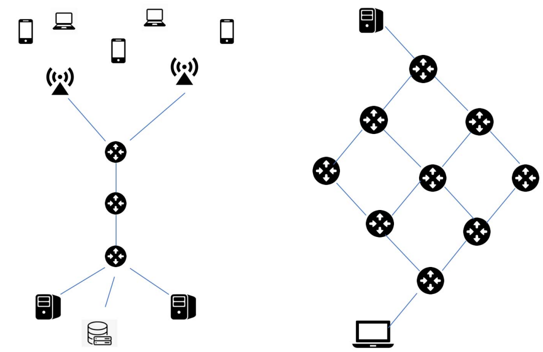
2021 IEEE 18th International Conference on Mobile Ad Hoc and Smart Systems (MASS) | 978-1-6654-4935-9/21/$31.00 ©2021 IEEE | DOI: 10.1109/MASS52906.2021.00035

# Mesh Network Reliability Analysis for Ultra-Reliable Low-Latency Services

Junhao Hu, Lin Cai, Jianping Pan
University of Victoria

***Abstract*—In a mesh network, to ensure high reliability and low latency, we can explore path diversity. In other words, a packet can be transmitted using all active links in a network to reach the destination. Here, a critical, difficult issue is to calculate the end-to-end reliability of a mesh network, given the reliability of each active link. In this paper, we derive the mesh network reliability with a new approach, which is of lower computational cost and more scalable than the state-of-the-art. Based on a Markov model, the closed-form network reliability as a polynomial expression of link reliability is obtained using the Hop-State Algorithm (HSA). Furthermore, we propose two metrics to assist in selecting the links in a network for routing to ensure performance while reducing link cost. From the analysis and simulation evaluations, exploring path diversity can effectively support Ultra-Reliable Low-Latency (URLL) services.**



(a) Tree topology (b) $2 \times 2$ grid

Fig. 1. Tree vs. mesh networks.

## I. Introduction

Many emerging applications require Ultra-Reliable and Low-Latency (URLL) services, and they will be a driving force for the future growth of communication networks. First, for many Internet-of-Things applications, real-time sensing and control information needs to be exchanged among machines or algorithms which are less intelligent and not error-resilient, demanding high-reliability and in-time/on-time services. Here are the delay requirements for such real-time control applications: industry 4.0 (a few ns to a few ms); in cellular systems, Common Public Radio Interface ($\leq$ 100 $\mu$s), and inter-site coordinated multipoint (Co-MP) ($\leq$ 250 $\mu$s); smart grid ($<$ 5 ms); vehicular communications for autonomous driving (a few ms). Second, applications such as high-frequency trading need to compete with each other at the ms level to profit from the high correlation of financial data distributed globally. For these applications, reducing a milli-second in delay can lead to million-dollar profits.

Compared to the previous delay-sensitive multimedia applications such as voice/video over IP, the above applications not only have a more stringent delay requirement but also cannot tolerate packet loss. The combination of high reliability and low latency brings tremendous challenges to packet-switching communication networks. It also attracts extensive research and standardization activities, covering both the backbone, access networks, and specialized/dedicated network systems. For the Internet community, IETF Deterministic Networking (DetNet) and IEEE 802.1 Time-Sensitive Networking (TSN) working groups closely cooperate with each other, aiming to support end-to-end service guarantee [1], [2]. In cellular systems, the 3GPP standard organization investigates how to provide URLLC services with the 5G New Radio (NR) radio access networks, 5G core networks, and 5G fronthaul/backhaul [3]. For data center networks, how to apply priority flow control to ensure URLL services has also been heavily investigated.

Given the traffic statistics and communication link properties, how to reserve/allocate resources in a single link to ensure latency/reliability has been extensively investigated. In a tree-topology network shown in Fig. 1(a), there is a single path for each flow, so we can ensure the URLL services of each link along the path to ensure the end-to-end performance. However, this approach is ineffective for mesh networks. In a $2 \times 2$ mesh network shown in Fig. 1(b), there are 6 "shortest" paths (in terms of hop count) between the source and destination pair. In a $4 \times 4$ and $9 \times 9$ one, the number of "shortest" paths is 70 (eight choose four) and $48,620$ (eighteen choose nine), respectively. Given the many possible paths, a key issue not fully explored yet is to deliver the message over multiple paths to ensure low latency and high reliability simultaneously.

If delivering a message over a mesh network using all possible paths, the end-to-end delay is the minimum of all paths. Given each link has only a certain probability to successfully deliver a packet within a bounded delay (e.g., without involving link-layer retransmission), the end-to-end reliability over the network is a key issue. The fundamental of this problem is the network reliability problem. In other words, when a message is geo-casted towards the destination (i.e., the message will be forwarded toward the destination hop by hop, so long as the next-hop neighbor is closer to the destination), given each hop has a probability of $p$ to successfully transmit the packet, what is the probability that it can reach the destination.

978-1-6654-4935-9/21/$31.00 ©2021 IEEE
DOI 10.1109/MASS52906.2021.00035

Note that we prefer closed-form solutions to guarantee quality of service (QoS) and optimize network planning and operation. Since there are many dependent paths in a mesh network, the calculation of network reliability is prohibitively complex if using the principle of inclusion and exclusion (PIE) probability theory, i.e., $O(2^{\binom{2n}{n}})$ for an $n \times n$ lattice grid network. Note that $2^{\binom{2n}{n}} \approx 7.2e75$ for $n = 5$, which is above the current computation capacity. To tackle the problem, a recursive approach was devised, which can obtain the reliability of two-dimensional lattice networks up to the size of ten by ten [4]. This work published in 2014 is also the best-known result. How to obtain a closed-form solution for a larger network remains open.

In this paper, we propose a new, more efficient approach that can calculate a two-dimensional lattice of the size of $15 \times 15$, and it can handle both regular and irregular topology networks. First, we pre-process a network to a directed graph with unambiguous hop count (from the source) for each vertex, so all vertexes with the same hop count can be grouped into a set. Then, we construct a Markov process to decouple the end-to-end reliability problem by calculating the reliability between any neighbor sets, which results in much lower complexity compared to the existing state-of-the-art solution. The proposed method is named Hop-State Algorithm (HSA), which can be applied to obtain the reliability of networks with both regular and irregular topologies. Furthermore, to reduce link cost without violating the service guarantee, two metrics are proposed to assist a trimming process to select a part of the network to deliver the message under the reliability constraint.

