Centralized Versus Distributed Routing for Large-Scale Satellite Networks

Rudrapatna Vallabh Ramakanth and Eytan Modiano

***Abstract*—An important choice in the design of satellite networks is whether the routing decisions are made in a distributed manner onboard the satellite, or centrally on a ground-based controller. We study the tradeoff between centralized and distributed routing in large-scale satellite networks. In particular, we consider a centralized routing scheme that has access to global but delayed network state information and a distributed routing scheme that has access to local but real-time network state information. For both routing schemes, we analyze the throughput and delay performance of shortest-path algorithms in networks with and without buffers onboard the satellites. We show that distributed routing outperforms centralized routing when the rate of change of the network link states is comparable to the inherent propagation and transmission rate. In particular, we show that in highly dynamic networks without buffers, the distributed scheme achieves higher throughput than a centralized scheme. In networks with buffers, the distributed scheme achieves lower delays for the same throughput.**

I. Introduction

Recently, there has been a growing interest in deploying large-scale satellite networks for data communication. These networks usually comprise Low-Earth Orbit (LEO) constellations or Medium-Earth Orbit (MEO) constellations that are designed to be dense and regular, resembling a grid-like structure. For data communication purposes, designing a good routing strategy for such large-scale networks is a challenging task. The underlying routing scheme of a network plays a critical role in its performance. In the context of satellite networks, an important design choice is whether routing decisions should be made on-board the satellites, or on a ground-based central controller.

Most satellite operators today route packets through their networks using centralized routing approaches, as these methods provide tighter control and troubleshooting from the ground [1]. In addition, a centralized algorithm is easier to implement and requires less resources onboard satellites. However, in such a paradigm, the controller cannot react to instantaneous changes in the network state, such as congestion effects or link failures. This is because there is an innate unavoidable delay in propagating, aggregating, and processing network state information from satellites to the ground (and vice versa), which ranges in the order of tens of milliseconds or longer.

On the other hand, there is also a strong appeal to use distributed routing for satellite networks. A distributed routing strategy enables satellites to react to instantaneous network changes in their neighborhood while making routing decisions. Distributed routing policies for satellite networks that exploit the density and structure of these networks have been proposed [2]–[6]. However, such a scheme would incur additional infrastructure costs for route computation and buffering onboard the satellite. Even though it is conceptually possible for satellites to obtain global state information for routing, it is impractical because of limited processing and storage capabilities onboard the satellites, along with the additional overhead and delays required to transmit large amounts of network state information. This forces distributed algorithms to only use local state information for routing. Thus, a trade-off arises between centralized and distributed routing strategies.

To that end, we seek to answer the following question: **under what network conditions does a distributed routing scheme outperform a centralized routing scheme in terms of throughput and delay in regular, yet dynamic networks?**

Routing is a well-studied problem. In most terrestrial networks, operators use distance-vector routing protocols or link-state routing protocols, both of which are relatively static shortest-path routing protocols [7]. These are not suitable for satellite networks, where the links change state on the same time scale as the transmission of a packet across the network [2]. There exist methods to compute optimal routing policies using flow-based models [7]. However, these policies are very sensitive to traffic conditions and network parameters, rendering them impractical [7]. On the other hand, while routing for dynamic *ad hoc* wireless networks has been studied for a long time [8]–[11], they are not suitable for satellite networks, as they would disregard much of the regular structure. Of course, there exists a plethora of prior work on studying various routing strategies for satellite networks [1]–[6], [12]–[19]. However, to the best of our knowledge, existing work does not analytically characterize the conditions under which a distributed (or centralized) routing strategy yields better performance.

In this work, we consider routing packets across a grid-like network topology (see Fig. 1) with dynamic links that evolve in time according to a Markov process. Finding an optimal route in such a system requires the solution to a dynamic program (Markov Decision Process) with a state space that grows exponentially with the network size [20]. Since the solution to such a dynamic program is computationally prohibitive, we focus on practical shortest-path routing strategies that establish sufficient conditions on when each routing paradigm is better than the other. In particular, for the centralized case, we consider a Shortest Connected Path Routing policy that selects the shortest path from the source to the destination *only using links that are observed to be available for use*. For the distributed case, we consider a Greedy Routing policy,

ISBN 978-3-903176-73-7

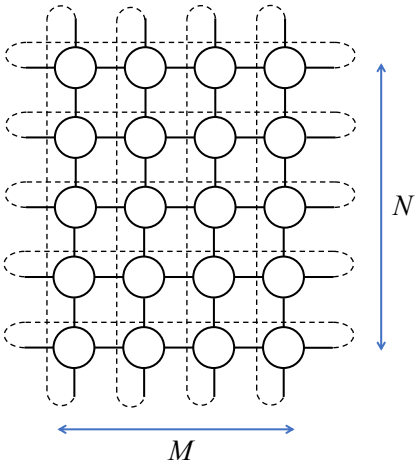


Fig. 1: Toroidal Mesh grid of size $N \times M$, where $N = 5$ and $M = 4$. Each node has degree 4.

that routes packets only along links that take it closer to the destination at every step. Note that both algorithms are types of "shortest-path algorithms" and are loop-free.

The main contribution of this work is in characterizing when distributed routing outperforms centralized routing. In particular, when the rate of change of link states is comparable to the inherent propagation and transmission rates of the network, we show that in networks without onboard satellite buffering, distributed routing generally achieves a higher throughput, and in networks with onboard buffering, distributed routing achieves the same throughput but a lower delay. However, in a network where the rate of change of link states is much lower than the propagation and transmission rate of the network, the centralized scheme has lower delay and higher throughput in networks with and without buffers respectively.

