Abstract

We discuss a region-partitioning approach to the problem of reconfiguring a point-to-point satellite computer network that use lasers as communication links. Since the network is highly mobile and the communication links are constrained to the line-of-sight between satellites, the network must be reconfigured dynamically by re-targeting the transceivers among satellites. The objective of the reconfiguration is to maximize survivability and performance by optimizing the network connectivity. Our approach to this reconfiguration is a divide-and-conquer method. The network is partitioned into small subnetworks by dividing the satellite orbiting space into small regions. Then links are assigned to maximize the node-connectivity of the subnetwork. Finally, subnetworks are connected together to form a complete network. We show that this approach provides maximum connectivity for large-scale networks. The algorithm is very efficient and has \( O(n \log n) \) time complexity. The algorithm is replicated to each satellite, and is run simultaneously by all satellites on a periodic basis.

1. Introduction

A low altitude satellite communication network connected through the use of laser transceivers installed in each satellite has the advantages of high communication speed, large channel capacity and strong security [1], Such networks are of particular interest to military applications such as SSI. The link assignment problem in satellite networks, unlike that for conventional store-and-forward computer networks, has the following properties: 1) variable propagation delay due to the changing length of the links, 2) visibility constraints as due to the blocking of the Earth, not every pair of satellites can see each other, and 3) periodic link reconfiguration as the line of sight between two connected satellites may be blocked by the Earth over time. Since network reconfiguration in the laser-based point-to-point satellite networks involves link reassignment, we will use the two terms, link assignment and reconfiguration, interchangeably throughout this paper.

The problem of maximizing connectivity of a network has been studied extensively in point-to-point store-and-forward networks [3, 10, 11, 1]. This problem can also be transformed into that of finding the maximum number of node-disjoint paths between every pair of nodes in a graph [8, 12, 15]. In particular, Schumacher [15] has given an \( O(n^2) \) algorithm (where \( n \) is the number of nodes in the graph) for constructing \( k \)-regular graphs with maximum connectivity. The algorithm presented by Schumacher is the fastest algorithm as far as the time complexity is concerned. However, Schumacher’s approach is a pure graph algorithm in which every pair of nodes are connectable. Thus the algorithm is inapplicable (at least directly) to satellite network reconfigurations. The visibility constraint on the satellite computer networks makes it more difficult to reconfigure a network with respect to maximizing network connectivity because each node can see only a subset of the nodes. Relatively few works have been done on maintaining network connectivities for point-to-point satellite networks. McLochin [13] and Ward [17] proposed methods to handle a special case of this link reconfiguration problem by imposing strict constraints on satellite orbits so that satellites circulate on several equally-spaced planes and within each plane satellites are approximately equidistant from their neighbors.

Another issue in point-to-point satellite networks is to determine how reconfiguration is invoked. Research on other mobile networks also deals with the problem of network reconfiguring such as those for packet-radio networks [8, 9, 14]. Most of these works adopt the failure-driven reconfiguring approach, i.e. as long as the node or link is properly functioning, no reconfiguration is needed [4]. One criterion often adopted in packet-radio network reconfigurations is “minimum disturbance” [8] in the sense that reconfiguration normally requires updating or swapping links locally and thus the unaffected links should be preserved. In point-to-point satellite networks, this “minimum disturbance” is very difficult to obtain because most of the existing links will be blocked by the Earth sooner or later. Considering the periodic orbiting of satellites, the link reconfiguration needs to be carried out globally for there is no way to restrict the reconfiguration to a local level. Instead, we choose to do over-all network reconfiguration on a periodic basis to ensure that a network is highly-connected all the time except at the moments of reconfiguration.

In this paper, we provide an algorithm to the link assignment problem of the satellite network with no constraints on satellites’ orbits. The criterion adopted in our approach is maximum network connectivity. The algorithm consists of two levels of tasks: In executing the first-level task, the space in which the satellites orbit is partitioned into a number of approximately equal-sized regions. Each satellite identifies itself as a member of a region in which a subnetwork is to be formed. Each subnetwork is constructed by invoking the link assignment algorithm. The algorithm, based on the number of satellites in the region and some pre-determined configurations, will efficiently yield a highly connected topology for this subnetwork. In processing the second-level task, the “backbone satellites” are invoked to set up the whole network by establishing “backbone links”. We show that the algorithm has \( O(n \log n) \) time complexity, and that maximum connectivity can be achieved if \( n \), the number of nodes in the network, is sufficiently large.