The rest of the paper is organized as follows. Sec. II gives the related work. The new HSA approach to derive the end-to-end network reliability is given in Sec. III. Sec. IV applies the HSA approach in two-dimensional lattice networks to compare its performance and validate its correctness with the existing work that can handle 2-D lattice networks. How to select links considering the tradeoff between cost and connectivity/reliability is also given. Sec. V presents the numerical validation, followed by the concluding remarks and further research issues in Sec. VI.

## II. Related Work

Network reliability, also named network connectivity in the literature, is a fundamental network performance metric, and it has been investigated for different types of networks [5]. In ad hoc networks, due to the random location of the nodes in ad hoc networks, the multi-hop forwarding has to be characterized probabilistically [6]. The approximate formula is presented for the probability of network connectivity [8]. With the assumption of a uniform distribution of nodes, the exact analytical expressions of the probability of connectivity were obtained in one-dimensional networks but only approximate bounds for the connectivity in two-dimensional networks [7]. The connectivity of message propagation in the two-dimensional ladder case was derived [9].

On the other hand, geometric algorithms have been used in wireless sensor networks [10]. Different techniques based on stochastic geometry and the theory of random geometric graphs (including point process theory, percolation theory, and probabilistic combinatorics) have led to different results on connectivity, capacity, outage probability, and other fundamental limits of wireless networks [11], [12]. Due to the physical space covered by nodes, networks show a unique geometric characteristic such as triangles, rectangles, and hexagons [13]–[15]. Although hexagons and rhombuses are also used, the square lattices network is most widely used in city scenarios. In addition, extra nodes can be distributed to improve connectivity by exploring geometric structures of sensor network.

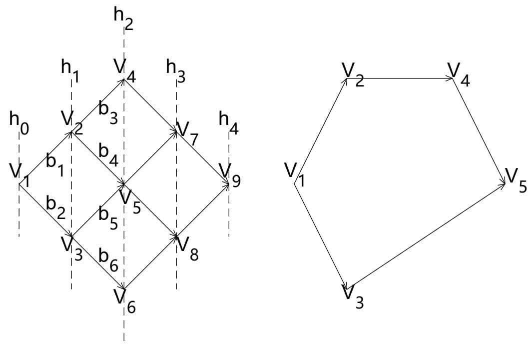
Another approach to compute the probability of network connectivity is percolation theory [16]. Given a destination, messages flood to the certain directions in geographical forwarding which is similar to a directed percolation process [19]. For messages with a given destination, or vehicles traveling in certain directions, geographical forwarding is often deployed to minimize the network overhead due to flooding [9]. Thus directed percolation becomes an often-used model in such scenarios, and most existing work applies the results from isotropic or directed percolation on square lattices. However, the directed percolation problem only cares about the existence of a giant component, while network connectivity has to determine the exact connectivity to each vertex, which is more relevant to network performance [17], [18].

The most related existing work is [4]. Considering a 2D lattice topology, a recursive decomposition approach has been developed by extending the 2D ladder connectivity to establish the analytical expression. Instead of splitting up a network into parts, this approach decomposes a lattice in one path and the union of all other paths. Although it can determine the network connectivity, the maximum it can calculate realistically is $10 \times 10$ lattices. We are motivated to develop a more efficient analytical approach that can handle networks with a larger size and more general topology.

## III. Mesh Network Reliability Analysis

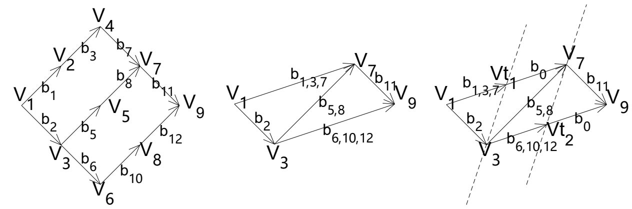
In this paper, we assume a message from the source node can take all possible directed paths in a mesh network to reach the destination. Assume this message can be carried in a single packet in any link, and message/packet is used interchangeably below. This message delivery process mimics the filtering of fluids through porous materials along a given direction, due to the effect of gravity. As shown in Fig. 2(a), at the source and all intermediate nodes, the message will be duplicated and forwarded to the neighboring nodes using directed links. To avoid the blindly flooding cost, we assume that each message will be delivered over each link once. In other words, if a router receives a duplicated message, it will discard the message. Then, the link cost to deliver the message depends on the number of active links in the network.

This network can be viewed as a directed graph where vertexes are network nodes (e.g., routers), and edges are the links between the vertexes. There is no loop given the directed routing in this directed graph. At any moment, each link or




(a) (b)

Fig. 2. Hop count ambiguity: (a) unambiguous; (b) ambiguous.




(a) (b) (c)

Fig. 3. Pre-process network graph: (a) simplification; (b) converting to unambiguous network as shown in (c).

edge (used interchangeably) can reliably deliver a packet with a probability. This probability is defined as link reliability. The physical meaning of the link reliability can be defined in different applications, e.g., the probability that a packet can be successfully delivered over the link within a delay bound. We also assume that the volume of traffic for URLL services is limited so network congestion losses are ignored.

To calculate the end-to-end reliability in a directed graph, we first define hop count of each vertex or node (used interchangeably), and pre-process the network graph based on hop count. Then, we apply a Markov process to calculate the network reliability.

### A. *Unambiguous hop count*

**Node with unambiguous hop count:** Similar to topological sort, if every directed path from the source to reach a vertex has the same number of hops, the vertex has an *unambiguous* hop count. For example, in Fig. 2(a), $V_1$ is the source, and there are two paths to reach vertex $V_5$, i.e., $V_1 \to V_2 \to V_5$ and $V_1 \to V_3 \to V_5$. The hop count of both paths is two, so $V_5$ has an unambiguous hop count of two.

**Network with unambiguous hop count:** If all vertexes in a network have an unambiguous hop count, we define the network as an unambiguous hop count network; otherwise, it is an ambiguous one. For example, in Fig. 2(a), the source vertex $V_1$ has the hop count of 0, the hop counts of vertex $V_2$ and $V_3$ are both 1, those of vertex $V_4$, $V_5$, and $V_6$ are 3, and so on. Thus, this network has unambiguous hop count. In Fig. 2(b), the hop count to reach vertex $V_5$ can be either 2 or 3 when taking different paths, so it is an ambiguous hop-count network.

In an unambiguous hop-count network, the vertexes can be grouped into sets $G_h$ where $h$ is the hop count to reach the vertexes from the source. For instance, in Fig. 2(a), the dotted line $h_i$ passes all vertexes in group $G_i$.

