A Percolation-Based Approach to Model DTN Congestion Control

Aloizio P. Silva *, Marcelo R. Hilário†, Celso M. Hirata* and Katia Obraczka§
*Inst. Tec. de Aeronáutica, São José dos Campos, São Paulo, Brasil
Email: aloizio@ita.br, hirata@ita.br
†Univ. Federal de Minas Gerais, Belo Horizonte, Brasil
Email: mhilario@mat.ufmg.br
§UC Santa Cruz, Santa Cruz, California, USA
Email: katia@soe.ucsc.edu

***Abstract*—In this paper, we propose a novel modeling framework to study congestion in delay- and disruption tolerant networks (DTNs). The proposed model is based on directed site-bond percolation where sites represent space-time position of DTN nodes, and bonds are contact opportunities, *i.e.* communication links that can be established whenever nodes come in range of each other. To the best of our knowledge, this is the first model of DTN congestion using percolation theory. The proposed modeling framework is simple yet general and can be used to evaluate different DTN congestion control mechanisms in a variety of scenarios and conditions. We validate our model by showing that its results match quite well results obtained from the ONE DTN simulation platform. We also show that our model can be used to understand how parameters like buffer management policy, buffer size, routing mechanism, and message time-to-live affect network congestion.**

I. Introduction

Delay- and disruption-tolerant networks (DTNs) refer to networks that are characterized by intermittent connectivity, long delays, and often constrained bandwidth. They were originally motivated by space exploration and its need for deep space communication [1]; however, over time, a diverse set of DTN applications for *extreme* environments have emerged including vehicular networks [2], emergence response and military operations [3], surveillance [3], tracking and monitoring applications [4], as well as bridging the digital divide [5]. In these scenarios, long delays arise as a consequence of either the fact that distances are long or that connections are episodic.

The DTN architecture proposed in [6] addresses message delivery challenges posed by intermittently connected networks using the *store-carry-and-forward* paradigm where messages are forwarded only when a contact opportunity arises. As a result, messages might remain stored for long periods of time in persistent storage at intermediate nodes before reaching their destinations. Because of the inability to guarantee end-to-end connectivity at all times, congestion may build up in the network. As a result, persistent storage at nodes may fill up and eventually overflow, causing data loss and consequently network performance degradation.

Understanding network congestion has motivated a number of research efforts that focus on modeling network behavior under congestion. Existing models employ a variety of techniques including renewal theory [7] [8], fixed point methods [9], fluid models [10], financial models [11], Markov chains [12], and control theory [13]. In this work, we develop a simple, yet general mathematical framework to model congestion in DTNs based on percolation theory [14]. Our goal is to use the resulting modeling framework to evaluate and validate existing and future DTN congestion control mechanisms. An important feature of the proposed percolation model is the fact that, instead of requiring global knowledge about the whole network, it relies exclusively on local information, i.e., information related to a node and its neighbors. As will become clear in the remainder of the paper, formulating the DTN congestion problem as a percolation process happens quite intuitively and the resulting percolation model is simple, general, and easy to derive.

To the best of our knowledge, our work is the first to model DTN congestion using percolation theory. The proposed model defines the probability of delivering a message between a given source-destination pair as the probability there exists a path between the source and destination containing only non-congested nodes (i.e., nodes that contain available buffer space for arriving messages) and active links (nodes in contact with one another). As a result, our model yields the following network congestion related metrics: average buffer occupancy, average buffer blocked time, average message delivery probability, and average delivery delay. Ultimately, our goal is to use the proposed model to evaluate DTN congestion control mechanisms in terms of how cost-effective they are in preventing/containing congestion.

The remainder of this paper is organized as follows. Section II describes DTN congestion problem and presents a brief overview of percolation theory while in Section III, the DTN congestion problem is described as a percolation problem. Section IV presents the experimental settings and defines some evaluation metrics which are used to evaluate our model. Section V presents an experimental analisys of the proposed model using both Matlab and the ONE simulator. Section VI investigates the related work for percolation theory applied at multi-hop ad hoc networks and DTN. Final remarks and conclusions are drawn in Section VII.

II. DTN Congestion and Percolation

In this section, we discuss DTN congestion and introduce definitions, notations, and results of percolation theory which will be used in our model. More details including proofs of percolation results discussed here can be found in [15].

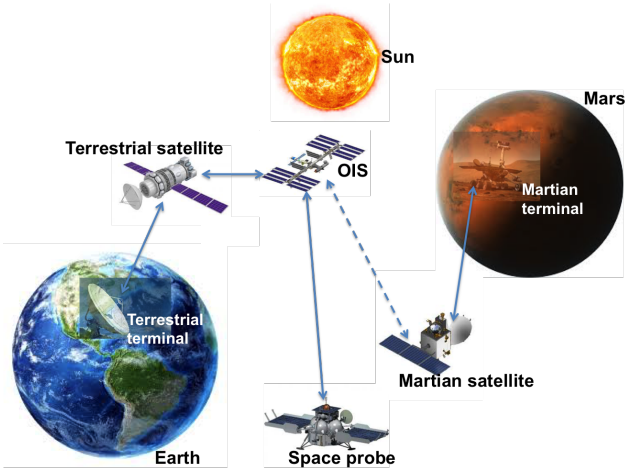


Figure 1. Deep space communication scenario

A. DTN Congestion

The challenges of controlling congestion in DTNs are mainly due to two reasons: (1) end-to-end connectivity between nodes cannot be guaranteed, and (2) arbitrarily long latency caused by high propagation delays and/or intermittent connectivity. Take for example the deep space communication scenario in Figure 1, in which a terrestrial terminal is connected to a terrestrial satellite. The terrestrial satellite is connected to an Orbital International Station (OIS) in orbit around the sun, which in turn connects to a Martian satellite and a space probe. Additionally, there is a Martian terminal connected to the Martian Satellite. The link between the OIS and the Martian Satellite is interrupted whenever the planet Mars is between the OIS and the orbiting satellite, as well as whenever the sun is between Mars and the OIS. Therefore, traffic on the "link" between the Terrestrial and Martian satellites may need to be buffered at the OIS for long- and varying periods of time. If the OIS becomes heavily congested, it will significantly hamper communication between the Terrestrial satellite and the space probe. Under these conditions, "traditional" end-to-end congestion control mechanisms, a la TCP, do not work well. As DTN nodes become congested, incoming messages may be discarded due to buffer overflow. This increases the drop rate and raises network overhead which leads to inefficient use of bandwidth and further worsens congestion conditions.