The rest of the paper is organized as follows. In Section II, we describe the model of the satellite network, and define the routing schemes and performance metrics in detail. Section III analyzes the performance of a practical centralized routing algorithm, and Section IV analyzes the performance of a practical distributed routing algorithm. In Section V, we compare the results of the analysis in the previous sections to understand the conditions favorable for centralized routing and decentralized routing. Finally, we conclude the paper in Section VI.

II. System Model and Preliminaries

A satellite network has a time-varying topology due to the orbital dynamics of the satellites. The topology of the network evolves in the order of minutes, whereas routing over the network happens in the order of milliseconds. For this reason, the satellite network is modeled as a collection of snapshots with a fixed topology [13]. We are interested in the routing of packets in a given snapshot of the satellite network. We will now describe our model for the satellite network.

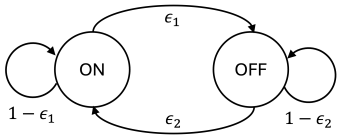


Fig. 2: Markovian Link dynamics of $l(t)$

A. Satellite Network Model

Consider a toroidal mesh grid of size $N \times M$ (see Figure 1). This models a particular snapshot of the satellite network. It is an abstract logical network model in which every satellite node is connected to its four nearest neighbors– two links in the same orbital plane and two links which are cross-orbit [15], [16], [21]. Here, N denotes the number of satellites per orbital plane and M denotes the number of orbital planes. In this model, there are NM nodes. Each node is labeled with a coordinate (x, y), such that

$$x \in \left\{ \left\lfloor -\frac{M}{2} \right\rfloor + 1, ..., -1, 0, 1, ..., \left\lfloor \frac{M}{2} \right\rfloor \right\}, \text{ and} \quad (1)$$

$$y \in \left\{ \left\lfloor -\frac{N}{2} \right\rfloor + 1, ..., -1, 0, 1, ..., \left\lfloor \frac{N}{2} \right\rfloor \right\}. \quad (2)$$

We divide the duration of the snapshots into discrete time slots, where each time slot is the time it takes to transmit a fixed-size packet from one satellite node to its neighbor. For simplicity, we assume that the time taken by a fixed-length packet to be transmitted across any link in this network is the same and equal to one time slot. Henceforth, we measure time in the units of slots. We consider a directed network, where the link from node (x_1, y_1) to (x_2, y_2) is different from the link from (x_2, y_2) to (x_1, y_1). The links in this network are modeled to be random, with dynamics that evolve according to a Markov Process. The state of link l at time t is given by

$$l(t) = \begin{cases} 1 & \text{Link } l \text{ is available for use at time slot } t \\ 0 & \text{otherwise.} \end{cases} \quad (3)$$

In other words, link l is ON at time t if $l(t) = 1$ and it is OFF at time t if $l(t) = 0$. The distribution of $l(t)$ follows Markovian dynamics, as depicted by Figure 2. In particular,

$$\begin{aligned} &\mathbb{P}(l(t) = 1 \mid l(t-1) = 0) = \epsilon_2 \\ &\mathbb{P}(l(t) = 0 \mid l(t-1) = 0) = 1 - \epsilon_2 \\ &\mathbb{P}(l(t) = 1 \mid l(t-1) = 1) = 1 - \epsilon_1 \\ &\mathbb{P}(l(t) = 0 \mid l(t-1) = 1) = \epsilon_1 \end{aligned}. \quad (4)$$

We assume that the state dynamics of each link are independent of the other for simplicity. However, even with this assumption, we can still model many phenomena over satellite links. For example, a link state $l(t)$ can model whether a queue buffer between two nodes is at capacity or not at capacity, hence capturing whether a link is congested or not. A second example would be when $l(t)$ models random optical link failures over time, where a failed optical link will take some random amount of time to be set up again and a functional optical link will remain functional for some random duration.

Another example is when the satellite links are allocated based on user priority, where the link capacities are mainly reserved for the primary users and the unused capacity is allocated to secondary users [21]. The dynamics of the availability of capacity to secondary users could be captured by Markovian dynamics.

The link state transition probabilities ϵ_1 and ϵ_2 model how fast the link state changes compared to the time it takes to transmit a fixed-size packet from one node to the other (time slot). The steady-state probability of the link being ON in a time slot is p and it is given by,

$$p := \mathbb{P}(l(t) = 1) = \frac{\epsilon_2}{\epsilon_1 + \epsilon_2}. \tag{5}$$

Consequently, the probability of the link being OFF is $(1-p)$. We will also be interested in the $k-$stage transition probabilities of the links, namely,

$$\mathbb{P}\left(l(t+k) = j \;\middle|\; l(t) = i \right) = p_{ij}(k). \tag{6}$$

We have the following expression for $p_{ij}(k)$,

$$\begin{aligned}
p_{00}(k) &= \frac{\epsilon_1 + \epsilon_2(1-\epsilon_1-\epsilon_2)^k}{\epsilon_1+\epsilon_2} = (1-p) + p\mu^k \\
p_{01}(k) &= \frac{\epsilon_2 - \epsilon_2(1-\epsilon_1-\epsilon_2)^k}{\epsilon_1+\epsilon_2} = p - p\mu^k \\
p_{10}(k) &= \frac{\epsilon_1 - \epsilon_1(1-\epsilon_1-\epsilon_2)^k}{\epsilon_1+\epsilon_2} = (1-p) - (1-p)\mu^k \\
p_{11}(k) &= \frac{\epsilon_2 + \epsilon_1(1-\epsilon_1-\epsilon_2)^k}{\epsilon_1+\epsilon_2} = p + (1-p)\mu^k,
\end{aligned} \tag{7}$$

where $\mu := 1 - \epsilon_1 - \epsilon_2$ is the *memory parameter* of the link. When the *memory parameter* $\mu = 0$, the links have no memory (i.e., i.i.d. in time), and when $\mu \to 1$, the links are static once realized. We assume that $\mu \geq 0$, i.e., the links have *positive memory* [22]. This assumption ensures that a link that is ON is more likely to be ON after k time slots, compared to the probability of a link that is OFF turning ON, i.e., $p_{11}(k) > p_{01}(l), \forall l$. Loosely speaking, this assumption prevents network link states from switching rapidly, allowing for sensible routing decisions.