In section 2, we discuss a general model for the design and analysis of the satellite network. Under this model, the region concept and the partition approach are introduced. In section 3, we describe the two-level partition approach in detail. In section 4, we
show that the maximum connectivity can be obtained if some constraints are satisfied. In section 5, we summarize what we have achieved and discuss some directions for future work.

2. A Satellite Network Model

2.1 Assumptions

For systems as complex as satellite networks, it is necessary to make some basic assumptions so that solutions are feasible. Here we assume the following things hold:

(1) **Global Knowledge** - It is assumed that information about exact locations, directions of motion and other related data about the satellites can always be derived through mathematical computations based on orbital mechanics or alternatively from broadcasting of ground-based tracking stations.

(2) **Fixed Number of Links at Each Satellite** - Ward[9] has shown that it is most cost-effective if each satellite is installed with 4 transceivers. In our model we also assume that each satellite has exactly 4 transceivers. So, except in some unusual circumstances, the network can be regarded as a 4-regular graph, i.e., each node has exactly 4 edges.

(3) **Global Time Scheme** - Each satellite will need a clock synchronized with those on the other satellites. Since communication links between satellites are established by targeting laser transceivers to each other, some kind of synchronization is necessary. In this case we believe that using a global clock to drive the link assignment process is the simplest way to ensure reliable retargeting.

(4) **Periodic Reconfiguration** - The reconfiguration is done by retargeting transceivers among satellites to establish a new set of links on a periodic basis. Each satellite synchronizes its reconfiguring process to the others at regular periods. The reconfiguration is synchronized by having the satellites run their reconfiguring process simultaneously. During this short period of network reconfiguration, all packets can be exchanged between satellites. The advantage of periodic reconfiguration is that it allows satellites to perform more smoothly by taking reconfiguration into account. For example, by knowing the next reconfiguration is approaching, a node may decide to disconnect the virtual circuits built on the present topology and/or set up some new ones based on the incoming topology.

2.2 Partitioning Space into Regions

A good solution to the link assignment problem should satisfy two goals - maximize connectivity, and be computationally efficient. The approach we propose satisfies these requirements in handling large networks. In this method, a network is partitioned into small subnetworks. Each subnetwork independently generates a locally optimized topology in terms of connectivity. And then these subnetworks are connected together to form a complete network with maximum connectivity.

Applying this method to satellite networks, the space in which satellites orbit is partitioned into a number of regions of approximately the same size. Each region contains a subnetwork. For the sake of easy demonstration, the ensuing discussions assume that all satellites orbit in the same altitude. The multiple-altitude cases can be handled essentially in the same way. The size of a region is determined based on the principle that two satellites in a link are visible to each other all the time before the next reconfiguration occurs. The optimization of the region size will be discussed in detail in section 5.

To specify regions and satellites formally, a coordinate system (γ, θ, φ), where γ represents the altitude, θ the latitude, and φ the longitude, is adopted. Basically, the surface where satellites orbit is partitioned in such a way that the latitude is partitioned into twice as many intervals as is the longitude. For instance, if the longitude has x intervals, [φ0, φ1], [φ1, φ2], ..., [φx-1, φx], then the latitude has 2x intervals, [θ0, θ1], [θ1, θ2], ..., [θ2x-1, θ2x]. Note that, φx = φ0 and θ2x = θ0 because the surface is round. So the network is partitioned into 2x2 regions. Each region is represented by ([Ri, θi1], [φi1, φi2]). Since we intend to make the area of each region approximately the same, these intervals are not of equal width. For example, the widths of θ-intervals near two poles, as their two borders merge into poles, should be greater than those of intervals in other areas.

Let vi, v2, ..., vn denote all satellites in the network, and R1, R2, ..., Rn represent all regions partitioned from the spherical surface. A node vi at (θi, φi, γi) is in region Rk if θi ∈ [θk, θk+1) and vi ∈ [φk, φk+1) where φk(θ), φk(θ), φk(φ), and φk(φ) are the four borders of Ri. If two borders merge into poles, they are broken arbitrarily. Note that not every region has a northern or southern border. For instance, a region near a pole may lack a northern or southern border (in fact, the border reduces to a point), and the shape of a region in this area is much like a triangle on a spherical surface as illustrated in Figure 2.1.