DTN congestion control has attracted considerable attention from the network research community. As a result, a number of congestion control techniques have been proposed [16], most of which have been evaluated empirically using network simulation platforms. Our goal, in this work, is to propose a simple, yet general framework that can model DTN congestion mathematically and that can be used to study the performance of DTN congestion control solutions, complementing and validating empirical studies. We explored a number of modeling approaches and found that percolation theory is well suited to describe congestion in DTNs. In the next section, we provide a brief overview of percolation theory and describe how we apply it to the DTN congestion problem.

B. Percolation Theory Overview

Percolation was first introduced in the mathematics literature motivated by the problem of how fluids flow (or percolate) in different materials (or media) [14]. It immediately caught the attention of mathematicians and physicists for its simplicity and wide applicability [15]. To date, percolation has been employed in a variety of contexts ranging from complex networks, control of epidemic diseases, and wildfires.

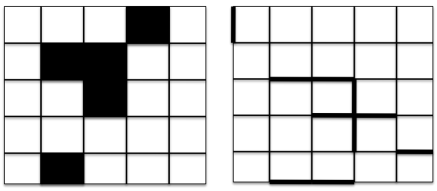
In percolation, the medium or network is modeled as a graph. A graph is a pair (V, E), where V is a countable set of points called vertices or sites, and E is a set of edges, *i.e.* unordered pairs of vertices $\langle v, w\rangle$, also called bonds. When $\langle v, w\rangle \in E$, we say that v and w are adjacent. A set of distinct vertices $\{v_1, v_2, v_3, \ldots, v_n\} \subset V$ is called *path* when consecutive vertices are adjacent in the lattice. The graph distance between two vertices is defined to be the minimum amount of bonds necessary in order to establish a path that connects them. Consider each vertex $v \in V$ to be open with probability ρ and closed with probability $1 - \rho$, independently of every other vertex. Thus we can define P_ρ as being the probability distribution that describes the state of the graph as a whole. This distribution is defined in the sampling space $\Omega = \{0, 1\}^V$. Elements $\omega \in \Omega$ are represented as $\omega = (\omega(v) : v \in V)$. So, when $\omega(v) = 0$ we say that v is closed whereas, when $\omega(v) = 1$, we say that v is open.

Given a configuration $\omega \in \Omega$, we say that a path is open if all of its vertices are open, that is, $\omega(v_i) = 1$ for all $\{i = 1, 2, \ldots, n\}$. Two sites x and y in V are said to be connected $(x \leftrightarrow y)$ if there is a open path $\{v_1, v_2, \ldots, v_n\}$ where $x = v_1$ and $y = v_n$.

Given one configuration $\omega \in \Omega$ and a vertex x, we can consider the set of all vertices that are connected to x. This set is called the open cluster of x in the configuration ω and it is represented by $C_x(\omega)$, or simply by C_x. More formally, we have $C_x = \{y \in V; \omega \in (x \leftrightarrow y)\}$. When x is equal to the origin $0 \in \mathbb{Z}^d$ (where $\mathbb{Z}$ is an Euclidian integer lattice with dimension d) we drop the subscript and write $C = C_0$ to denote the open cluster containing the origin. Vertices in the lattice are called *sites* and edges *bonds*. In the so-called *bond percolation* models, bonds are declared "open" with probability ρ, or "closed" with probability $1-\rho$. In the context of percolation's original application, open bonds correspond to channels through which fluid can flow. Open paths consist of a sequence of adjacent open bonds together with their end-vertices. Sites are declared "unblocked" with probability ρ or "blocked" with probability $1 - \rho$. An open path is a sequence of neighboring unblocked sites. Two sites belong to the same open connected component, or simply to the same cluster, if they are linked by an open path.

To illustrate bond- and site percolation, we consider the square lattice shown in Figure 2.

- In site percolation (see Figure 2a), we view the lattice as a rectangular array of squares. Each square is declared to be unblocked with probability ρ (black squares), and blocked with probability $1 - \rho$ (white squares). A *cluster* is defined as a maximal connected set of neighboring unblocked squares.
- In bond percolation (see Figure 2b), the lattice is a



(a) Site Percolation (b) Bond Percolation

Figure 2. Types of percolation

graph consisting of horizontal and vertical edges. Each lattice edge is open (thicker) with probability ρ, and closed (thinner) with probability $1-\rho$. A *cluster* is defined as maximal connected subgraph of the lattice consisting of open edges.

Bond- and site percolation are commonly used to study the connectivity properties of clusters in the lattice as a function of the parameter ρ. The main question is to determine wether the probability that there exists an infinite open path starting at the origin is non-zero, or equivalently, wether the origin belongs to an infinite cluster with non-zero probability.

In order to model congestion in DTNs, we use another formulation called directed site-bond percolation. In this formulation, both sites and edges have their states assigned randomly and independently. Furthermore edges will be oriented in order to represent the flow of information in the lattice. In the proposed percolation-based DTN congestion model: (1) *open bonds* represent contacts between DTN nodes, (2) *blocked sites* represent DTN nodes whose buffers are 100% occupied (i.e., congested nodes), and (3) *unblocked sites* represent DTN nodes that are not congested. Our model allows us to derive important network performance metrics such as average message delivery probability, average buffer occupancy ratio and blocked time, as well as average message delivery delay. These metrics are defined in Section IV-B. Our model also allows us to understand the impact of parameters such as buffer management policies, routing mechanisms, message time-to-live on network congestion control.