### B. *Pre-process network graph*

We take two steps below to pre-process the network graph.

First, to simplify the analysis, we can convert a network to an equivalent one with a simpler topology by combining chained links passing through the nodes with a single link in and a single link out. For example, as shown in Fig. 3(a), $V_2$ and $V_4$ each has a single link in and a single link out, so the links in and out from them, $b_1$, $b_3$, and $b_7$ can be combined into one, named $b_{1,3,7}$ as shown in Fig. 3(b). The reliability probability of $b_{1,3,7}$ equals the multiplication of the reliability probabilities of $b_1$, $b_3$, and $b_7$.

Second, an ambiguous hop-count network can be converted to an unambiguous one as follows. For a vertex, if it can be reached by several paths with different hop counts, virtual links (each has the reliability of 1) should be inserted into the shorter paths until all paths have the same hop count to reach the vertex. For instance, the network in Fig. 3(b) is an ambiguous hop-count network. Two paths to reach vertex $V_7$ are $V_1 \to V_7$ and $V_1 \to V_3 \to V_7$. Thus, we add a virtual link $b_0$ into the first path by adding a virtual vertex $V_{t_1}$, as shown in Fig. 3(c). Similarly, we add a virtual vertex $V_{t_2}$ in the bottom link between $V_3$ and $V_9$ to make $V_9$ unambiguous. Then, the network becomes an unambiguous one as shown in Fig. 3(c).

With the two-step pre-processing, we can obtain the simplified unambiguous network with the same reliability as the original one. In the following, we can focus on the reliability analysis of unambiguous networks.

### C. *Markov chain model*

Next, we build a Markov chain sequenced by hop count to assist the reliability analysis. We first define the states of this Markov chain, and then derive the state transition probabilities.

*1) Label states for each hop set:* For an unambiguous network, we can partition all the vertexes into sets, $G_h$, based on their hop count $h$ from the source.

For a given set $G_h$, considering whether or not each vertex in $G_h$ received the message, there are $2^{r_h}$ different states, where $r_h$ is the number of vertexes in $G_h$. For example, the vertex set $G_1$ in Fig. 2(a) has 2 vertexes $V_2$ and $V_3$. Thus, one of the following three states should be reached to ensure end-to-end reliability: a) $V_3$ receives the message but $V_2$ not, b) $V_2$ receives the message but $V_3$ not, and c) both of them receive the message.

Obviously, if a message can reach its destination, the message must successfully reach at least one vertex in each vertex set $G_i$, where $i$ is smaller than or equal to the hop count of the destination. The state of $G_i$ depends on the states of

$G_{i-1}$ and the reliability of links between the two sets only, and we do not need to consider how the state of $G_{i-1}$ is reached. This is in fact an important Markov property that can help us to simplify the end-to-end reliability analysis substantially.

To denote the state of each set, we first use a binary number to label whether or not a vertex receives the message by 1 and 0, respectively. Then the sequence of the binary labels of all vertexes in a set can be converted to its state label. For instance, the three states of $G_1$ in the above example can be labelled by state $01_2 = 1$, $10_2 = 2$, and $11_2 = 3$, respectively.

Denote all states for the message reaching at least one vertex in set $G_h$ as $S_k^h$, where $k$ is the state label. We have $1 \leq k \leq 2^{r_h} - 1$ where $r_h$ is the number of vertexes in $G_h$. Using this notation, $S_5^2$ implies that in $G_2$, the first and third vertexes receive the message, and the rest do not.

*2) State transition probabilities:* We define $P(S_k^h)$ as the probability of reaching state $S_k^h$ and $P(b_n)$ as the probability of link $b_n$ is reliable.

Considering the links between two neighbour vertex sets, the state transition probability from $S_k^h$ to $S_{k'}^{h+1}$ is denoted by $H_{k,k'}^h$.

For instance, as shown in Fig. 2(a), for state $S_k^2$, given four links $b_3$, $b_4$, $b_5$, and $b_6$, and the states in $G_1$, we can obtain the states $S_k^2$ ($1 \leq k \leq 2^{r_h} - 1$) in $G_2$ as follows.

For $S_1^1$, $V_2$ is connected to the source $V_1$, so we need to consider its outgoing links $b_3$ and $b_4$. If $b_3$ is reliable but $b_4$ is not, $V_4$ is reached, and thus state $S_1^2$ can be reached from $S_1^1$. We have $H_{1,1}^1 = P(b_3)(1 - P(b_4))$.

For $S_2^1$, only $V_3$ received the message and it cannot deliver the message to $V_4$.

For $S_3^1$, $V_2$ and $V_3$ both received the message, so $b_3$, $b_4$, $b_5$, and $b_6$ should be considered. When $b_3$ is reliable but others are not, only $V_4$ received message and state $S_1^2$ is reached. We have $H_{3,1}^1 = P(b_3)(1 - P(b_4))(1 - P(b_5))(1 - P(b_6))$.

Then, $P(S_1^2)$ is given by

$$P(S_1^2) = P(S_1^1) \cdot H_{1,1}^1 + P(S_3^1) \cdot H_{3,1}^1. \tag{1}$$

Similarly, all probabilities of states in $G_3$ can be given, as shown in Fig. 4(a).

Using the Markov property, given the states in the $G_h$, and the connectivity of links between $G_h$ and $G_{h+1}$, the probability to reach states in $G_{h+1}$ is given by

$$P(S_{k'}^{h+1}) = \sum_{k=1}^{2^{r_h}-1} P(S_k^h) \cdot H_{k,k'}^h. \tag{2}$$

Next, we elaborate how to obtain the state transition probability $H_{k,k'}^h$. Using the binary sequence of state $k$, we define vertex set $A$ includes those who have received the message and can send it to their next hop as

$$A = \{Va_1, Va_2, Va_3, ..., Va_\alpha\}, \tag{3}$$

where $\alpha$ is the number of vertexes in $A$.

For instance, in Fig. 4(b), to reach $H_{7,10}^h$, $7 = 111_2$ means that $V_1^h, V_2^h, V_3^h$ can send messages to vertexes in $G_{h+1}$. Given the links from $G_h$ to $G_{h+1}$, we have $E = \{Ve_1, Ve_2, Ve_3, ..., Ve_\epsilon\}$ who may receive the message, where $\epsilon$ is the number of vertexes in $E$.

