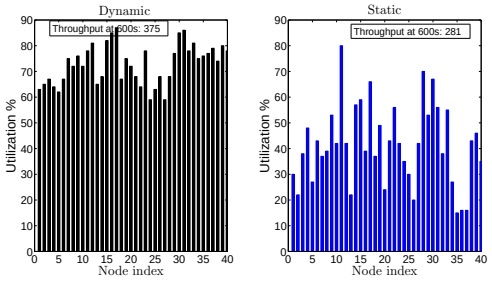
From Fig. 5 and 6, we can see that control policy with dynamic cost can achieve much evenly balanced loads and higher utilization in all the nodes of the network and better throughput. Since a node can not predict the coming request in advance, if the opportunity cost is fixed it is possible to reject the request even if the utilization is still low. Therefore it is a dangerous control to set a fixed cost to obtain optimization solution in revenue and utilization [18]. On the contrary, dynamic policy can adapt the opportunity cost in a node based on varying storage space and the space usage can be optimized.

Fig. 7 shows node utilization and throughput for 40/6 node mix at 600s, where simulation conditions are the same as those of Fig. 5a except that traffics are generated at constant rate (0.5 message/second) from the static nodes. Form Fig. 7, we can see that the dynamic policy can achieve better buffer utilization and throughput than the static policy even in this extreme case. For other scenarios, we have very similar results that are omitted here due to space limitation.

8. CONCLUSION



(a) traffic generated at constant speed 1/2 message/s



(b) traffic generated at constant speed 5/9 message/s

Figure 5: Load distribution in nodes - 40/6 node mix

In this paper, we have developed an optimal congestion management strategy for delay tolerant networks based on the concept of revenue management and dynamic programming. Relying only on the information of local storage space, our scheme can be readily applied to the dynamic and often unpredictable environments of delay tolerant networks. Our simulation results show that the proposed congestion management strategy can effectively outperform simple static, threshold based scheme.

In our future work, we plan to study how to jointly address the congestion management and routing issue in delay tolerant networks.

9. REFERENCES

[1] *DTN simulator*. [online] http://tuxmaster.usc.edu/.

[2] Y. Amir, B. Awerbuch, C. Danilov, and J. Stanton. A cost-benefit flow control for reliable multicast and unicast in overlay networks. *IEEE/ACM Trans. on Networking*, 13(5):1094–1106, October 2005.

[3] D. Bertsekas, A. Nedić, and A. Ozdaglar. *Convex Analysis and Optimization*. Athena Scientific, 2003.

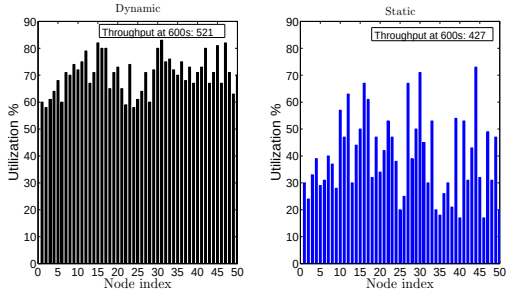
[4] D. P. Bertsekas. *Dynamic Programming and Optimal Control*. Athena Scientific, 2007.

[5] S. Burleigh, E. Jennings, and J. Schoolcraft. *Autonomous Congestion Control in Delay-Tolerant Networks*. [online] http://hdl.handle.net/2014/40636.

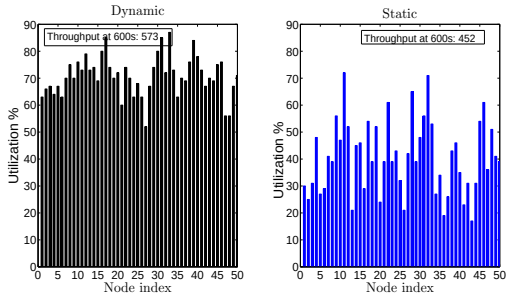
[6] B. Burns, O. Brock, and B. Levine. Mora routing and capacity building in disruption-tolerant networks. *Ad Hoc Networks*, 6(4):600–620, June 2008.