III. DTN Congestion Model

DTN congestion can be expressed as a percolation problem by examining whether there exists a path between data sources and destinations and whether the paths are open, i.e., non-congested. In other words, the probability of delivering data from a node x to a node y is equivalent to the probability of finding an open path between x and y. As such, our model tries to find, in the graph represention of the network, *open paths*, i.e, sequences of adjacent *open edges* and *unblocked sites* over which data can be delivered from sources to destinations. To this end, in addition to finding end-to-end paths between DTN sources and destinations, the model also considers buffer occupancy at each node along the path.

While our percolation-based model is quite general and applicable to lattices of arbitrary dimension d, for simplicity, in this work we focus on the special case of $d=1$ which still captures the main DTN congestion features that we want to study. The experimental validation of our model presented in Section V-A shows that results obtained from the model match quite well simulations run on the ONE DTN simulator [17]. Figure 3 shows the DTN representation we use in our percolation model; the network is represented as a collection of nodes in a domain $\mathbb{S} \subset \mathbb{Z}^d$, where each node is a vertex in a $d=1$ lattice, i.e., a line. We model the state evolution of the nodes by adding the temporal dimension, which is represented by parallel lines labeled $t_0, t_1, \cdots, t_n$ as shown in Figure 3. Our model is thus defined in $\mathbb{Z}^{d+1} = \mathbb{Z}^d \times \mathbb{Z}^1$. A site in $\mathbb{Z}^{d+1}, d=1$ will be denoted by (x,t) where $x \in \mathbb{Z}^d$ represents the position of the site at time $t \in \mathbb{Z}$.

As previously highlighted, in order to establish whether a node x is blocked (congested) or unblocked (non-congested) at time t, our model calculates x's buffer occupancy given by: x's buffer occupancy at time $t-1$, plus the number of new messages created at x, plus the number of incoming messages at x received from its neighbors, minus the number of outgoing messages from x. Below, we proceed to calculate these different buffer occupancy components. Table I summarizes the variables used in our model and their definitions.

We consider that, at each time t, a node x can only forward a message to its left and/or right neighbors ($x-1$ and $x+1$, respectively). This corresponds to the SE (southeast) and SW (southwest) oriented edges in Figure 3, respectively. We use vertical edges to represent the fact that a node can keep messages stored locally from time t to $t+1$. These edges are represented as dashed lines in Figure 3.

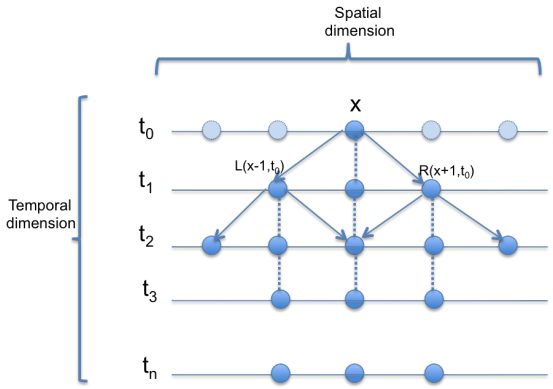


Figure 3. Network model

Let $\mathcal{Q}$ be a positive integer denoting the buffer capacity at nodes, *i.e.* the maximum number of messages that can be stored at each node. Let $\mathcal{M}(x,t)$ denote the set of messages stored at node x at time t and $m(i,x,t) \mid i = \{1,2,\ldots,\mathcal{M}(x,t)\}$ refers to an individual message stored at x at time t.

For a message $m(i,x,t)$:

$$f(i,x,t) = \begin{cases} 1 & \text{if } m(i,x,t) \text{ has already been forwarded,} \\ 0 & \text{if } m(i,x,t) \text{ has not been forwarded yet.} \end{cases} \quad (1)$$

For each site $(x,0)$, let us set an initial buffer occupancy $O(x,0)$. Here we assume that $\{O(x,0); x \in \mathbb{Z}\}$ is initially empty. We say that node x is blocked at time t if $O(x,t) = \mathcal{Q}$.

We define $B(x,t)$ as the number of messages created at x at time t; $B(x,t)$ is an independent random variable given by

Table I. DTN PERCOLATION MODEL VARIABLES AND DEFINITIONS

Variable Name	Description
$\mathcal{N}(x)$	number of neighbors of x
$\mathcal{Q}$	maximum number of messages that can be stored in a node's buffer
$\mathcal{M}(x,t)$	set of messages stored at node x at time t.
$m(i,x,t)$	message i stored at x at time t
$f(i,x,t)$	specifies whether message $m(i,x,t)$ stored at x at time t was forwarded or not
$l(x_i,j,x,t)$	specifies whether message $m(j,x_i,t)$ forwarded by neighbor x_i already exists in x's buffer at time t
r_c	node's transmission radius
$O(x,t)$	x's buffer occupancy at time t
$B(x,t)$	number of messages generated by x at time t
E	set of all bonds
$\mathcal{S}(x,t)$	number of outgoing messages at x at time t
$\mathcal{D}(x,t)$	number of drop messages at x at time t
$\mathcal{C}_{msg}(x,t)$	number of messages consumed by applications running on x at time t
$F(x,t)$	number of messages forwarded by x at time t
e	bond between two sites/nodes
$\eta(e)$	specifies whether a bond between x and x_i is open or closed
β	specifies message forwarding mechanism: replication if $\beta = 0$ or "plain" forwarding if $\beta = 1$
$\omega(x,t)$	specifies whether x blocked (congested) or unblocked (uncongested) at time t
Γ	path that contains only unblocked sites and open bonds
$\kappa(x,t)$	number of messages received from right neighbor at time t
$\nu(x,t)$	number of messages received from left neighbor at time t
$L(x,t)$	number of messages forwarded to left neighbor at time t
$R(x,t)$	number of messages forwarded to right neighbor at time t

a Bernoulli distribution with parameter $\lambda \in (0,1)$ called the message generation rate. On average, a message is created at x every $1/\lambda$ unities of time.