In the above example, given $\alpha = 3$, $A = \{V_1^h, V_2^h, V_3^h\}$, and thus $E = \{V_1^{h+1}, V_2^{h+1}, V_3^{h+1}, V_4^{h+1}\}$ with $\epsilon = 4$. Using the binary number of $k'$, we have vertexes $C = \{Vc_1, Vc_2, Vc_3, ..., Vc_\delta\}$ ($\delta > 0$) who may receive the message corresponding to this state. Obviously, $C$ should be a subset of $E$.

We define set $I = \{Vl_1, Vl_2, Vl_3, ..., Vl_\iota\}$ ($\iota \geq 0$), which contains the vertexes in $E$ but not in $C$, i.e., $E = C \cup I$ and $\emptyset = C \cap I$. In the example shown in Fig. 4(b), $10 = 1010_2$, so $C = \{V_2^{h+1}, V_4^{h+1}\}$ and $\delta = 2$. Correspondingly, $I = \{V_1^{h+1}, V_3^{h+1}\}$ and $\iota = 2$.

From the definitions, vertexes in $C$ receive the message, and vertexes in $I$ do not receive the message. Define $P(C_i)$ as the probability of at least one message is received by vertex $Vc_i$, and $P(I_j)$ as the probability of vertex $Vl_j$ receiving no message. We have

$$H_{k,k'}^h = \Pi_{i=1}^{\delta} P(C_i) \cdot \Pi_{j=1}^{\iota} P(I_j). \tag{4}$$

For set $A$ defined in (3), define $\{Va_{c1}, Va_{c2}, ..., Va_{cp_i}\}$ as the vertexes who are linked with $Vc_i$, and $p_i$ is the number of these links, as shown in Fig. 4 (c). For the example shown in Fig. 4(b), $\{V_1^h, V_2^h\}$ ($p_2 = 2$) are linked with $V_2^{h+1}$ and the two links are $b_1$ and $b_2$. [1] Following the same principles, for $V_4^{h+1}$, it is linked with $\{V_3^h\}$ ($p_4 = 1$) through link $b_6$. These links are denoted as $bc_{i,\kappa}$ ($1 \leq \kappa \leq p_i$), and their link reliability probability is $P(bc_{i,\kappa})$, respectively. For $V_{C_i}$ to be reached, at least one link of $bc_{i,\kappa}$ should be reliable, so we have

$$P(C_i) = 1 - \Pi_{\kappa=1}^{p_i}(1 - P(bc_{i,\kappa})). \tag{5}$$

Following the approach, using the example in Fig. 4(b), $C = \{V_2^{h+1}, V_4^{h+1}\}$ has two vertexes. The first vertex $V_2^{h+1}$ can be reached through links $b_2$ and $b_3$, so $P(C_1) = 1 - (1 - P(b_2))(1 - P(b_3))$. The other vertex $V_4^{h+1}$ can only receive the message from link $b_6$ because vertex $V_4^h$ is not connected, so $P(C_2) = P(b_6)$.

As shown in Fig. 4(d), similarly, vertex $Vl_j$ can receive messages from $\{Va_{l1}, Va_{l2}, ..., Va_{lq_j}\}$, where $q_j$ is the number of these links. The probability of link $bl_{j,\kappa}(1 \leq \kappa \leq q_j)$ is reliable is denoted by $P(bl_{j,\kappa})$. For vertexes in $I$ which are not connected, we have

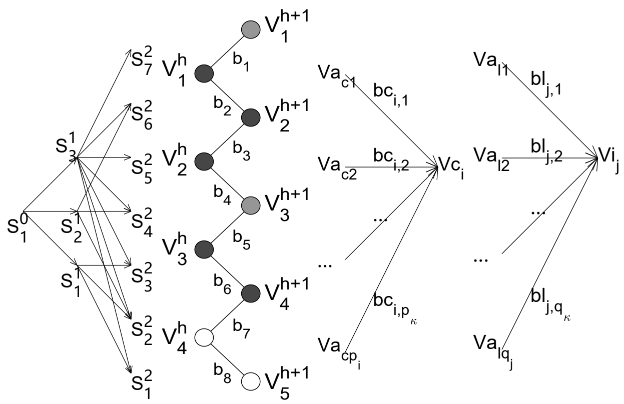
$$P(I_j) = \Pi_{\kappa=1}^{q_j}(1 - P(bl_{j,\kappa})). \tag{6}$$

In the example in Fig. 4(b), group $I$ has two vertexes, $V_1^{h+1}$ and $V_3^{h+1}$. If $V_1^{h+1}$ is not connected, link $b_1$ is not reliable, and $P(I_1) = 1 - P(b_1)$. If $V_3^{h+1}$ is not connected, links $b_4$ and $b_5$ are both unreliable, we have $P(I_2) = (1 - P(b_4))(1 - P(b_5))$.

Then the state transition probability $H_{k,k'}^h$ can be expressed as

$$\begin{aligned} H_{k,k'}^h = &\Pi_{i=1}^{\delta}[1 - \Pi_{j=1}^{p^i}(1 - P(bc_{i,j}))] \\ &\times \Pi_{i'=1}^{\iota}\Pi_{j'=1}^{q^{i'}}(1 - P(bl_{i',j'})). \end{aligned} \tag{7}$$

[1]Note that $p_i$ is equal to or less than 2 in lattice networks, but in other networks, e.g., in triangle networks, $p_i$ can be larger.



(a) (b) (c) (d)

Fig. 4. State transitions

The state transition probability of the example in Fig. 4(b) is

$$\begin{aligned} H^h_{7,10} =& P(C_1)P(C_2)P(I_1)P(I_2) \\ =& [1-(1-P(b_2))(1-P(b_3))]P(b_6) \\ & (1-P(b_1))(1-P(b_4))(1-P(b_5)). \end{aligned}$$

### D. Hop-State Algorithm (HSA)

Based on the above analysis, we devise the Hop-State Algorithm (HSA) shown in Algorithm 1 to calculate the end-to-end reliability of mesh networks which are hop-count unambiguous. In the Algorithm, the termination condition is when the final hop reaches the destination. When we calculate for a network with one destination, the final hop set has one vertex only, and the corresponding connected state probability is also the end-to-end reliability.