We are interested in routing between a source and a destination on a packet-by-packet basis. In a particular snapshot of the network, the satellite network forms a toroidal mesh grid (Figure 1) over the Earth. The source and destination on the ground are generally served by the satellites closest to them, within their line of sight. To this end, routing a packet from the on-ground source to the on-ground destination in a particular snapshot essentially boils down to routing the message from the satellite node serving the source to the satellite node serving the destination over the dynamic satellite network. This is reasonable as it is fairly straightforward to precompute the satellite that would serve a geographic region (consequently, the on-ground source and destiantion) at any given time.

Hence, without loss of generality, we can fix the destination satellite node to have label $(0,0)$. Then, a satellite node (x,y) is located $|x|+|y|$ hops away from the destination. It would take at least $|x|+|y|$ time slots to transmit a packet from (x,y) to $(0,0)$. Without loss in generality, we assume that $x > 0, y > 0$ and $|x|+|y| = x+y$ in the rest of the paper. This is due to the symmetry of the network along the vertical and horizontal axes.

B. Routing Schemes

1) *Centralized Routing Scheme*: In this scheme, we assume that a centralized router has delayed knowledge of the satellite links' state. This delay arises due to various factors, but it is primarily due to the time it takes for an electromagnetic wave to propagate from the satellite back to the Earth. Whenever a ground station (GS) needs to transmit a fixed-length packet to another GS through the satellite network, the centralized router finds a path for the packet using the network state information and appends the path onto the packet. Essentially, a *virtual circuit* is established from the source GS to the destination GS through the satellite network. This form of routing is also sometimes referred to as *virtual circuit routing* or *layer 2 routing*. Once the centralized router computes a path for the packet, the source GS forwards it to the first satellite (source satellite). One by one, the satellites decode the path appended with the packet, and forward it to the next satellite in the path, until the destination satellite is reached. The satellites have no onboard intelligence and only forward the packet to the next node according to the path appended in the header. The centralized router finds a path from source to destination with high probability, using stale global network state information. However, the links can change state after some time, due to their Markovian dynamics.

2) *Distributed Routing Scheme*: In this scheme, every satellite has onboard routing intelligence and there is no centralized router. Every satellite knows the state of adjacent inter-satellite links in near real-time. However, the satellite router does not have global knowledge of all link states in real-time. Though it is possible to garner global network state information and use it for routing decisions, it leads to increased computation and communication complexity and is best avoided onboard satellites. This means that the router must make routing decisions with partial, yet real-time network state information. The source ground station (GS) transmits its packet to the first satellite, and each satellite then decides the next satellite to which the packet must be forwarded in the network, until the destination GS is reached. This form of routing is also sometimes referred to as *IP routing* or *layer 3 routing*, as it involves route computation at each node.

C. Performance Metrics and Scenarios of Interest

We are interested in two metrics, *throughput* and *delay*. We define throughput T_{xy} as the probability of successfully routing a packet from the source (x,y) to the destination $(0,0)$. We define the delay D_{xy} as the average number of slots it takes to route a packet from source (x,y) to destination $(0,0)$. We compare the throughput and delay of the routing schemes in the network model defined above in two scenarios, 1) without onboard buffering and 2) with onboard buffering.

In the first scenario, the satellite nodes have no buffering capabilities. In other words, the packet in consideration is dropped by a satellite node whenever the respective outgoing link along the chosen path is in the OFF state. This means that the packet suffers no significant buffering or queueing delay along its path. Since the packet only experiences propagation and transmission delays, any routing policy would have nearly minimum possible delay in this scenario. However, the throughput will naturally suffer as it is likely that a packet will be dropped in transit. We are interested in routing policies that maximize throughput in this scenario.

In the second scenario, the satellite nodes have buffering capabilities. Consequently, the throughput is nearly maximum for any routing policy, as packets are rarely dropped. However, they can be significantly delayed due to buffering, while the packet waits for a link to become available again. We are interested in routing strategies that minimize the buffering delay in this scenario.

In the following sections, we analyze centralized and distributed routing schemes in the two scenarios.

III. Analysis of a Centralized Routing Scheme

In this section, we introduce and motivate a simple centralized routing policy called the Shortest Connected Path Routing (SCPR). We will then analyze its performance without buffering and with buffering.

Recall that in a centralized setting, the router gets global network state information after some inherent delay. Let parameter t_c denote the amount of time since the network state information was most recently observed, until the packet reaches the source satellite. The parameter t_c includes the propagation delays, path computation delay, as well as other delays in the system.

We will use γ to denote a path from (x, y) to $(0, 0)$. The path γ is a collection of links which connects (x, y) to $(0, 0)$. Now, we define the term "connected" in the Shortest Connected Path Routing.

Definition 1 (Connected Path)**.** *A path γ is connected at time t if all the links $l \in \gamma$ are ON at time t. In other words, $l(t) = 1$, $\forall l \in \gamma$.*

The Shortest Connected Path Routing (SCPR) is a simple practical algorithm. The policy observes the network state at the current time t and selects the shortest connected path from the source (x, y) to the destination $(0, 0)$. If there exists no connected path between (x, y) and $(0, 0)$, then it just selects one of the shortest paths at random. SCPR is visually summarized in Figure 3.