Let $r_c > 0$ be an integer denoting the node's transmission radius. We add a bond for each pair of sites $e = ((x,t),(x_i,t+1))$ if $|x_i - x| \leq r_c$. Let us denote by E be the set of all bonds obtained by this procedure (Figure 3 presented an illustration for the case $d = 1$ and $r_c = 1$). We define $\{\eta(e) \mid e \in E\}$ a family of independent random variables with Bernoulli parameter ρ indexed by the bonds of the network. We say that the bond e is open if $\eta(e) = 1$ and closed otherwise. This allows us to model node encounters in DTNs, i.e., when a bond $e = ((x,t),(x_i,t+1))$ is open, nodes x and x_i are in contact and thus can communicate. This is also how we model the effect of node mobility which influences the appearances and disappearances of bonds in the lattice.

Now, let $\mathcal{S}(x,t)$, which is given by Equation 2, be the total number of outgoing messages at node x at time t. $\mathcal{S}(x,t)$ includes the number of messages discarded ($\mathcal{D}(x,t)$), the number of messages that have been consumed by applications running on x ($\mathcal{C}_{msg}(x,t)$), as well as the number messages forwarded by x ($F(x,t)$). Note that to calculate $F(x,t)$, we use $\beta \in \{0,1\}$ which specifies whether the forwarding mechanism uses replication or not. More specifically, if $\beta = 0$, a copy of the message is forwarded and the original message is maintained in the buffer. Otherwise, if $\beta = 1$, the original message is itself forwarded and thus not maintained in the buffer. Indeed, this is how we model the underlying routing mechanism being used, i.e, whether it is based on replication or "plain" forwarding. $\mathcal{D}(x,t)$ represents the number of messages discarded either because the message time-to-live (TTL) has expired or because the buffer filled up. In the latter case, messages will be discarded according with a pre-defined drop policy [16]. In the experimental evaluation of our model (see Section V), we vary the drop policy and observe its impact on network congestion.

Recall that, at time t, a node x can only forward messages to nodes $x-1$ (left neighbor) and $x+1$ (right neighbor). Thus, the number of messages forwarded by x, $F(x,t)$ can be written as $F(x,t) = L(x,t) + R(x,t)$, where $L(x,t)$ is the number of messages forwarded by x to its left neighbor $x-1$ at time t, and $R(x,t)$ is the number of messages forwarded by x to its right neighbor $x+1$ at time t.

$$\mathcal{S}(x,t) = \mathcal{D}(x,t) + \mathcal{C}_{msg}(x,t) + \beta \overbrace{\sum_{i=1}^{O(x,t)} f(i,x,t)}^{\text{F(x,t)}} \quad (2)$$

According to Equation 3, $\kappa(x,t)$, the number of messages received by x from its right neighbor, depends on whether there is an edge between x and its right neighbor during $(t,t+1)$, which is determined by $\eta((x+1,t),(x,t+1))$. Clearly, if x and $x+1$ are not in contact with one another $\kappa(x,t) = 0$. If there is an edge between x and $x+1$, the number of messages received (and stored) by x is given by $L(x+1,t)$ minus the number of $x+1$'s messages which x already has in its buffer. These messages, if exchanged by the two nodes, will be discarded by x as duplicates. The number of duplicate messages is given by $\sum_{j=1}^{O(x+1,t)} l(x+1,j,x,t)$, where $l(x+1,j,x,t)$ is defined by Equation 5.

$$\kappa(x,t) = \eta((x+1,t),(x,t+1))[L(x+1,t) - \sum_{j=1}^{O(x+1,t)} l(x+1,j,x,t)]. \quad (3)$$

Similarly, $\nu(x,t)$, the number of messages received by x from its left neighbor is expressed by Equation 4.

$$\nu(x,t) = \eta((x-1,t),(x,t+1))[R(x-1,t) - \sum_{j=1}^{O(x-1,t)} l(x-1,j,x,t)]. \quad (4)$$

$$l(x_i, j, x, t) = \begin{cases} 1 & \text{if } f(j, x_i, t) = 1 \text{ and } m(j, x_i, t) \in \mathcal{M}(x, t), \\ 0 & \text{otherwise.} \end{cases} \quad (5)$$

We can now write the expression for the buffer occupancy at x at time $t+1$ in Equation 6 as x's buffer occupancy at time t, plus the amount of new messages created at x at time t, minus $S(x,t)$, the number of messages leaving x at time t, plus the number of messages received from x's right neighbor at time t (if a contact existed between them), plus the number of messages received by x at t from x's left neighbor (if a contact existed between them).

$$O(x, t+1) = \{(O(x,t) + B(x,t) - \mathcal{S}(x,t) + \kappa(x,t) + \nu(x,t)) \wedge \mathcal{Q}\}, \quad (6)$$

where for any real numbers a and b, $a \wedge b \equiv \min\{a, b\}$.

We then say that x is blocked at time t when Equation 7 is true.

$$\{O(x,t) + B(x,t) - \mathcal{S}(x,t) + \kappa(x,t) + \nu(x,t)\} \geq \mathcal{Q}, \quad (7)$$

and define:

$$\omega(x,t) = \begin{cases} 0 & \text{if } x \text{ is blocked in time } t, \\ 1 & \text{if } x \text{ is not blocked in time } t. \end{cases} \quad (8)$$

Recall that our goal is to find open paths Γ, i.e., sequences

$$\Gamma = [(x^0, t^0), (x^1, t^1), (x^2, t^2), (x^3, t^3), \ldots, (x^n, t^n)] \quad (9)$$

such that

$$((x^i, t^i), (x^{i+1}, t^{i+1})) \in E \quad \forall \quad i = 0, \ldots, n-1 \quad (10)$$

and, in addition,

- $\omega(x^i, t^i) = 1 \ \ \forall \ \ i = 0, \ldots, n$ and
- $\eta((x^i, t^i), (x^{i+1}, t^{i+1})) = 1 \ \ \forall \ \ i = 0, \ldots, n-1$.

As shown in the results presented in Section V, using our model we can compute network performance metrics such as buffer occupancy, buffer block time, message delivery probability, and delivery delay which are congestion indicators. Our model can also be used to evaluate the effectiveness of different congestion control mechanisms.