**Algorithm 1** Hop-State Algorithm (HSA)

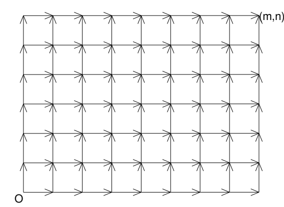
$h = 0$ # hop count number
**for** not reach the destination **do**
  $k' = 0$ # state number
  **for** $k' < 2^{r_{h+1}}$ **do**
    $k' = k' + 1$
    $P(S^{h+1}_{k'}) = 0$
    $k = 0$ # state number
    **for** $k < 2^{r_h}$ **do**
      $k = k + 1$
      **if** The corresponding $C$ belongs to $E$ **then**
        # add the probability contributing to state $k'$ in hop $h+1$ from state $k$ in hop $h$
        $P(S^{h+1}_{k'}) = P(S^{h+1}_{k'}) + P(S^h_k) \cdot H^h_{k,k'}$
      **end if**
    **end for**
  **end for**
  $h = h + 1$
**end for**

## IV. Two-Dimensional Lattice Network Reliability

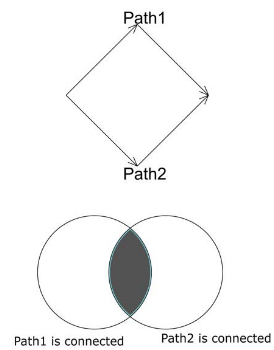
Different network scenarios correspond to different topology. As lattice networks have a wide range of applications, we here focus on analyzing 2-dimensional (2D) lattice networks in this section. We first present the complexity analysis of the proposed method for 2D regular lattice networks and compare it with the state-of-the-art. We then analyze the complexity and efficiency in dealing with other irregular lattice networks.

### A. Regular 2D lattice networks

In an $m \times n$ lattice network for directional percolation routing as shown in Fig. 5(a), each path from the source (at the origin) to the destination (at $(m, n)$) has $m$ west-east links and $n$ south-north links. Given the combinations of an arbitrary sequence of $m$ west-east links and $n$ south-north links, the number of the paths between the source and destination equals $\binom{m+n}{m}$.




(a) $m \times n$ lattice network




(b) PIE principle

Fig. 5. Lattice networks

To reach the destination successfully, at least one of the paths should be reliable, i.e., all links along this path should be reliable. Using the Principle of Inclusion-Exclusion (PIE), to calculate the network reliability, we need to sum the probabilities of each path is reliable, minus the probabilities that any pair of paths is reliable, plus the probabilities that any triple of paths is reliable, and so on. For instance, a $1 \times 1$ lattice network has two paths as shown in Fig. 5(b). The reliability of the network is the sum of probabilities of two paths minus the probabilities of two paths are both reliable, as shown in the gray area of the Venn diagram. The complexity is dominated by the number of path combinations, i.e., $\sum_{i=1}^{\binom{n+m}{n}} \binom{\binom{n+m}{n}}{i} = 2^{\binom{n+m}{n}} - 1$. Thus, the total complexity of the PIE approach is $O(2^{\binom{n+m}{n}})$. For an $n \times n$ lattice, the PIE complexity is asymptotically $O(2^{\frac{4^n}{\sqrt{\pi n}}})$, extremely high.

In [4], a recursive approach (REC) to calculate the connectivity (reliability) of lattice networks was developed. With this approach, the network is decomposed step by step, and the connectivity of a network (whether with the lattice topology or a decomposed tower topology) can be calculated based on the decomposed network connectivity. For an $m \times n$ lattice network, the complexity of the REC algorithm in [4] is given by $\sum_{i=1}^{n-1} \binom{m+i-1}{i} \cdot [1 + (m-1)(n-i)]$. Using the big-$O$ notation, the complexity of this algorithm for an $n \times n$ network is $O\left(n^2 \frac{2n!}{n!(n+1)!}\right)$, which is asymptotically $O(\sqrt{n}4^n)$, much lower than that of PIE.

In our proposed approach, decoupling the network vertexes into sets with the same hop count, links are also divided into $m+n$ groups, where the links in each group connecting two neighboring vertex sets. We then can consider cases one by one in an $m \times n$ $(m \geq n)$ lattice. The complexity of the proposed hop-state algorithm is

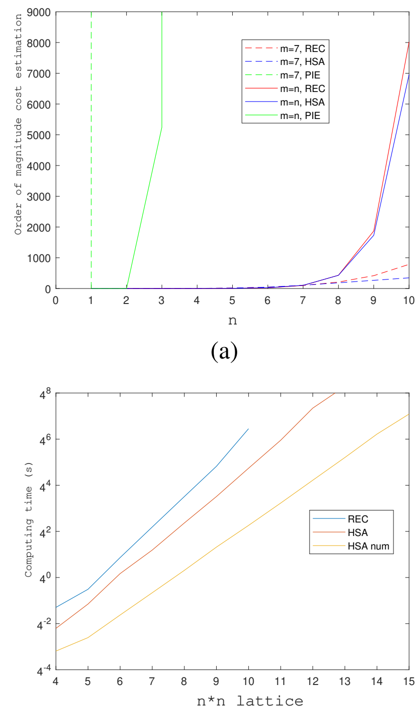
$$2 \sum_{i=1}^{n-1} \left[(2^i - 1)(2^{i+1} - 1)\right] + (m-n) \cdot (2^n - 1)^2. \qquad (8)$$

In other words, the complexity of the proposed algorithm is $O(4^n)$ for a regular $n \times n$ lattice. Comparing to the state-of-the-art in [4], we achieve a $\sqrt{n}$ reduction. Furthermore, the space complexity of the recursive approach in [4] is much higher, resulting in a much slower speed overall when compared with our solution.

We use different approaches to calculate the closed-form network reliability for homogeneous $m \times n$ lattice networks where each link has a reliability of $p$, and the end-to-end reliability is a polynomial function of $p$. For instance, the reliability of $2 \times 2$ network is $p^4 - 6p^6 - 4p^7 + 2p^8 + 4p^9 + 2p^{10} - 4p^{11} + p^{12}$. The coefficients of the polynomial functions can be obtained by different approaches and we can compare them to validate the correctness of the new approach. First of all, we have compared the results of lattice networks up-to $10 \times 10$ using HSA and REC, and they give the same results. This validates the correctness of the two approaches.

Second, the detailed complexity performance comparisons for regular $m \times n$ lattice networks are given in Fig. 6(a). Here, the $x$-axis represents $n$, the size of one dimension of the lattice. The dashed lines are for the cases the other dimension size $m = 7$, the solid lines are for the cases $m = n$. From the figure, the complexity (super-)exponentially grows with $\min\{m, n\}$ for all approaches. The green line is the PIE approach which has explosive growth and it reaches our computation capacity limit for $4 \times 4$ lattice networks. For the other two approaches, they can handle larger networks so long as $\min\{m, n\}$ is reasonably small.