![Figure 2.1 Regions and their connections near a pole.](image_url)

2.3 Reconfiguring Networks

Since a global timing scheme is used, each satellite will be running the same link assignment algorithm simultaneously to reconfigure the network. The synchronization of link establishment for the network is realizable as long as each node keeps a copy of the whole network graph. When every node is executing the same algorithm, a consensus can be reached as each node knows which edges should be discarded and which edges should be set up.

The links formed by a satellite are determined by the region it is in and by its location within the region at the time of reconfiguration. Between reconfiguration (link assignments), it is permissible for a satellite to cross the borders into the other regions. It is assumed that transceivers on each link will keep pointing/tracking each other once the link is set up, regardless of where the other satellite is. If the region size and the reconfiguration period are properly determined, links will not be broken when satellites cross the borders.

2.4 Scheduling Network Reconfigurations

As has been pointed out, the reconfiguration of the network is done periodically. The next question is how often should a network be reconfigured. Since each reconfiguration requires a temporary halting of message exchanges, to maximize network utilization the interval between two consecutive reconfigurations should be as long as possible. This is the principle that no links will be broken in between. Thus, the maximum reconfiguration period is the minimum of the longest time interval any two linked satellites can orbit and still maintain their common link. This can be determined after the region size has been decided. Figure 2.3 illustrates how the period is calculated.
that, for \((x_0, y_0) \in \{(x, y) : |x - y| = \text{the maximum distance that 2}\) satellites are visible to each other \). Then the time for a satellite to travel from \(x_1\) to \(x_2\) or from \(y_1\) to \(y_2\) is the longest period of reconfiguration for which no links will be broken. This can be computed as soon as the region size and the orbits of the satellites are determined.

3. The Hierarchical Approach for Reconfiguration

Before we describe the hierarchical approach, let's introduce some terms. As the network is partitioned into regions, some of the nodes have to set up links across the regions so that packets can go through these links to other regions. Those nodes having border-crossing links are called backbone nodes and the others are called internal nodes or non-backbone nodes. Similarly, those border-crossing links are named as backbone links and the others are internal links. The role each node plays is not fixed. It depends on the location of the node. In general, each region will select nodes closest to the 4 borders (eastern, southern, western, and northern borders) to be its backbone nodes. Backbone nodes play more important roles in routing packets because inter-region packets will be forwarded via at least one of these nodes. In each region, except for the one-node case as shown in Fig. 3.1.a, there are 6 backbone links; 4 links cross the northern and the southern borders; the other 2 links cross the eastern and the western borders. The distribution of links over borders is not unique. For instance, another alternative would be to put 4 links on the eastern and western borders and 2 links on the others. The basic principle ("single-hop" principle) is to allow the by-passing messages to be forwarded in one hop vertically as well as horizontally. The 6-backbones-links arrangement is derived partly from the 4-regular graph constraint and partly from empirical results of constructing primitive topologies for very small networks. There are possibly other arrangements with different numbers of backbone links in a region available to serve the purpose of obtaining regional maximum connectivity and of observing the single-hop principle as mentioned above.

The hierarchical link assignment algorithm can be divided into two levels: the first-level task is to determine a regional topology which includes establishing internal links and selecting backbone nodes. The second-level task is to combine regions into a connected network by setting up backbone links among backbone nodes of adjacent regions. The algorithm is designed to maximize regional connectivity as well as the whole network connectivity.

In the first-level task, regions are handled in two different ways: the primitive case and the recursive case. If a region contains less than or equal to 7 nodes, it is handled as the primitive case. Otherwise, it is regarded as the recursive case. The primitive case consists of 7 topologies. In the primitive case, the regional topology is predetermined and links are set up accordingly. In the recursive case, the link assignment algorithm works by "peeling off" layers of nodes; 4 nodes at a time, until it ends as one of 4 terminal cases; each layer contains the 4 nodes nearest to the 4 borders. The four terminal cases are shown in Figure 3.3.