IV. Experimental Methodology

The evaluation of our model was carried out in two ways. First, we validate the model by comparing its output against simulation results from the ONE DTN simulator [17]. Then, we use our model to study how parameters like buffer management policy, buffer size, routing mechanism, and message time-to-live (defined in Section IV-B below) affect network congestion and can be used to perform congestion control. We implemented our model in MatLab.

A. Experimental Setup

For experiments presented in Section V, the parameters of our model and their values are summarized in Table II while the parameters of the ONE simulator [17] and their values are listed in Table IV.

Some considerations about the implementation of our model are noteworthy. For example, when a message is delivered to its intended destination, it is consumed by the application running at the destination within $TimeApp$ seconds. This is the period of time the message stays in the node's buffer after being received and before being removed from the buffer. In our experiments, unless stated otherwise, we use $TimeApp = 0$, which means that the message is promptly removed from the buffer as soon as it arrives. Each time two nodes are in contact with one another, they try to exchange all their stored messages if there is available space in their buffers. To this end, both nodes compute the buffer occupancy according to Equation 6. If the node's buffer is considered blocked (see Equation 7), the buffer management policy of choice is activated to discard the appropriate messages. Four drop policies have been considered in our experiments (see Table III). Each message has a TTL (Time-To-Live) whose value is decremented by one unit at every clock tick. When a message's TTL expires, the message is automatically removed from the system.

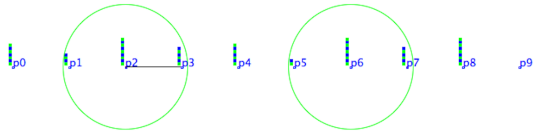


Figure 4. Snapshot of the ONE simulator scenario for 10 DTN nodes.

In order to validate our model against the ONE simulator [17], we run ONE simulations using the line topology shown in Figure 4). We set the node transmission range such that a node can only communicate with its two adjacent neighbors in the line topology as depicted in Figure 4). To be able to simulate using the ONE the two forwarding strategies that we use in our model, namely replication and single-copy forwarding, we use two of the ONE's routing schemes. Replication in our model ($\beta = 0$) is simulated using *epidemic routing* [18] in the ONE, which replicates a message and forwards it to all nodes that are within range (i.e., nodes that have open bonds with the current node). To match single-copy forwarding ($\beta = 1$), we use the ONE's implementation of the *first contact protocol* [19], according to which only one copy of a message exists in the network at any point in time. This means that a message is relayed to the first encountered node

Table III. Drop policies

Drop Policy	Description
DROP-INCOMING	the incoming message is dropped.
DROP-HEAD	the first message in the buffer, i.e., the head of the queue, is dropped.
DROP-LAST	the newly received message is first removed.
DROP-OLDEST	the message with the shortest remaining life time (closest to TTL expiration) is dropped.

Table II. SIMULATION PARAMETERS AND THEIR VALUES FOR MATLAB EXPERIMENTS

Parameters		
Name	Description	Value
Discrete clock time	simulation time	1000 seconds
TTL	messages time-to-live (TTL)	[100, 200, 300, 400, 500] seconds
Lambda (λ)	message generation rate	[1/5, 1/10, 1/20, 1/30]
BufferSize	buffer capacity in number of messages	[10, 20, 30 ,40, 50 , 60] messages
Rho (ρ)	connection success probability	0.5
BufferPolicy	message drop scheme if congestion happens	[DROP-INCOMING, DROP-HEAD, DROP-LAST, DROP-OLDEST]
Nodes	number of nodes	50
TimeApp	once the message arrived at the destination how long it will take for it to be consumed by the application	0 seconds
Beta (β)	forwarding strategy (replication or single-copy)	[0, 1]
MessageSize	message size	1 KB

it is a duplicate, and is then removed from the message buffer of the previous hop.

Table IV. SIMULATION PARAMETERS AND THEIR VALUES FOR ONE SIMULATIONS

Parameters		
Name	Description	Value
Scenario.endTime	simulation time	1000 seconds
btInterface.transmitSpeed	bandwidth	2.5 Mbps
btInterface.transmitRange	transmitting range	26 m
Group.router	routing protocol	[EpidemicRouter,FirstContactRouter]
Group.movementModel	mobility model	LinearFormation
Group.bufferSize	node buffer size	[10, 20, 30, 40, 50, 60] KB
Group.msgTTL	message time to live	[100, 200, 300, 400, 500] seconds
Group.nrofHosts	number of nodes in network	50
Movimentmodel.worldSize	area where simulation takes place	1km x 1km
Events1.size	message size	1KB
Events1.interval	i.e. one new message every 1 to 5 seconds	[1-5, 1-10, 1-20, 1-30] seconds

In the ONE simulator, two nodes are in contact whenever they are within the communication range of one another. So, if the separation distance between the nodes is less than the r_c, all links will be always active. In order to simulate the fact that node neighborhoods change, we simulate links between neighbors going up and down according to a Bernoulli distribution with parameter ρ. To this end, we modify the ONE simulator to include this feature and set $\rho = 0.5$ to match our model.

B. Evaluation Metrics

One of the goals of our model is to describe the congestion problem taking into account the main characteristics of a DTN, for example, limited resources, high delays, and intermittent connectivity. Additionally, we also want to be able to analyze DTN congestion under a variety of scenarios and conditions (e.g., deep space communication).

To this end, we define the following metrics that can be derived from the proposed model. These metrics not only provide us with insight into the performance of existing congestion control mechanisms but also help us to propose new ones that can be more cost-effective. We also use these metrics to cross-validate our model against simulation results from the ONE simulator.

1) **Average buffer occupancy** is given by the average ratio between the number of messages stored at a node and the size of the node's buffer; in our model it is equivalent to $O(x,t)$.
2) **Average buffer blocked time** is defined by the average time that each site remains blocked (or congested). It corresponds to $\omega(x^i, t^i)$ in our model.
3) **Average message delivery probability** is given by the probability of existence of a space-time path that connects the message source to its final destination. This metric allows us to infer network congestion levels and corresponds in our model to the probability that Γ exists.
4) **Delivery delay** is defined as the total time elapsed between message generation and its delivery and is equivalent, in our model, to the time it takes for a message to be delivered over Γ, if Γ exists.