As shown in Fig. 6(b), comparing to [4], the proposed HSA algorithm has substantially lower running time and is more scalable. Note that the coefficient of the reliability expression becomes extremely large when the size of the network is large. For example, if $10 \times 10$, the largest coefficient is $5.0537e36$ and if $18 \times 18$, the largest coefficient is $2.8092e125$. Due to the coefficient sensitivity, these coefficients should be




(a)

(b)

Fig. 6. Complexity comparison, for lattice network

exact integers. Thus, it costs a lot of time for Big-number calculation. The blue line is HSA for calculating analytical expressions (the coefficients of the polynomial expression of end-to-end reliability) and the red line is with the REC approach.

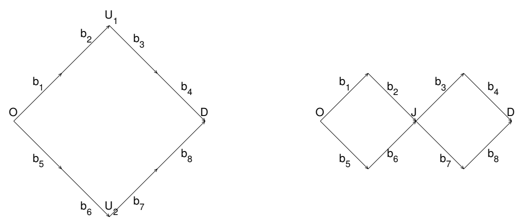
On the other hand, if we know the numerical link probability $p$ of each link, the numerical calculation of reliability is much faster. Orange line in Fig. 6 (b) gives the running time for numerical calculation of reliability using the proposed HSA.

### *B. Irregular lattice networks*

In realistic network scenarios, due to the physical limits or given the trade-off of link cost and reliability/delay performance, not all links in a lattice network are used for data transmission. When we use a part of the network links for routing, these active links can be viewed as an irregular lattice network. An important, difficult question is how to select the links for routing to ensure the performance while minimizing the cost.

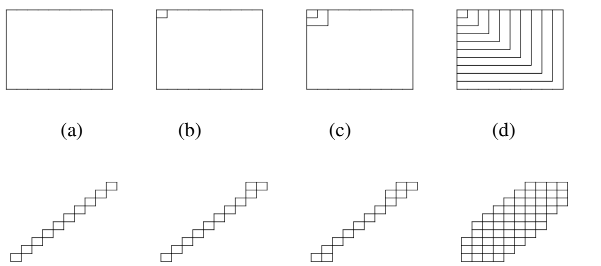
*1) Joint vs. disjoint vertexes:* If two paths passing the same vertices other than the source and destination, we name the vertices as joint vertices. In Fig. 7, the source vertex is $O$ and the destination vertex is $D$. The paths in Fig. 7(a) do not pass any common vertex other than $O$ and $D$, but paths in Fig. 7(b) are joint at vertex $J$.

Given the same number of links, if there are more joint vertexes, the total number of paths will be higher. For example, Figs. 7(a) and (b) can be viewed as selecting 8 edges in a




(a) Disjoint vertex (b) Joint vertex

Fig. 7. Networks with vs. without joint vertexes



(a) (b) (c) (d)

(e) (f) (g) (h)

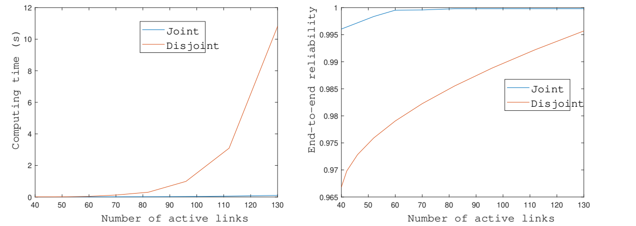
Fig. 8. Active links of disjoint (a-d) vs. joint network (d-h)

$2 \times 2$ lattice network for delivering messages from $O$ to $D$. The number of four-hop paths in Fig. 7(a) is two, and the path number in Fig. 7(b) is four, which results in a higher reliability according to the PIE principle.

Therefore, comparing the reliability of two same-sized networks with the same number of links, we generally prefer the one with more joint vertexes. It is difficult to quantify how joint paths affecting the overall reliability though. We define a simple metric, $Jd = \dfrac{A_b}{A_v}$, to describe the degree of joint vertexes in a network, where $A_b$ is the number of links and $A_v$ is the number of vertexes in the network. If using the same number of links, we prefer the network with a larger $Jd$, which may lead to a higher reliability. This can be used as a guideline when we have the freedom to choose a limited number of links to be activated in a network while maintaining high connectivity/reliability.

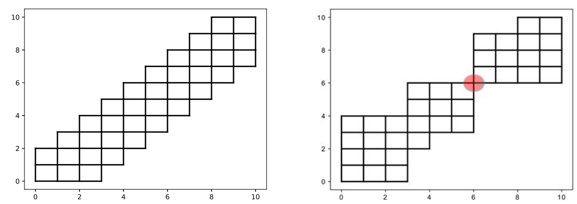
Here are some examples to demonstrate the above guideline. We create networks with few joint vertices that follow the style of Fig. 8(a), (b), (c), and (d). Then, we create joint vertices networks with the same active link number that follows the style of Fig. 8(e), (f), (g), and (h). It takes more time to calculate the reliability for disjoint networks than the joint networks, as shown in Fig. 9(a). Setting all link probabilities as 0.99, the end-to-end reliabilities are obtained. From the results, the networks with a larger $Jd$ have a higher reliability, as shown in Fig. 9(b), which supports the proposed guideline.

*2) Variance of the group size for each hop count:* In addition to $Jd$, another metric affects the reliability is the variance of set size of different hop counts. For instance, as shown in Fig. 10, both networks are joint networks but the reliability of the network in (a) is higher than the network




(a) Running time (b) Reliability

Fig. 9. Computing time and reliability of joint and disjoint networks



(a) Small variance (b) Large variance

Fig. 10. Networks with different variances in terms of the size of hop count set

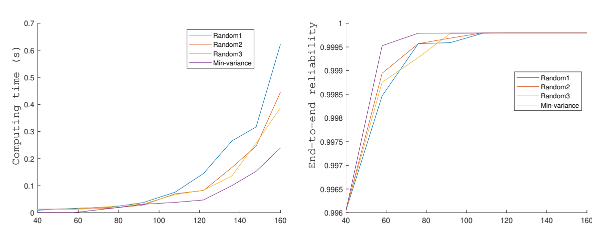
in (b) because of the bottleneck marked by the red circle. In order to quantify the degree of the bottleneck, we define the variance of the number of vertices in each hop set, $\sigma^2$, as another metric.