3.1 Topologies for The Primitive Case

Seven primitive topologies are provided for the construction of regional subnetworks with less than 8 nodes. Figure 3.1 illustrates all of them. Note that these topologies are chosen to maximize the regional connectivity. By maximum regional connectivity, we actually mean that a node can construct 4 node-disjoint paths to any other node in the same region via the nodes in the same regions or in neighboring regions.

The primitive topologies are among the few topologies which meet the maximum connectivity requirement and the single-hop principle. The topologies have the following properties:

1. no links are redundant in the sense that every transceiver is used and no more than one link is assigned to any pair of nodes;
2. messages generated from a host inside a region can be forwarded efficiently out of this region via less than 3 intermediate nodes;
3. "straight-line hops", east-west or north-south message deliveries, are favored. There are backbone nodes in each case (e.g. \(v_1\)) and \(v_2\) in Fig. 3.1.b, or \(v_1\) and \(v_2\) in Fig. 3.1.c-3.1.g) that have 2 backbone links crossing east-west and north-south borders. Therefore, one-hop transmitting is allowed through this region. However, with the 4-link constraint, no similar "convenience" is offered if a by-passing message wants to "make a turn" through this region.

3.2 Topologies for The Recursive Case

If the number of nodes in a region is greater than 7, the strategy is to construct a topology recursively. The link assignment algorithm works in the following manner: First, 4 nodes, each of them the nearest node to a border, are chosen to form the first layer of the region. Note that the nodes in the first layer will also be the backbone nodes of the region, and will connect to backbone nodes in the neighboring regions. Second, the algorithm continues to peel off a set of 4-nearest-to-border nodes to form intermediate layers until one of the terminal cases is reached for which topologies exist. Figure 3.3 shows the topologies of all four permissible terminal cases. Finally, any two adjacent layers, \(i\) and \(i+1\), are connected by their corresponding 4 nodes (e.g. \(v_1^{(i)}\), \(v_2^{(i)}\), etc.) as shown in Figure 3.2.b where the \(v_j^{(i)}\)'s are of one layer above the \(v_j^{(i+1)}\)'s. Note that the property (3) in the primitive case is also preserved. In general, outgoing messages originating from a node of a particular layer are forwarded via the corresponding nodes of its upper layers all the way up to the corresponding backbone node and then transmitted to other regions.
3.3 Topologies for Inter-region Links

The inter-region backbone links are established in the second-level task. Only backbone nodes are involved in this task level. Generally, each region will have 4 backbone nodes. To illustrate how inter-region links are established, consider a region R, denote the backbone node closest to the west border by $u_1$, the backbone node closest to the south border by $u_2$, the backbone node closest to the east border by $u_3$, and the backbone node closest to the north border by $u_4$. Let $R_i, i \in \{1, 2, 3, 4\}$, be the neighboring region of R, e.g., let $R_1$ be the west neighboring region, and $R_2$ be the south neighboring region, etc.; and the backbone node closest to $R_i$'s west border is represented as $u_{i,1}$, etc. See examples in Figure 3.3.

The inter-region links are set up based on the following rules:

1. $u_1$ is connected to $u_{1,1}$, i.e., "horizontal links" are established (since the positions of the nodes change over time, the links are not strictly horizontal, but are so called for the sake of convenience).
2. $u_2$ is connected to $u_{2,2}$ and $u_{2,4}$, i.e., "vertical links" are established.
3. $u_3$ is linked to $u_{3,1}$ and $u_{3,3}$ is linked to $u_{3,2}$.
4. If the number of backbone nodes in R is less than 4, then some backbone nodes have to play the roles of more than one nodes.

For example, in Figure 3.3a R contains only 2 nodes, while in Figure 3.3b R contains at least 4 nodes. Therefore, $u_1$ and $u_2$ in Figure 3.3a are performing the roles of $u_1$ and $u_2$, and $u_3$ and $u_4$, respectively, in Figure 3.3b.