SCPR is representative of the class of centralized policies since any such policy would choose reliable paths with delayed network state information. This entails knowing that the path was observed to be ON, and minimizing path length, as shorter paths are more reliable. As discussed in the introduction, the optimal centralized policy to maximize throughput or minimize delay requires solving a complex dynamic program, and SCPR is not always optimal. However, it is easy to see

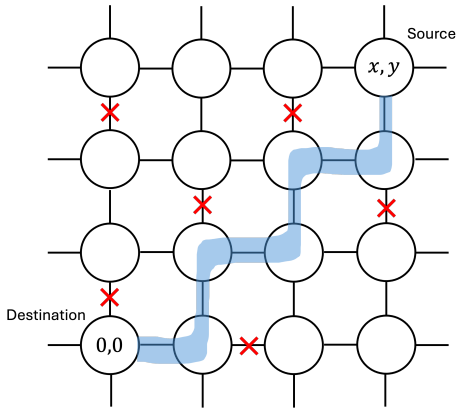


Fig. 3: Given the network state at $t = 0$, the SCPR policy gives us the shortest connected path from source to destination, as highlighted. Here, the red crosses denote links being unavailable for use.

that the shortest *connected* path is throughput optimal if it is also one of the shortest paths from (x, y) to $(0, 0)$. In fact, the shortest connected path between is also very likely to be a shortest path based on the numerical analysis in [23]. In the following subsection, we will analyze SCPR throughput in the case where satellites do not have onboard buffering capabilities.

A. Shortest Connected Path Routing Without Buffering

The throughput for the SCPR policy in this scenario is given by the probability that there exists a path to the destination and the probability that all links are ON at the time they are traversed. Claim 1 states an upper bound for throughput under the SCPR policy.

Claim 1 (Throughput Under SCPR Policy)**.** *For the centralized SCPR policy, the throughput is upper bounded as*

$$T_{xy}^{SCPR} \leq \prod_{i=0}^{x+y-1} \left(p + (1-p)\mu^{t_c+i}\right). \tag{8}$$

We provide an upper bound because exactly characterizing T_{xy}^{SCPR} requires counting all cuts from source to destination in the graph (which is provably NP-hard in general [24]). However, the upper bound is a good approximation to T_{xy}^{SCPR}, and is close to its true value, verified in Section V. We defer the proof of the claim to our technical report [25]. The proof mainly relies on the fact that the path length is lower bounded by $x+y$, which consequently gives us an upper bound on throughput. It is interesting to note that the throughput decays exponentially with the distance between the source and destination, as well as the inherent delay, t_c (for fixed μ).

B. Shortest Connected Path Routing With Buffering

If we define X_i as the delay incurred to traverse the i^{th} link in γ, then, the total expected delay due to SCPR, D_{xy}^{SCPR} is

$$D_{xy}^{SCPR} = \mathbb{E}\left[\sum_{i=0}^{|\gamma|-1} X_i\right].$$

It is important to note that the distribution of X_i depends on the delay incurred before reaching link i, i.e., $\sum_{k=0}^{i-1} X_k$. This makes the expression for the delay D_{xy}^{SCPR} non-trivial to compute. Claim 2 provides a recursive method to lower bound the mean delay of the SCPR policy.

Claim 2. *For the centralized SCPR policy with buffers, the mean delay is lower bounded as*

$$D_{xy}^{SCPR} \geq \frac{d}{dt} M_{x+y}(t)\Big|_{t=0} \tag{9}$$

where,

$$M_i(t) \triangleq \mathbb{E}\left[\mu^{t\sum_{k=0}^{i-1} X_k}\right] / \log \mu,$$

and it satisfies the following recursive equation

$$M_i(t) = A(t)M_{i-1}(t) + B(t)M_{i-1}(t+1),$$

and

$$A(t) = \frac{p(1-\mu^t)\mu^t + \epsilon_2\mu^{2t}}{1-(1-\epsilon_2)\mu^t}$$
$$B(t) = \frac{(1-p)\mu^{t_c+t}(1-\mu^t)}{1-(1-\epsilon_2)\mu^t}$$
$$M_0(t) = 1/\log(\mu).$$

Even though Claim 2 provides a lower bound for the delay, it is a close approximation for the true mean delay of the SCPR policy for large values of p. This is also verified in Section V through simulation. The approximate value can be efficiently numerically computed using the recursive relation in Claim 2. The proof of Claim 2 is deferred to the technical report [25]. We obtain the expression in Claim 2 by first lower bounding the path length by $x+y$, and then handling the correlation between X_i and $\sum_{k=0}^{i-1} X_i$ using conditional expectations. To compute the conditional expectation, we use $M_i(t)$, which is indeed the moment generating function of $\sum_{k=0}^{i-1} X_i$. The details of the proof are skipped for brevity.

IV. Analysis of a Distributed Routing Scheme

In this section, we consider the setting where satellites have onboard routing capabilities. In this setting, the satellite nodes know the state of nearby links (local information) in real-time and make real-time routing decisions. A practical shortest-path distributed routing policy is the greedy routing (GR) policy. The principle of the GR policy is to move closer to the destination at every step, whenever possible. It makes sense to do so as we have only restricted information about the network state. Clearly, the GR policy always takes the shortest path to the destination. To define the GR policy more formally, let the packet initially be at (x, y) that $x, y > 0$ for the sake of simplicity. Moreover, let l_h and l_v be the links along the horizontal and vertical directions respectively, that move toward the destination, i.e., moving along l_v at (x, y) would take you to $(x, y-1)$, and moving along l_h would take you to $(x-1, y)$. At every node (x', y'), repeat the process (see Figure 4). We define the GR policy as follows–

1) At some node (x', y'), observe the state of the local outgoing links l_v and l_h.