Using the variance formula,

$$\sigma^2 = \frac{\Sigma_1^{h_{\max}} (r_h - \bar{r})^2}{h_{\max} - 1}, \tag{9}$$

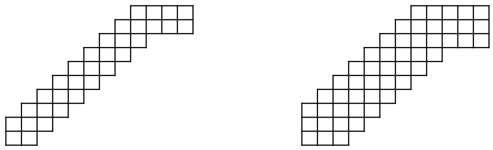
where $\bar{r}$ is the mean value of all hop sets, and $h_{\max}$ is the hop number of the destination. For example, for the networks in Fig. 10, the hop set sizes for $G_0$, ... $G_h$ are $1, 2, 3, 3, 3, 3, 3, 3 \ldots$ and $1, 2, 3, 3, 4, 3, 3 \ldots$, respectively, and their variances are approximately $0.4$ and $0.8$, respectively.

We run the simulation using a joint network following the style of Fig. 10(a) and some random joint networks following the style of Fig. 10(b). The simulation results are shown in Fig. 11. A larger variance often leads to some large size sets, resulting in longer computing time, as shown in Fig. 11(a). A smaller variance can generally lead to a higher reliability, as shown in Fig. 11(b). Thus, we can rely on the metric to select the links to achieve a higher reliability.



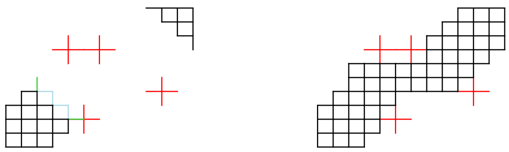

(a) Computing time (b) Reliability

Fig. 11. Simulation results, with different variance



(a) $n_t = 4$ (b) $n_t = 6$

Fig. 12. $12 \times 10$ lattice network, without hot spot



(a) Strategy of selecting link (b) Avoiding hot spot

Fig. 13. $12 \times 10$ lattice network, with hot spots

### C. Selecting links to ensure reliability

First, considering a two-dimensional $12 \times 10$ lattice network where the links are homogeneous, and there is no hot-spot in the network. As analyzed in Sec. IV-B, we prefer to choose the links leading to a large $Jd$ and small $\sigma^2$. We thus prefer the cases where the number of active links between each neighbour hop is the same except at both ends. Then, we can choose active links as shown in Fig. 12 (a). If the network needs a higher reliability, links between each neighbour hop can be added as shown in Fig. 12 (b).

Next, considering a more complicated case, where there are a few hot spots in the network, so a few links are congested which have a longer delay and a higher loss rate. When we select links for message delivery, we can avoid the busy links to achieve better load balancing and performance. We use an example to demonstrate how we select links. As shown in Fig. 13, we consider a $12 \times 10$ lattice network with hot spots. In the corners, the two links connected to the source and destination can be selected first, as shown in Fig. 13(a). In the middle, let the number of active links between neighbour hop be $n_t$. We start with one corner and choose the active links step by step. The end vertexes can be $n_t/2+1$ or $n_t/2$. When the end vertexes are $n_t/2$, we prefer the links in the middle unless they are busy links. In each step, as shown in Fig. 13(a), if all blue links are not busy and at least one green link is not busy, we can choose them; if one of the blue links or both green links are busy, we need to take a step back, and re-select the previous hop's links to probe for other better choices. With this recursive algorithm, the resulting link selection is shown in Fig. 13 (b), where the black links are selected for routing and the red links are the busy ones being avoided.

## V. Performance Evaluation

$p$ is the link reliability. If each packet is transmitted over a link once, the packet loss rate of a link equals $1 - p$.

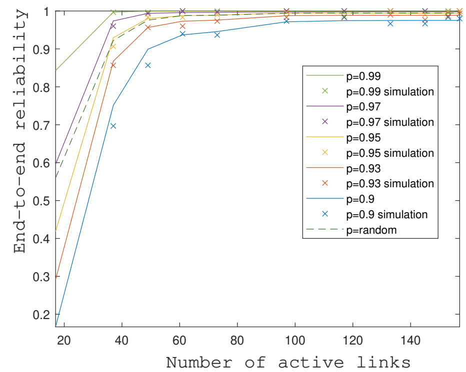



Fig. 14. Active link number vs. end-to-end reliability, $10 \times 7$ lattice

Given the lattice grid and the reliability of each link, the end-to-end reliability (or connectivity) can be calculated using the HSA algorithm in Sec. III. Based on the end-to-end reliability obtained using HSA, we can further select active links in the network to make a tradeoff of transmission cost and performance. Such a tradeoff can be observed from the performance evaluation below.

### A. End-to-end reliability

To investigate the tradeoff of link cost and performance, we can activate a part of links in a network. We use the number of active links instead of path number for performance comparison. Here, the number of active links is proportional to the communication cost.

We consider the $10 \times 7$ lattice network as an example, where there are in total 157 links in the network. To deliver a message from the source to the destination, the least number of active links is 17 when a single 17-hop path is used. We chose the optimal single-path to obtain the path-reliability and tuned $p$ for all links from 0.9 to 0.99, and we also investigated the scenario where the reliability of each link is randomly chosen between 0.9 and 0.99.

As shown in Fig. 14, using single-path routing, when $p = 0.99$, the end-to-end reliability is 0.85 only, not desirable for URLL services. With the increase of the number of active links, we can achieve a higher reliability. When $p = 0.99$, using 61 active links can achieve a close to one reliability, so we can turn off the rest 97 to save cost. In the situation of congestion, we can inactivate those links with higher traffic loads. When $p$ is as low as 0.9, we can still achieve above 0.98 end-to-end reliability when we activate about 100 links. Fig. 14 also shows that the simulation results match the analysis well. The dotted line shows the results with heterogeneous link reliability (average of 0.95). The performance is similar to the homogeneous link reliability case ($p = 0.95$) when there are more than 37 active links.