3.3 The Link Assignment Algorithm and Its Analysis

The link assignment algorithm consists of procedures REGIONLINK and NETWORKLINK. REGIONLINK handles the first level task and NETWORKLINK deals with the second level task. It is assumed that the space is partitioned and thus the region borders are determined before the network is initialized. Each node is associated with an ID. To execute the algorithm, each node should have the positional data, represented as (ID, $\theta$, $\phi$), of all the nodes. The data are obtained through calculations of orbits. In fact, the data can be precomputed, stored in each node, and accessed during reconfiguration. With this data, the memberships of the nodes (i.e., to which region a node belongs) can be determined during the reconfiguration, and they remain unchanged until the next reconfiguration. If a node is detected dead or missing, the information will be broadcast across the network and the node will be ignored for the reconfiguration. The procedures are described as follows:

Procedure REGIONLINK;
Input: a set of nodes of a region of the form (ID, (e, $\theta$, $\phi$))
Steps:
1. If $\#$ of nodes $< 4$, then construct the topology according to the arrangement of Figure 3.3a; return.
2. Construct 2 bi-directional queues from the nodes, $Q_+ \cup Q_-$; $Q_+$ is ordered non-decreasingly according to the distance between a node and the eastern border; $Q_-$ is ordered non-decreasingly according to the distance between a node and the northern border; ties are broken arbitrarily. /* recursive cases */
3. Remove the first and the last "unremoved" nodes from $Q_+ \cup Q_-$.
4. If $\#$ of "unremoved" nodes in each queue $> 7$, then go to 5.
5. Construct final layer topology as specified in Figure 3.3 (examples are shown in Figure 3.3).
end REGIONLINK.

Let R be the region in which the node executing the algorithm is located.

Procedure NETWORKLINK;
Input: the set of nodes of the network of the form (ID, (e, $\theta$, $\phi$))
Steps:
1. Call REGIONLINK 5 times to derive the regional topologies of $R_1, R_2, R_3, R_4, R_5$.
2. Construct inter-region topology according to the 4 rules specified in section 3.3 (examples are shown in Figure 3.3).
end NETWORKLINK.
The analysis of the time complexity of the link assignment algorithm is straightforward. Let $k$ be the number of nodes in a region. In REGION-LINK, step 1 and 5 take $O(1)$ (note that the process of checking whether a node is "removed" or "unremoved" in a queue takes $O(1)$) as it can be done by indexing an binary array $REM[k+VDE[k]]$. Step 2, mainly the sorting procedure, take $O((klogk))$. Step 3 and 4 serve as a loop. Since $4$ nodes are handled each time, the number of loopings is approximately $k/4$. Hence, step 3 and 4 jointly take $O(k)$. So, the total time complexity of REGION-LINK is $O((klogk))$.

In NETWORK-LINK, as each node is only concerned with the topologies of its $4$ neighboring regions and its own, the time complexity is still $O((klogk))$. Now, let us be the total number of nodes in the network and be the number of partitioned regions. It takes time $O(n)$ to determine the memberships of the nodes. By assuming that the number of nodes in each region is approximately $n/m$, it can be seen that NETWORK-LINK takes time $O((n/m)(log(n/m)))$ which is bounded by $O((klogk))$. So, the algorithm is completed in $O((klogk))$ time.

3.4 Applying The Algorithm

Given that there are $4$ transceivers in each node, the maximum node-connectivity of a network is no more than $4$. However, consider the case where there exists a region with only one node. In order to maintain a $4$-connected network, all the other regions of the same longitude must contain exactly one node. This is simply because there is only one backbone link crossing each of the northern and southern borders for the one-node regions, while there are two such links for the other cases (see Figure 3.1). Consequently, a good decision on the size of a region should manage to prevent the one-node-in-a-region case when a network starts its reconfiguration.

In fact, we will show that the link assignment algorithm maximizes the node-connectivity of a network if every region of the network contains at least $2$ nodes during each moment of reconfiguration. A natural way to achieve this is to increase the region size (i.e. to reduce the total number of regions) such that every region contains at least $2$ nodes all the time, which can be done by calculating satellites' orbits and adjusting region sizes accordingly.

The algorithm is most applicable when dealing with a large number of gateways, and not very suitable for small satellite networks in which the number of satellites is small, large region sizes are needed. But, if a region is too large, it will make two nodes within the region invisible to each other; this will make the algorithm fail.

In dealing with large satellite networks, the algorithm can handle the recursive case fairly efficiently. It can take advantage of the locality of nodes and come up with relatively short links (in a dynamic sense, of course, in order to handle networks in which the number of satellites is small, large region sizes are needed. But, if a region is too large, it will make two nodes within the region invisible to each other; this will make the algorithm fail.