V. RESULTS

A. Model Validation

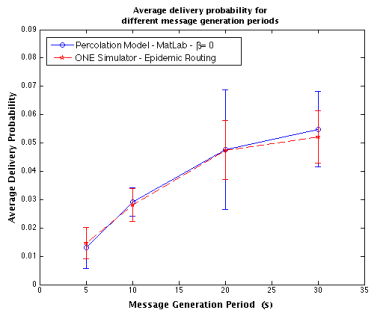
We validate our model by comparing its results obtained from its Matlab implementation against simulation results from the ONE DTN simulator. To evaluate the impact of congestion on network performance, we try to generate enough load to congest the network. Figure 5 shows the average message delivery probability as a function of the message generation period resulting from our model and from the ONE simulator. We observe that both curves essentially overlap with an average percentage difference [1] of 1.58% in the case of Figure 5a and 0.96% in Figure 5b. As expected, for longer message generation periods (i.e., lower message generation rate, or λ in our model), which results in less messages generated, the average delivery probability increases. Note that average delivery probabilities from this experiment are quite low since no congestion control scheme has been used. In this case, the drop policy is frequently activated, causing messages to be discarded before they are delivered. According to Figures 5a and 5b, we observe that the delivery probability when using $\beta = 0$ (epidemic routing) is slightly lower than when using $\beta = 1$ (first contact routing) because in the former case there are more message copies in the network resulting in buffer overflow and consequently increased drop ratios. As a result the average delivery probability is lower.

Figure 6 shows the delivery latency for different TTLs. Similarly to the average delivery probability results, the curves resulting from our model and the ONE simulator are quite close: the average percentage difference between the latencies obtained from our model and results from the ONE is 6.68% when $\beta = 0$ and 3.06% when $\beta = 1$ (Figures 6a and 6b respectively). We observe that when the message time-to-live increases, the latency has a tendency to increase. This is because higher message time-to-live results in a greater number of messages in the network, which in turn generates overload. Consequently, it increases the message delivery time since the messages stay longer into network.

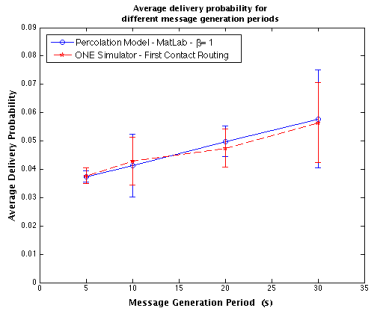
B. Understanding Network Congestion

Here we use our model to understand network behavior under congestion and how certain policies and parameters can

[1] The percentage difference comparison calculates the percentage difference between two values in order to determine how close they are.

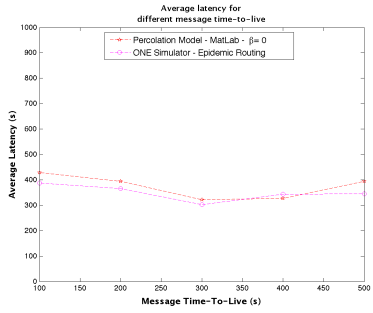


(a)

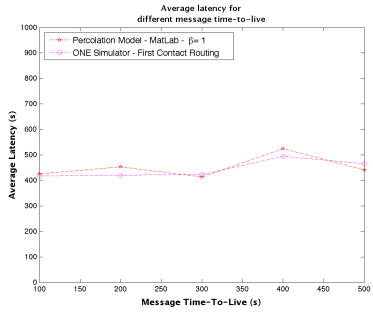


(b)

Figure 5. Average delivery probability for different message generation periods (TTL of $300s$, Drop-policy $DROP - OLDEST$, buffer size of $60\,kbytes$).



(a)



(b)

Figure 6. Average latency for different message time-to-live (Drop-policy $DROP - OLDEST$, buffer size of $60\,kbytes$, message generation period of $5s$). The standard deviation for the results from our model are 46.49 for $\beta = 0$ and 43.70 for $\beta = 1$. For the ONE experiments, we obtain a standard deviation of 31.40 for $\beta = 0$ and 34.26 for $\beta = 1$.

control congestion. We begin with the impact of different drop policies on delivery probability which is shown in Figure 7.

What we observe is that the delivery probability for different policies does not vary considerably when $\beta = 0$. However, when $\beta = 1$, we note that the variability of the behavior among the different drop policies increases, favoring "drop-head" and "drop-last".

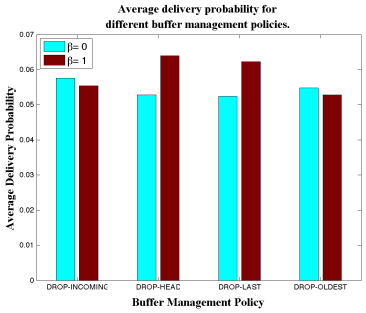


Figure 7. Message delivery probability for different drop policies with $\beta = 0$ and $\beta = 1$ (TTL of $300s$, buffer size of $60\,kbytes$, message generation period of $30s$).

Figure 8 shows that, as expected, increasing the buffer size results in shorter buffer block times.

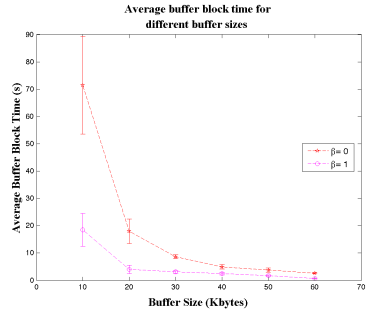


Figure 8. Average buffer block time for different buffer sizes with $\beta = 0$ and $\beta = 1$ (TTL of $300s$, message generation period of $5s$).

From Figure 9, we observe that higher time-to-lives increase average buffer block time. This is mainly because messages can stay longer in the buffer until being forwarded or delivered. As a result the buffer becomes full faster.