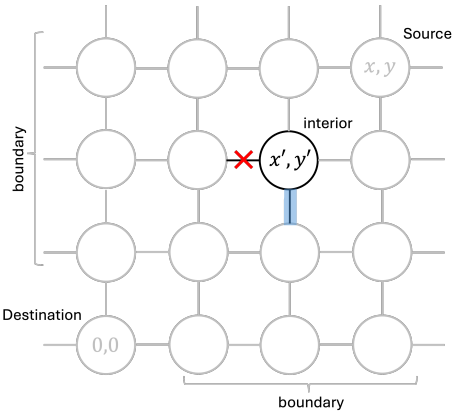


Fig. 4: Greedy Routing (GR) Policy at node (x', y'). The nodes and links with a dark outline are the ones under consideration. The GR policy only considers the local links going towards the destination for routing. Since the horizontal link is unavailable, the routing decision is to move along the vertical, as highlighted.

2) One of the following cases occurs–

a) $x' > 0$ and $y' > 0$. We are in the interior of the grid between the source and the destination. In this case, we have two edges that can potentially take us closer to the destination, l_v and l_h. We further have four cases.
   i) *l_v and l_h are both ON*: Choose to move along l_v with probability u (or along l_h with probability $\bar{u} = 1-u$).
   ii) *l_v is OFF, l_h is ON*: Choose to move along l_h.
   iii) *l_h is OFF, l_v is ON*: Choose to move along l_v.
   iv) *l_v and l_h are both OFF*: Depending on whether or not we use onboard buffers, we either buffer the packet or drop it.

b) $x' = 0$. In this case, we are at the boundary. Only the l_v link can take us closer to the destination. We move along l_v if it is ON. Else, we drop the packet or buffer it depending on whether the satellites have buffers.

c) $y' = 0$. This case is similar to the previous case. We move along l_h if it is ON. Else, we drop the packet or buffer it depending on whether the satellites have buffers.

3) We repeat this policy until we reach the destination, i.e. $x' = 0$ and $y' = 0$.

It is important to point that the above algorithm ignores the memory in the network. Every routing decision is made only using the state of the outgoing links at a given node. Therefore, the state of the current outgoing links are **independent** of all routing decisions made earlier. This point is very important to the analysis that follows.

The GR policy is not the optimal distributed policy in general. For the general case, finding the optimal distributed routing problem becomes a complex dynamic program as discussed in the introduction. We study the GR policy as a representative distributed routing algorithm to simplify analysis. In the policy defined above, we use randomization to

break ties when both links l_h and l_v are ON, i.e., we choose l_v with probability u and l_h with probability $\bar{u} = 1 - u$ to simplify analysis. In the rest of the section, we analyze the greedy policy's performance.

A. Greedy Routing (GR) Policy Without Buffering

When there are no buffers onboard satellites, we are interested in the throughput of the routing policy. Claim 3 states the expression for the throughput of the greedy policy.

Claim 3. *The throughput of the Greedy Routing policy is given as*

$$T_{xy}^{GR} = p^y \left(\frac{1-up}{\bar{u}}\right)^x I_{\bar{u}p}(x,y) + p^x \left(\frac{1-\bar{u}p}{u}\right)^y I_{up}(y,x), \tag{10}$$

where, u is probability of routing along the vertical direction when both links are ON, $\bar{u} = 1 - u$ and $I_v(a,b)$ denotes the Regularized Incomplete Beta Function [26] defined as

$$I_v(a,b) = \frac{\int_0^v t^{a-1}(1-t)^{b-1}dt}{\int_0^1 t^{a-1}(1-t)^{b-1}dt}. \tag{11}$$

The proof of the claim mainly relies on analyzing the probability of reaching a state when no outgoing link toward the destination is ON. In this case, we drop the packet. Consequently, this requires us finding the distribution of when the packet reaches the **boundary** of the grid (see Figure 4), because, along the boundary, **there is only one outgoing link toward the destination**. This is contrary to the case in the **interior**, where we always have **two potential links toward the destination**. Figure 4 shows the interior and boundary regions for the GR policy. Therefore, we want to avoid reaching the boundary early so as to always have the option to take one of the two links that takes us closer to the destination. Geometrically, this also means that we want to take a path to the destination that is as close to the diagonal between (x, y) and $(0, 0)$ as possible. The proof then involves counting the number of ways of successfully reaching the destination. This gives rise to combinatorial sums which we conveniently express using the Regularized Incomplete Beta Function [26]. The details of the proof of Claim 3 are deferred to our technical report [25] due to space constraints.

Observe that the throughput in Claim 3 is not symmetric with respect to x, y, and depends on u. But from the symmetry of the network, we would expect $T_{xy}^{GR} = T_{yx}^{GR}$. Moreover, as mentioned earlier, we want to stay in the interior of the grid for as long as possible to increase the possibility of encountering an ON link toward the destination. Choosing $u = \frac{y}{x+y}$ (and $\bar{u} = \frac{x}{x+y}$) helps satisfy these requirements. Firstly, this would result in staying close to the diagonal. Secondly, (10) becomes symmetric as we pick u appropriately depending on x, y. One can also show that $u = \frac{y}{x+y}$ maximizes T_{xy}^{GR}, asymptotically as $x \to \infty, y \to \infty$.

B. Greedy Routing (GR) Policy With Buffering

When the satellites have buffering capabilities, we are interested in the delay performance of the routing scheme. Claim 4 gives an approximate formula for the mean delay of the GR Policy with buffers.