4 Proofs on Network Connectivity

Connectivity is one of the important criteria in network topology design. A network should be able to remain connected even if some links or nodes fail. Before discussing the connectivity of a network, we formally define some terms.

Definition 4.1 Let $G=(V, E)$ be an undirected graph representing a network $N$. $V$ is the set of nodes in $N$ and $E$ is the set of edges/links in $N$. A path from $x$ to $y$ is a sequence of edges of the form $<v_i, v_{i+1}, \ldots, v_j>$ where $v_{i-1}, v_i$ is an edge for $1 \leq i \leq j$, and $x = v_0, y = v_j$. Two paths from $v_i$ to $v_j$ are node-disjoint if they contain no common nodes except the two end nodes $v_i$ and $v_j$. The node-connectivity of a pair of nodes is the minimum number of nodes whose removals will disconnect the two nodes. Equivalently, it can be defined as the maximum number of node-disjoint paths between them. The node connectivity of a network is defined to be the minimum node-connectivity of all pairs of nodes in $V$. There is another connectivity for a graph, called edge-connectivity which can be defined as the minimum number of edges whose removals will disconnect a graph. Since node-connectivity is a stronger condition than edge-connectivity in terms of reliability, node-connectivity is adopted here throughout this paper.

Given that each node has at most $4$ links in the network, we will show that the link assignment algorithm NETWORK-LINK yields a surprisingly good connectivity in satellite network topology design.

Definition 4.2 Two regions $R_i$ and $R_j$ are adjacent if $R_i$ is contiguous to $R_j$, or if they intersect in a pole and are in radially symmetric positions with respect to the pole. A region path between two regions is a sequence of regions of the form $<R_1, R_2, R_3, R_4, \ldots, R_{i-1}, R_i, R_{i+1}, R_{i+2}, \ldots, R_n>$ where $R_k$ is adjacent to $R_{k+1}$, $2 \leq k \leq n-1$. Two region paths from $R_i$ to $R_j$ are region-disjoint if they have no regions in common except $R_i$ and $R_j$. The proof is straightforward. We can show that by explicitly setting up $4$ region-disjoint paths between any pair of regions.

There are two possible cases to be discussed:

Case 1. $R_i$ is adjacent to $R_{i+1}$. It is trivially true (see Figure 4.1a for example).

Case 2. $R_i$ is nonadjacent to $R_{i+1}$. Without loss of generality, suppose that $i < j$ as shown in Figure 4.1b. Construct a path $P_1$ as $<R_{i+1}, R_{i+2}, \ldots, R_j>$. Construct the second path $P_2$ as $<R_i, R_{i+1}, R_{i+2}, \ldots, R_j>$. Construct the third path $P_3$ as $<R_i, R_{i+1}, \ldots, R_j>$. Construct $P_4$ as $<R_i, R_{i+1}, \ldots, R_j>$. It is easy to observe that $P_1, P_2, P_3, P_4$ are region-disjoint.

The other cases such as $i \geq j$ can be demonstrated similarly and thus are omitted here.

Each time when REGION-LINK is completed, an intra-region topology is constructed with layers of nodes. Except for the final layer which has as many $7$ nodes, each layer consists of $4$ nodes of the form $(v_i, v_{i+1}, v_{i+2}, v_{i+3})$. For convenience we may call $(v_i, v_{i+1}, v_{i+2}, v_{i+3})$ layer $i$. In particular, layer $1$ is the backbone layer, and also denoted as $(v_0, v_1, v_2, v_3)$. Notice that the links $(v_i, v_{i+4})$, $(v_i, v_{i+5})$, $(v_{i+1}, v_{i+6})$ and $(v_{i+2}, v_{i+7})$ for all $1 \leq i \leq 4$ and $j \geq 1$, will be assigned in REGION-LINK.

Lemma 4.2 There exist disjoint paths from any node $x$ to the other $3$ nodes in the same layer $i$ such that these $3$ paths consist of nodes only in layer $i$, layer $i+1$, or outside the region.

Proof. There are three cases to be discussed.