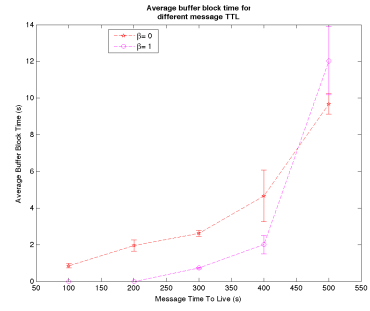


Figure 9. Average buffer block time for different message time-to-live with $\beta = 0$ and $\beta = 1$ (buffer size of $60\,kbytes$, message generation period of $5s$).

Figure 10 shows that as the buffer size increases, so does the average buffer occupancy. Basically, for higher buffer sizes the node can store more messages, and consequently the buffer occupancy ratio has a tendency to increase.

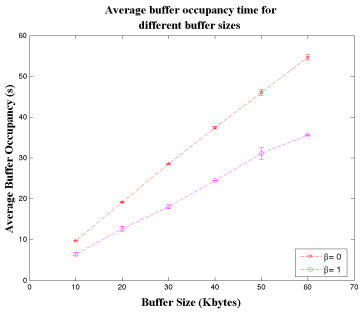


Figure 10. Average buffer occupancy for different buffer sizes with $\beta = 0$ and $\beta = 1$ (TTL of $300s$, message generation period of $5s$).

As it can be seen in Figure 11, for higher time-to-live values, buffer occupancy tends to increase. The reason for this is that messages stay in the buffer longer. However, beyond a certain value of the TTL, buffer occupancy does not change significantly. This is probably due to the fact that messages have enough time to be delivered to their intended destination. For example, in Figure 11, for TTL values greater than $400s$, the average buffer occupancy does not vary significantly.

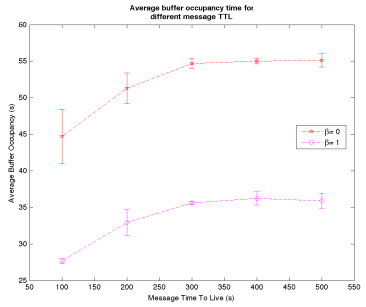


Figure 11. Average buffer block time for different message time-to-live with $\beta = 0$ and $\beta = 1$ (buffer size of $60\,kbytes$, message generation period of $5s$).

VI. Related Work

In this section we review prior research efforts focusing in the two areas related to our work, namely: percolation theory applied to mobile ad-hoc networks (MANETs) and network congestion modeling.

A. Applications of Percolation in MANETs

Since MANETs consider propagation of information over a random structure, percolation theory offers a theoretical framework to study the behavior of such systems. Recent studies [20] [21] [22] [23] [24] showed the existence of phase transition phenomena in MANETs using percolation theory. They showed, for instance, the existence of a connectivity threshold needed to guarantee the communications in the network. In addition these studies evaluate the probability of routing success when links go up and down, and, like our work, they argue that the evolution of the network over time generates many more possible configurations, enabling the percolation cluster to become much larger.

Space-time percolation, detection by mobile nodes, and information dissemination have attracted the attention of researchers ([25] [26] [23] [27]). In [27] a critical percolation threshold for aligned cylinders was derived, which provides a lower bound for the required node degree as it relates to the performance of opportunistic networking. Nodes are assumed stationary and it is shown that opportunistic content dissemination schemes, such as "floating content", can be analyzed by using a three-dimensional continuum percolation model.

A study of DTN and its capacity to store, carry, and forward messages so that messages eventually reach their final destination(s) was presented recently [28]. Percolation theory is used to characterize the mean density of nodes required to support communication in DTNs. In [29], a lower bound for node buffer size in intermittently connected wireless networks is described using percolation theory.

B. Modeling Network Congestion

A variety of methods have been employed to model congestion in traditional networks, including queuing [30] [31], auto-regression [32] [33], and Markov chains [32] [33]. Our work complements this prior work by considering congestion in DTNs.

There are also some studies that treat the congestion problem using financial models. For instance, market theory suggests that differentiated service can arise from a pricing scheme based on the level of congestion. As the resource becomes overloaded, only those who are willing to pay higher usage prices remain on the network. An economic pricing model is used by [11] to study the congestion problem and how to mitigate it in DTNs.

An analytical framework based on bulk arrival and bulk service queues to model DTN node behavior is proposed in [12]. The authors compute the stationary discrete probability densities of the sizes of the arriving bulks. In addition, they investigate a class of forwarding strategies, based on epidemic routing, used by DTN nodes where an expression for the average buffer occupancy is derived.

VII. Conclusion and Future Work

In this paper, we introduce the first DTN congestion model based on percolation theory, in particular, site-bond percolation. Using percolation theory to describe congestion in DTN scenarios allows us to propose a simple yet general model that derives the probability of delivering a message between a given source-destination pair. The resulting delivery probability is the probability that there exists a path that contains only uncongested nodes and active contact opportunities in the network. Ultimately, our goal is to use the proposed model to evaluate DTN congestion control mechanisms in terms of how cost-effective they are in preventing/containing congestion. Additionally, we use the theory of percolation to derive performance metrics that are key to understand DTN congestion and how to mitigate it.

The current model considers epidemic- and single-copy forwarding as data forwarding strategies. We plan to add to the model other forwarding strategies such as Prophet, Spray

and Wait, Spray and Focus, etc. We will also extend our model to 2- and 3 dimensions as well as incorporate node mobility explicitly.

REFERENCES

[1] "Deep space network," http://deepspace.jpl.nasa.gov, jet Propulsion Laboratory, California Institute of Technology. Accessed in February 2012.

[2] M. Gerla and L. Kleinrock, "Vehicular networks and the future of the mobile internet," *Computer Networks, Elsevier*, vol. 55, no. 2, pp. 457–469, 2011.

[3] J. P. Sterbenza, D. Hutchisonb, E. K. Cetinkayaa, A. Jabbara, J. P. Rohrera, M. Schollerc, and P. Smithb, "Resilience and survivability in communication networks: Strategies, principles, and survey of disciplines," *Computer Networks, Elsevier*, vol. 54, no. 8, pp. 1245–1265, June 2010.