### B. Delay performance

Next, we simulated the shortest path routing and the one using all active links for routing to compare the delay performance. For shortest single-path routing, we chose the path

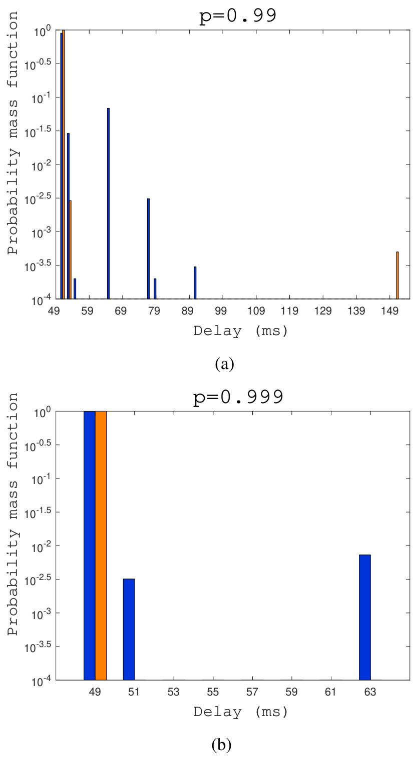



Fig. 15. Delay distributions (blue: single path; orange: all active links)

with the smallest delay between the source and destination, and the lost packet in each link will be retransmitted in the link layer. On the other hand, when using all active links to deliver the packet, we disabled the link-layer retransmission, and the packet will be retransmitted only if timeout happens (when zero copy is reached in the destination and no end-to-end ACK is received by the source). For each setting, $10,000$ packets were transmitted and their delays were measured.

The resulting delay distributions when $p = 0.99$ and $p = 0.999$ are shown in Fig. 15(a) and (b), respectively, where the blue bars are the delay distributions with single-path and the orange bars are those using all active links. From the figures, not only the average delay using all active links is lower, more importantly, using the single path with shortest-path routing, about $10\%$ of packets suffer a delay jitter more than 20 ms when $p = 0.99$ due to link layer retransmissions, while only $0.02\%$ of packets suffering delay outage when we explore the path diversity. When $p = 0.999$, about $1\%$ of packets suffering a delay jitter around $14$ ms using the shortest-path routing, and all packets can reach the destination within $50$ ms using all active links, which is more desirable for URLL services.

## VI. Conclusion

In a mesh network, to ensure high reliability and low latency, we can explore path diversity. In this paper, based on a Markov model, the closed-form network reliability as a polynomial expression of link reliability has been obtained using the proposed HSA, which is of lower computational complexity and more scalable than the state-of-the-art in the literature. Furthermore, we proposed two metrics that can be used for selecting the links in a network for routing to ensure performance while reducing link cost. From the analytical and simulation evaluation, exploring path diversity is an effective approach to ensure the reliability and delay performance for URLL services.

There are many open issues beckoning further investigation. For instance, how to further reduce the computation complexity to analyze the reliability of an even larger network, how to optimize the link selection process to minimize the link cost given the reliability and delay constraints, how to jointly optimize link parameters (such as transmission power, modulation and coding configuration) and the link selection in routing for URLL services, and how to handle network congestions if URLL traffic volume is high.

## References

[1] N. Finn, P. Thubert, B. Varga, and J. Farkas, “Deterministic networking architecture,” Internet Eng. Task Force, Fremont, CA, USA, Internet Draft draft-ietf-detnet-architecture-04, Oct. 2017.
[2] “IEEE 802.1 Time-Sensitive Networking Task Group,” http://www.ieee802.org.
[3] 3GPP 33.825, Study on the Security of Ultra-Reliable Low-Latency Communication (URLLC) for the 5G System (5GS), technical report, Release 16, 2019.
[4] L. Zhang, L. Cai, J. Pan, and F. Tong, “A new approach to the directed connectivity in two-dimensional lattice networks,” *IEEE Trans. Mobile Comput.,*vol. 13, no. 11, pp. 2458–2472, Nov. 2014.
[5] M. Desai and D. Manjunath, “On the connectivity in finite ad hoc networks,” *IEEE Comm Letters,* 6(10):437–439, 2002.
[6] J. He, L. Cai, J. Pan, and P. Cheng, “Delay analysis and routing for two-dimensional VANETs using carry-and-forward mechanism,” IEEE Trans. on Mobile Computing, vol. 16, no. 7, pp. 1830-1841, July 2017.
[7] O. Dousse, P. Thiran, and M. Hasler, “Connectivity in ad-hoc and hybrid networks,” in *Proc. IEEE Infocom’02,* pp. 1079–1088, 2002.
[8] A. Ghasemi and S. Nader-Esfahani, “Exact probability of connectivity in one-dimensional ad hoc wireless networks,” *IEEE Communications Letters,* vol. 10, no. 4, April 2006.
[9] Y. Zhuang, J. Pan, and L. Cai, “A probabilistic model for message propagation in two-dimensional vehicular ad-hoc networks,” *in Proc. ACM VANET’10,* pp. 31–40, 2010.
[10] J. Gao and L. Guibas, “Geometric algorithms for sensor networks,” *Philosophical Trans. of the Royal Society A,* 370(1958):27–51, 2012.
[11] M. Haenggi, J. G. Andrews, F. Baccelli, O. Dousse, and M. Franceschetti, “Stochastic geometry and random graphs for the analysis and design of wireless networks,” *IEEE J. Sel. Areas Commun.,* vol. 27, no. 7, pp. 1029–46, Sep. 2009.
[12] F. Tong, Y. Wan, L. Zheng, J. Pan, and L. Cai, “A probabilistic distance-based modeling and analysis for cellular networks with underlaying device-to-device communications,” IEEE Trans. on Wireless Communications, vol. 16, no. 1, pp. 451-463, Jan. 2017.
[13] Y. Zhuang and J. Pan, “Random distances associated with equilateral triangles,” *arXiv:*1207.1511, 2012.
[14] Y. Zhuang and J. Pan, “Random distances associated with rhombuses,” *arXiv:*1106.1257, 2011.
[15] Y. Zhuang and J. Pan, “Random distances associated with hexagons,” *arXiv:*1106.2200, 2011.
[16] I. Glauche, W. Krause, R. Sollacher, and M. Greiner, “Continuum percolation of wireless ad hoc communication networks,” *Physica A,*325(3):577–600, 2003.
[17] L. Zhang, L. Cai, and J. Pan, “Connectivity in two-dimensional lattice networks,” IEEE Infocom’13, Turin, Italy, April 2013.
[18] M. Ni, L. Zhang, J. Pan, L. Cai, H. Rutagemwa, L. Li, and T. Wei, “Connectivity in mobile tactical networks,” IEEE Globecom’14, Austin, TX, USA, Dec. 2014.
[19] J. Hu, L. Cai, C. Zhao, and J. Pan, ”Directed percolation routing for ultra-reliable and low-latency services in Low Earth Orbit (LEO) satellite networks,” IEEE 92nd VTC2020-Fall, New Network Architecture Powering Internet-of-Things Workshop, Victoria, Canada, Oct. 2020.