Case 1. $x$ is in the backbone layer. Without loss of generality, suppose $x = v_i$. Let $v_0, v_1$ denote the node $v_i$ in region $X(i)$, where $i \in \{1, 2, 3, 4\}$ and $X(i) \in \{W, NW, N, NE, E, SE, S, SW\}$. Then the $3$ disjoint paths are: $<v_0, v_2>$, $<v_0, v_3, v_4>$, $<v_0, v_5, v_6, v_7>$.
Lemma 4.3 There are disjoint paths from any node \( z \) to the other 3 nodes in the same layer \( i \), where layer \( i \) is an intermediate layer, such that these 3 paths consists of only nodes in layer \( i \) or \( i+1 \).

Proof. Let \( z \) be \( v_i^{(0)} \). Two obvious disjoint paths are \( <v_i^{(0)}, v_i^{(1)}> \) and \( <v_i^{(2)}, v_i^{(3)}> \). The third path, from \( v_i^{(0)} \) to \( v_i^{(1)} \), can be set up via node \( v_{i+1}^{(0)} \), i.e. \( <v_{i+1}^{(0)}, v_{i+1}^{(1)}, v_{i+1}^{(2)}, v_i^{(1)}> \). Since the link assignment in the intermediate layer is symmetric, it holds for the cases where \( z \in \{v_i^{(0)}, v_i^{(2)}, v_i^{(3)}\} \).

Lemma 4.4 There are 4 disjoint paths between every pair of nodes \((x,y)\) where \( x \) and \( y \) are in different layers.

Proof. Without loss of generality, suppose \( x \) is in layer \( j \) and \( y \) is in layer \( j < i \). From Lemma 4.2 we know that there exist 3 disjoint paths from \( x \) to the other three nodes in layer \( i \) by using nodes only in layer \( i \) or above, which are possibly nodes in neighboring regions. Now, since there is an edge \( \langle v_i^{(k)}, v_i^{(k+1)} \rangle \) for all \( k \in \{1, 2, 3, 4\} \) and \( j \) is either a backbone layer or an intermediate layer, 4 paths, \( <v_i^{(k)}, v_{i+1}^{(k)}, v_{i+2}^{(k)}, v_{i+3}^{(k)}>, \) can be constructed. Hence, we are able to build up 4 disjoint paths from \( x \) to \( v_i^{(k)} \) for \( k = 1, 2, 3, 4 \).

Lemma 4.5 The connectivity of every pair of nodes in the same layer \( i \) is 4.

Proof. It suffices to prove that, for each pair of nodes in the same layer, there exist \( 4 \) disjoint paths between them. This can be done by path enumeration. Since it will be a lengthy proof if we enumerate all the 4 disjoint paths for every pair of nodes, some typical cases are illustrated to show that the disjoint paths can be constructed explicitly. The rest of the cases can be done similarly.

case 1. \( i \) is the backbone layer (\( i = 1 \)). Let us show that, for \( \{v_i^{(0)}, v_i^{(1)}\} \), we can explicitly set up 4 disjoint paths connecting them as follows: The first path is \( <v_i^{(0)}, v_i^{(1)}, v_{i+1}^{(2)}, v_{i+2}^{(3)}> \). Now, via its west, north-west and north neighboring region, \( v_{i+1}^{(2)} \) can be reached. Thus, a second path can be established. Third, via east, south-east and south neighboring regions, the third path can be set up. Finally, the last path can visit \( v_{i+2}^{(3)}, v_{i+3}^{(3)}, v_{i+3}^{(4)} \), and then \( v_i^{(4)} \) (see Figure 4.5.a). It is easy to verify that there exist 4 edge-disjoint paths between any other pair in the backbone layer.

case 2. \( i \) is the final layer (\( i = L \), see Figure 3.3). If the number of nodes in the final layer is 4 (as shown in Figure 3.3.a), every pair of nodes is directly connected, so the node-connectivity is a maximum. If the number of nodes in the final layer is 5 (as shown in Figure 3.3.b), then, for \( \{v_i^{(0)}, v_i^{(1)}\} \), Figure 3.3.b, the 3 disjoint paths between \( \{v_i^{(0)}, v_i^{(1)}\} \) are: \( <v_i^{(0)}, v_i^{(1)}, v_i^{(2)}, v_i^{(3)}, v_i^{(4)}>, <v_i^{(0)}, v_i^{(1)}, v_i^{(2)}, v_i^{(3)}, v_{i+1}^{(4)}>, \) and \( <v_i^{(0)}, v_i^{(1)}, v_i^{(2)}, v_i^{(3)}, v_{i+1}^{(4)}> \). The fourth path will pass through \( v_i^{(4)} \) and \( v_{i+1}^{(4)} \) to \( v_i^{(0)} \). Following the same enumerative process, we can verify that it also holds for the 8-node case (as shown in Figure 3.3.c) and the 7-node case (as shown in Figure 3.3.d).