[4] U. Lee, B. Zhou, M. Gerla, and E. Magistretti, "Mobeyes: smart mobs for urban monitoring with a vehicular sensor network," *Wireless Communications, IEEE*, vol. 13, no. 5, pp. 52–57, October 2006.

[5] L. Pelusi, A. Passarella, and M. Conti, "Opportunistic networking: data forwarding in disconnected mobile ad hoc networks," *IEEE Communication Magazine*, vol. 44, no. 11, pp. 134–141, November 2006.

[6] K. Fall, "A delay tolerant netwok architecure for challenged internets," in *the 2003 conference on Applications, technologies, architectures, and protocols for computer communications*, SIGCOMM'03. ACM, 2003, pp. 27–34.

[7] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose, "Modeling tcp throughput: A simple model and its empirical validation," in *ACM SIGCOMM98*, 1998, pp. 303–314.

[8] M. Mathis, J. Semke, J. Mahdavi, and T. Ott, "The macroscopic behaviour of thetcp congestion avoidance algorithm," *Computer Communication Review*, vol. 27, no. 3, 1997.

[9] C. Casetti and M. Meo, "Anew approach to model the stationary behaviour of tcp connections," in *INFOCOM 2000 Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies*, 200.

[10] V. Misra, W. B. Gong, and D. Towsley, "Fluid-based analysis of a network of AQM routers supporting tcp flows with application to RED," in *ACM SIGCOMM'00*, 2000, pp. 151–160.

[11] S. Burleigh, E. Jennings, and J. Schoolcraft, "Autonomous congestion control in delay tolerant networks," in *AIAAA SpaceOps'06*, 2006.

[12] M. Cello, G. Gnecco, M. Morchese, and M. Songuineti, "A model of buffer occupancy for ICNs," *IEEE Communication Letters*, vol. 16, no. 6, 2012.

[13] D. Cavendish, "A control theoretical approach to congestion control in packet networks," *IEEE/ACM Transactions on Networking*, vol. 12, no. 5, pp. 893–906, 2004.

[14] S. R. Broadbent and J. M. Hammersley, "Percolation process I: Crystals and mazes," in *Cambrige Philosophical Society*, 1957, pp. 627–641.

[15] G. Grimmett, *Percolation*. Springer, 1999.

[16] A. P. da Silva, S. Burleigh, C. M. Hirata, and K. Obraczka, "A survey on congestion control for delay and disruption tolerant networks," *Ad Hoc Networks*, August 2014, elsevier.

[17] A. Keranen, J. Ott, and T. Karkkainen, "The ONE simulator for DTN protocol evaluation," in *The 2nd International Conference on Simulation Tools and Techniques*, 2009.

[18] A. Vahdat and D. Becker, "Epidemic Routing for Partially Connected Ad Hoc Networks," Tech. Rep., July 2000.

[19] S. Jain, K. Fall, and R. Patra, "Routing in a delay tolerant network," *SIGCOMM Comput. Commun. Rev.*, vol. 34, no. 4, pp. 145–158, Aug. 2004. [Online]. Available: http://doi.acm.org/10.1145/1030194.1015484

[20] P. Basu, S. Guha, A. Swami, and D. Towsley, "Percolation phenomena in networks under random dynamics," in *Communication Systems and Networks (COMSNETS), 2012 Fourth International Conference on*, 2012, pp. 1–10.

[21] S. Guha and P. Basu, "Applications of directed percolation in wireless networks," http://www.academia.edu/2840955/Applications_of_Directed_Percolation_in_Wireless_Networks, 2012, raytheon BBN Technologies.

[22] R. Parshani, M. Dickison, R. Cohen, H. E. Stanley, and S. Havlin, "Dynamic networks and directed percolation," *A Letters Journal Exploring The Frontiers of Physics*, 2010.

[23] Z. Kong and E. M. Yeh, "Connectivity, percolation, and information dissemination in large-scale wireless networks with dynamic links," http:/arxiv.org/abs/0902.4449, 2009, cornell University Library.

[24] L. Zhang, L. Cai, and J. Pan, "Connectivity in two-dimensional lattice networks," in *INFOCOM, 2013 Proceedings IEEE*, April 2013, pp. 2814–2822.

[25] A. Stauffer, "Space time percolation and detection by mobile nodes," Microsoft Research, Tech. Rep., 2012, http://arxiv.org/abs/1108.6322.

[26] Y. Peres, A. Sinclair, P. Sousi, and A. Stauffer, "Mobile geometric graphs: Detection, coverage and percolation," in *the Twenty-Second Annual ACM-SIAM Symposium on Discrete Algorithms*, 2011, pp. 412–428.

[27] E. H. J. Virtamo, P. Lassil, and J. Ott, "Continuum percolation threshold for permeable aligned cylinders and opportunistic networking," *IEEE COMMUNICATIONS LETTERS*, vol. 16, no. 7, 2012.

[28] E. Hyytia and J. Ott, "Criticality of large delay tolerant networks via directed continuum percolation in space time," in *INFOCOM2013*, April 2013, pp. 320–324.

[29] Y. Xu and X. Wang, "Fundamental lower bound for node buffer size in intermittently connected wireless networks," in *INFOCOM2011*, April 2011, pp. 972–980.

[30] D. C. Galant, "Queuing theory models for computer networks," Ames Research Center, Tech. Rep. 101056, 1989, NASA Technical Memorandum.

[31] P. Zeephongsekul, A. Bedford, J. Broberg, P. Dimopoulos, and Z. Tari, "Queuing theory applications to communication systems: Control of traffic flows and load balancing," *Springer Handbook of Engineering Statistics*, 2006.

[32] B. Chandrasekaran, "Survey of network traffic models," http://www.cse.wustl.edu/~jain/cse567-06/ftp/traffic_models3.pdf, accessed in March 2015.

[33] T. M. Chen, *The Handbook of Computer Networks*. Hossein Bidgoli (ed.), Wiley, 2007, ch. Network Traffic Modeling.