Claim 4. *The mean delay of the GR Policy is upper bounded as*

$$\begin{aligned} D_{xy}^{GR} \leq \; & (x+y)\left(1 + \frac{(1-p)^2}{2\epsilon_2 - \epsilon_1^2}\right) \\ & + \frac{(1-p)(1-\epsilon_2+p)}{2\epsilon_2 - \epsilon_2^2}\sqrt{\frac{x+y}{2\pi w\bar{w}}}, \end{aligned} \tag{12}$$

where, $w = \frac{y}{x+y}$ and $\bar{w} = 1 - w$.

In the above expression, w is actually the probability of routing along the vertical direction, and clearly, it depends on u, which is the probability of choosing the vertical direction to break ties.

To prove the above claim, we need to calculate the average delay at every step in the routing algorithm, since every step is independent of the previous steps. It is straightforward to note that in the interior, the average delay experienced is $1 + \frac{(1-p)^2}{2\epsilon_2 - \epsilon_2^2}$, whereas, in the boundary region, the average delay experienced is $1 + \frac{1-p}{\epsilon_2}$. Note that the average delay in the interior is lower than the average delay in the boundary. Again, this is because along the boundary, **there is only one outgoing link toward the destination**, and so, if it is OFF, we drop the packet. This is contrary to the case in the interior, where we always have two potential links toward the destination. This implies that even if one is OFF, there is a likely chance we can take the other link towards the destination, in turn reducing delay. As noted above, w depends on u. Consequently, u is strategically chosen so as to make $w = \frac{y}{x+y}$. The reason to do this is the same – we want the packet to be routed through the interior for as long as possible. Finally, we have to determine when the packet reaches the boundary region on average, like in the proof of Claim 3, which gives rise to combinatorial terms. These terms are then upper-bounded using Stirling's Approximation [27] to get the result of Claim 2.

The detailed proof can be found in our technical report [25]. The analysis is similar to the one presented in [28], where they analyze the asymptotic average delay per hop in a square-grid store-and-forward network with i.i.d. links. Our analysis considers a non-asymptotic non-square grid setting where the links follow Markovian dynamics.

V. Comparison Of Routing Strategies and Numerical Analysis

Intuitively, a centralized routing policy would fare well when the links have a lot of memory, as the delayed network state information would still be valid after a long time, whereas, a distributed policy would work better in cases where there is no memory and the router needs to be adaptive. We confirm this intuition in this section. First, we begin with a comparison in the case without buffers, and subsequently compare them in the case with buffers. Additionally, we plot the empirical performance of these algorithms in a 100×100 network using Monte-Carlo methods over 2000 trials, and show that the simulations match the analytical results.

A. Comparison Of Routing Strategies Without Buffers

In this subsection, we compare the throughput performance of the SCPR policy and GR policy without buffers. Comparing equations (8) and (10), we see that a sufficient condition for the GR policy to be better than the SCPR policy is

$$\left(\frac{1-up}{\bar{u}p}\right)^x I_{\bar{u}p}(x,y)+\left(\frac{1-\bar{u}p}{up}\right)^y I_{up}(y,x) \geq \prod_{i=0}^{x+y-1}\left(1+\frac{1-p}{p}\mu^{t_c+i}\right). \quad (13)$$

We plot (8) and (10) and the simulated throughput of the routing algorithms in Figure 5. In the figures, note that the inherent delay t_c is normalized to the time it takes to transmit a packet along one hop in the satellite network. In all the numerical analysis, we fix $p = 0.9$, $N = M = 100$. In Figure 5a, observe that the SCPR policy only outperforms the GR policy when there is high memory in the system $\mu \approx 1$. This is because a connected path selected by the centralized controller is likely to be connected in the future only when there is high memory. In Figure 5b, we fix $\mu = 0.99$ and study how SCPR's throughput varies with t_c. Observe that throughput of SCPR policy falls exponentially with increasing t_c. Even with high memory, GR policy outperforms the SCPR policy when $t_c > 35$. This is because the network is very likely to have changed state, rendering the planned path useless when the inherent delay of the centralized routing scheme is large. Lastly, we compare the throughput of these policies as a function of distance between the source and destination. In Figure 5c we observe that the throughput of both policies falls drastically as the distances become large. In the small distance regime, with high memory and low inherent delay, a centralized policy can be better. However, for large distances, a distributed policy would dominate. From the figures, we see that the simulated performance closely matches the analytical results. The mismatch is largely due to the variance of Monte-Carlo methods.

B. Comparison of Routing Strategies With Buffers

In this subsection, we compare the delay performance of the SCPR policy and GR policy with buffers. Comparing equations (9) and (12), we see that a sufficient condition for the GR policy to be better than the SCPR policy is

$$(x+y)\left(1+\frac{(1-p)^2}{2\epsilon_2-\epsilon_2^2}+\frac{\frac{1-p}{\epsilon_2}\left(1-\frac{1-p}{2-\epsilon_2}\right)}{\sqrt{2\pi w\bar{w}(x+y)}}\right) \leq M'_{x+y}(0). \quad (14)$$

We plot the analytical mean delays, along with simulated delays, in Figure 6 to better understand their behavior. We fix $p = 0.9$. In Figure 6a, we plot the delay against μ and we observe that the delay increases with μ monotonically. Moreover, the GR policy greatly outpeforms the SCPR policy in terms of delay unless the network has high memory $\mu \approx 1$. In Figure 6b, we see how the delay varies with increase in t_c when $\mu = 0.99$. Note that the GR policy outperforms SCPR for $t_c > 32$. Similarly, looking at Figure 6c the GR policy is significantly better in terms of delay for large distances, and is comparable at small distances. From the figures, we verify that the simulation closely matches the analytical results.