case 3. \( i \) is an intermediate layer. As Figure 3.2.b shows, we can choose any pair, say \( \{v_i^{(0)}, v_i^{(1)}\} \), and construct the following 4 disjoint paths between them: \( <v_i^{(0)}, v_i^{(1)}, v_i^{(2)}, v_i^{(3)}, v_{i+1}^{(4)}>, <v_i^{(0)}, v_i^{(1)}, v_i^{(2)}, v_i^{(3)}, v_{i+1}^{(4)}>, \) and \( <v_i^{(0)}, v_i^{(1)}, v_i^{(2)}, v_{i+1}^{(3)}>, <v_i^{(0)}, v_i^{(1)}, v_{i+1}^{(3)}, v_i^{(2)}> \). The other cases for an intermediate layer are similar.

Lemma 4.6 Let \( N = (V, E) \) be the network topology derived from applying the NETWORK-LINK algorithm; and let \( Vl, V_2, \ldots, V_7 \) be the set of the nodes of the regions where each \( V_l \) resides in region \( R_l \).

Theorem 4.1 If \( |V_l| \geq 2 \) for all \( V_l \), then the network \( N \) has connectivity \( = 4 \).

Proof. Let \( v_l \) and \( v_j \) be two nodes in \( V_l \).

case 1. \( v_l \) and \( v_j \) belong to the same region \( R_l \). If \( R_l \) contains less than 8 nodes, i.e. a primitive case (Figure 3.1.b to Figure 3.1.f), we can enumerate all the 4 disjoint paths between any two nodes. These paths will visit 4 adjacent regions of \( R_l \) (i.e. \( N(R_l), S(R_l), W(R_l), \) and \( E(R_l) \)) and 4 non-adjacent regions (i.e. \( NW(R_l), NE(R_l), SE(R_l), \) and \( SW(R_l) \)). For the sake of brevity, they are not enumerated here (a similar example is shown in Figure 4.5.a). If \( R_l \) contains more than 7 nodes, it follows immediately from Lemma 4.4 and Lemma 4.5 that 4 disjoint paths exist for each pair.

case 2. \( v_l \) and \( v_j \) are not in the same region, (e.g. \( v_l \) is in \( R_l \) and \( v_j \) is in \( R_j \)). If \( R_l \) is reconfigured as a primitive topology, it is easy to verify that there exist 4 disjoint paths from any node in any of the 7 primitive cases to its 4 adjacent neighboring regions. If \( R_l \) contains more than 7 nodes, then as we have shown in the proof of Lemma 4.2 and Lemma 4.3, there are 4 disjoint paths from \( v_l \) to the four backbone nodes of \( R_l \) and each backbone node has an edge to its neighboring regions. Thus, regardless of the number of nodes \( R_l \) gets, there are always 4 disjoint paths from \( v_l \) to its 4 neighboring regions. Similarly, there are always 4 disjoint paths from \( v_j \) to \( R_j \) to its 4 neighboring regions. Now, by applying Lemma 4.1 we can construct 4 disjoint region-paths from \( v_l \) to \( v_j \) of its neighboring regions. Finally, via these disjoint region-paths 4 disjoint paths from \( v_l \) to \( v_j \) can be successfully established.

5. Conclusions

We present a hierarchical, partitioning approach to the link assignment problem of large-scale satellite networks. By partitioning a satellite network space into regions of approximately equal size, the link assignment problem of the network can be reduced to a number of regional link assignment problems which, due to their greater simplicity, can be solved recursively by connecting satellites in a layer-by-layer manner. The algorithm proves to be superior to other methods in terms of time complexity. It is also shown that, in the large-scale networks where each region contains at least two nodes, the node-connectivity of the network is maximized when each node is restricted to have 4 transceivers. With this property, packets routing can be more flexible, and the possibility of packet congestion can be effectively reduced.
Figure 4.1 Disjoint Region-paths

Figure 4.2

Figure 4.3

Figure 4.4
Figure 4.5

References