[7] K. Fall. A delay-tolerant network architecture for challenged internets. In *Proceedings of SIGCOMM'03*, pages 27–34, August 2003.

[8] K. Fall, W. Hong, and S. Madden. *Custody Transfer for Reliable Delivery in Delay Tolerant Networks*. [online]



(a) traffic generated at constant speed 1/2 message/s



(b) traffic generated at constant speed 5/9 message/s

Figure 6: Load distribution in nodes - 50/8 node mix

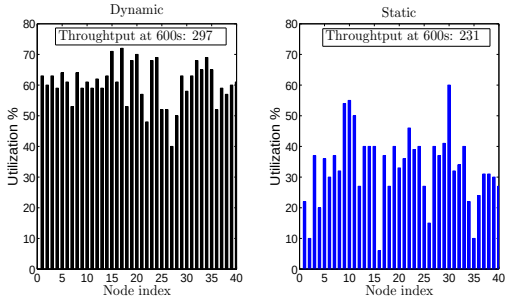


Figure 7: Load distribution in nodes - 40/6 node mix (traffic generated at constant speed 0.5 message/s)

http://www.dtnrg.org/papers/custody-xfer-tr.pdf.

[9] S. Farrell and V. Cahill. *Delay- and Disruption-Tolerant Networking*. Artech House, 2006.

[10] M. Grossglauser and D. Tse. Mobility increases the capacity of ad hoc wireless networks. *IEEE/ACM Trans. on Networking*, 10(4):477–486, August 2002.

[11] K. A. Harras and K. C. Almeroth. Transport layer issues in delay tolerant mobile networks. In *Proceedings of IFIP Networking Conference*, May 2006.

[12] C. Liu and J. Wu. Scalable routing in delay tolerant networks. In *Proceedings of Mobihoc'07*. ACM, September 2007.

[13] S. Merugu, M. Ammar, and E. Zegura. *Routing in Space and Time in Networks with Predictable Mobility*. [online] http://hdl.handle.net/1853/6492.

[14] M. Mukarram, B. Tariq, M. Ammar, and E. Zegura. Message ferry route design for space ad hoc networks with mobile nodes. In *Proceedings of Mobihoc'06*. ACM, May 2006.

[15] M. Park and V. Rodoplu. Traffic allocation in delay-tolerant wireless ad hoc networks. In *Proceedings of IEEE Wireless Telecommunications Conference*. IEEE, April 2006.

[16] M. Seligman, K. Fall, and P. Mundur. Alternative custodians for congestion control in delay tolerant networks. In *Proceedings of SIGCOMM'06 Workshops*, pages 229–236, Sept. 2006.

[17] T. Spyropoulos, K. Psounis, and C. Raghavendra. Efficient routing in intermittently connected mobile networks: The single-copy case. *IEEE/ACM Trans. on Networking*, 16(1):63–76, August 2008.

[18] K. Talluri and G. V. Ryzin. *The Theory and Practice of Revenue Management*. Kluwer Academic Publishers, 2004.

[19] Y. Wang and H. Wu. Dft-msn: The delay/fault-tolerant mobile sensor network for pervasive information gathering. In *Proceedings of IEEE INFOCOM'06*. IEEE, April 2006.

[20] J. Wu, S. Yang, and F. Dai. Logarithmic store-carry-forward routing in mobile ad hoc networks. *IEEE Trans. Parallel and Distributed Systems*, 18(6):735–748, June 2007.

[21] L. Ying and R. Srikant. Optimal delay-throughput trade-offs in mobile ad-hoc networks: Hybrid random walk and one-dimensional mobility models. In *Proceedings of ITA workshop*, 2007.

[22] W. Zhao, M. Ammar, and E. Zegura. A message ferrying approach for data delivery in sparse mobile ad hoc networks. In *Proceedings of Mobihoc'04*. ACM, May 2004.