VI. Conclusions and General Remarks

As we see from the above numerical analysis, the centralized policy is only advantageous when there is high memory in the network links, t_c is low and number of hops to reach the destination is small. This happens when the time scale of state changes is much larger than the time it takes to transmit a packet across one hop, or $\epsilon_1 + \epsilon_2 \ll 1$. An example of this case would be when we want to model link unavailability due to optical link failures. The time it takes to establish an optical link is in the order of minutes which is much larger than the transmit time of a fixed size packet, and the links remain fairly stable once an optical connection is made. In this scenario, it would make sense to use a centralized scheme for route planning. For other scenarios, it would make more sense to use a distributed policy. For example, if we want to model link unavailability due to user priority, the links transition from the ON to the OFF state and vice-versa, at around the same timescale as transmission of a packet along one hop. In this case, a distributed scheme makes more sense.

Acknowledgement

We would like to thank SES S.A. for funding this research. We especially extend our gratitude to Joel Grotz and Valvanera Moreno for their valuable insights and critique throughout the course of this research.

References

[1] J. A. Fraire and E. L. Gasparini, "Centralized and Decentralized Routing Solutions for Present and Future Space Information Networks," *IEEE Network*, vol. 35, no. 4, pp. 110–117, Jul. 2021.

[2] E. Ekici, I. Akyildiz, and M. Bender, "A distributed routing algorithm for datagram traffic in LEO satellite networks," *IEEE/ACM Transactions on Networking*, vol. 9, no. 2, pp. 137–147, Apr. 2001.

[3] P. S. Page, K. S. Bhargao, H. V. Baviskar, and G. S. Kasbekar, "Distributed Probabilistic Congestion Control in LEO Satellite Networks," in *2023 15th International Conference on COMmunication Systems & NETworkS (COMSNETS)*, Jan. 2023, pp. 335–339.

[4] X. Liu, X. Yan, Z. Jiang, C. Li, and Y. Yang, "A Low-Complexity Routing Algorithm Based on Load Balancing for LEO Satellite Networks," in *2015 IEEE 82nd Vehicular Technology Conference (VTC2015-Fall)*, Sep. 2015, pp. 1–5.

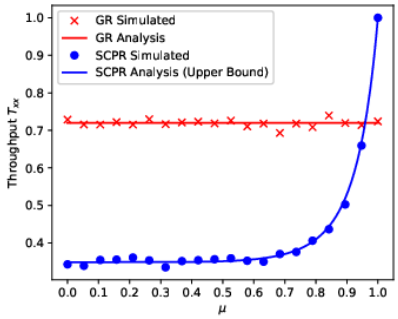
[5] T. Taleb, D. Mashimo, A. Jamalipour, K. Hashimoto, Y. Nemoto, and N. Kato, "SAT04-3: ELB: An Explicit Load Balancing Routing Protocol for Multi-Hop NGEO Satellite Constellations," in *IEEE Globecom 2006*, Nov. 2006, pp. 1–5.

[6] X. Deng, L. Chang, S. Zeng, L. Cai, and J. Pan, "Distance-Based Back-Pressure Routing for Load-Balancing LEO Satellite Networks," *IEEE Transactions on Vehicular Technology*, vol. 72, no. 1, pp. 1240–1253, Jan. 2023.

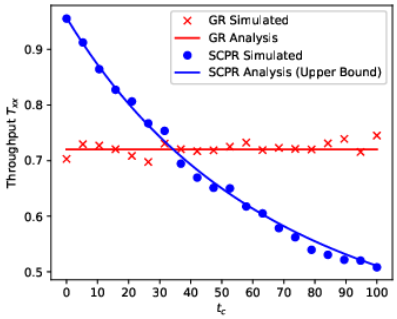
[7] D. P. Bertsekas and R. G. Gallager, *Data networks*, 2nd ed., ser. Prentice-Hall International editions. Englewood Cliffs, NJ: Prentice-Hall International, 1992.

[8] M. S. Corson and A. Ephremides, "A distributed routing algorithm for mobile wireless networks," *Wireless Networks*, vol. 1, no. 1, pp. 61–81, Mar. 1995.

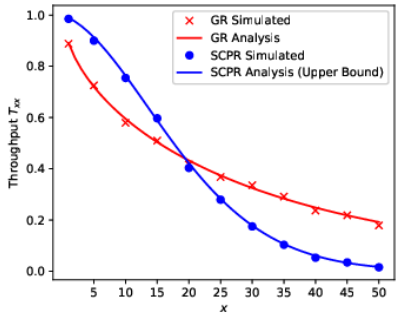
[9] C. Perkins and E. Royer, "Ad-hoc on-demand distance vector routing," in *Proceedings WMCSA'99. Second IEEE Workshop on Mobile Computing Systems and Applications*, Feb. 1999, pp. 90–100.



(a) Throughput vs. μ.
We fix $p = 0.9$, $x = y = 5$, $t_c = 5$.

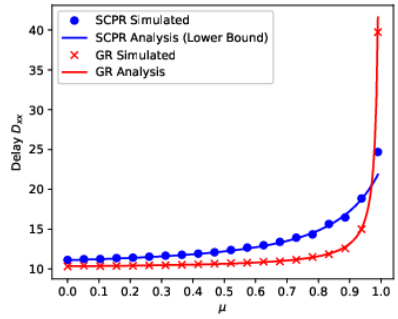


(b) Throughput vs. t_c.
We fix $p = 0.9$, $x = y = 5$, $\mu = 0.99$.

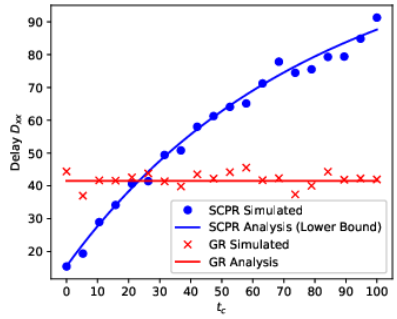


(c) Throughput vs. x.
We fix $p = 0.9$, $x = y$, $\mu = 0.99$, $t_c = 5$.

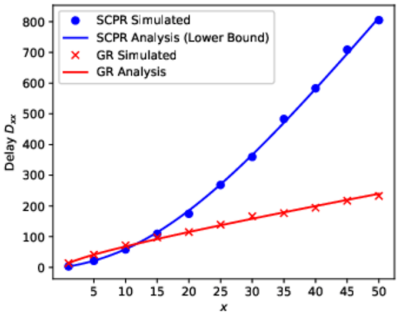
Fig. 5: SCPR and GR throughput against different network parameters



(a) Delay vs. μ.
We fix $p = 0.9$, $x = y = 5$, $t_c = 5$.



(b) Delay vs. t_c.
We fix $p = 0.9$, $x = y = 5$, $\mu = 0.99$.



(c) Delay vs. x.
We fix $p = 0.9$, $t_c = 5$, $x = y$, $\mu = 0.99$.

Fig. 6: SCPR and GR delay against different network parameters

[10] D. B. Johnson and D. A. Maltz, "Dynamic Source Routing in Ad Hoc Wireless Networks," in *Mobile Computing*, T. Imielinski and H. F. Korth, Eds. Boston, MA: Springer US, 1996, pp. 153–181.

[11] B. Karp and H. T. Kung, "GPSR: Greedy Perimeter Stateless Routing for Wireless Networks," in *Proceedings of the 6th annual international conference on Mobile computing and networking*, ser. MobiCom '00. New York, NY, USA: Association for Computing Machinery, Aug. 2000, pp. 243–254.

[12] E. Modiano and A. Ephremides, "Efficient algorithms for performing packet broadcasts in a mesh network," *IEEE/ACM Transactions on Networking*, vol. 4, pp. 639–648, Aug. 1996.

[13] M. Werner, "A dynamic routing concept for ATM-based satellite personal communication networks," *IEEE Journal on Selected Areas in Communications*, vol. 15, no. 8, Oct. 1997.

[14] M. Neely, E. Modiano, and C. Rohrs, "Power allocation and routing in multibeam satellites with time-varying channels," *IEEE/ACM Transactions on Networking*, vol. 11, no. 1, pp. 138–152, 2003.

[15] J. Sun and E. Modiano, "Capacity provisioning and failure recovery in mesh-torus networks with application to satellite constellations," in *Proceedings ISCC 2002 Seventh International Symposium on Computers and Communications*. Taormina-Giardini Naxos, Italy: IEEE Comput. Soc, 2002, pp. 77–84.

[16] ——, "Routing strategies for maximizing throughput in LEO satellite networks," *IEEE Journal on Selected Areas in Communications*, vol. 22, no. 2, Feb. 2004.

[17] M. Wang, L. Wei, Y. Wang, and Y. Liu, "Orbit-Grid-Based Dynamic Routing for Software Defined Mega-Constellation Network," in *GLOBECOM 2023 - 2023 IEEE Global Communications Conference*, Dec. 2023, pp. 844–849, iSSN: 2576-6813.

[18] Y. Li, L. Liu, H. Li, W. Liu, Y. Chen, W. Zhao, J. Wu, Q. Wu, J. Liu, and Z. Lai, "Stable Hierarchical Routing for Operational LEO Networks," in *Proceedings of the 30th Annual International Conference on Mobile Computing and Networking*. Washington D.C. DC USA: ACM, May 2024, pp. 296–311.

[19] G. Zheng, N. Wang, and R. R. Tafazolli, "SDN in Space: A Virtual Data-Plane Addressing Scheme for Supporting LEO Satellite and Terrestrial Networks Integration," *IEEE/ACM Transactions on Networking*, vol. 32, no. 2, pp. 1781–1796, Apr. 2024.

[20] E. C. Akrida, G. B. Mertzios, S. Nikoletseas, C. Raptopoulos, P. G. Spirakis, and V. Zamaraev, "How fast can we reach a target vertex in stochastic temporal graphs?" *Journal of Computer and System Sciences*, vol. 114, pp. 65–83, 2020.

[21] M. Handley, "Delay is Not an Option: Low Latency Routing in Space," in *Proceedings of the 17th ACM Workshop on Hot Topics in Networks*. Redmond WA USA: ACM, Nov. 2018, pp. 85–91.

[22] M. Johnston and E. Modiano, "Wireless Scheduling with Delayed CSI: When Distributed Outperforms Centralized," *IEEE Transactions on Mobile Computing*, vol. 17, no. 11, pp. 2703–2715, Nov. 2018.

[23] L. Zhang, L. Cai, and J. Pan, "Connectivity in two-dimensional lattice networks," in *2013 Proceedings IEEE INFOCOM*. Turin, Italy: IEEE, Apr. 2013, pp. 2814–2822.

[24] J. S. Provan and M. O. Ball, "The Complexity of Counting Cuts and of Computing the Probability that a Graph is Connected," *SIAM Journal on Computing*, vol. 12, no. 4, pp. 777–788, Nov. 1983.

[25] R. V. Ramakanth and E. Modiano, "Centralized versus distributed routing for large-scale satellite networks," 2025, arXiv:2501.13744.

[26] W. Feller, *An Introduction to Probability Theory and Its Applications*. Wiley, 1991.

[27] R. Gallager, *Information Theory and Reliable Communications*. Wiley, 1991.

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