Chapter 4

Protocols for Isochronet Operations

4.1 Introduction

4.1.1 The Problem

This chapter defines the main protocols necessary for Isochronet operations. The protocols deal with three main issues: (1) how to allocate trees in the networks, (2) how to assign bands to trees, and (3) how to synchronize bands at different nodes.

Tree allocation. In this problem, the objective is to allocate routing trees in the network so as to satisfy some optimization criteria. The exact criterion is dependent on the particular goals that the network must accomplish. For example, trees can be allocated to minimize the distance between sources and destinations. This strategy is particularly useful when links are operating at high speeds because the propagation delay is a major component in the total end-to-end delay. Another example particular to Isochronets is to allocate trees such that the number of bands necessary to cover all possible destinations is minimized. This goal can be accomplished by allocating as many non-interfering trees as possible in the same band. The tree allocation mechanism should be geared toward choosing

as many trees as possible that do not share links in the same direction.

Band synchronization. Given the band allocations in each node, the objective is to synchronize the bands, that is, assure that frames transmitted from a given node during band B_i reach the next node in the tree within the same band B_i. Synchronization mismatches in Isochronets may result in frames being misrouted. For example, if a frame transmitted during band B_i reaches an intermediate node during another band B_j, frames are forwarded from that point on toward B_j's destination. Another concern in band synchronization is to avoid bands intercepting each other. For example, if the intersection between bands B_i and B_j at some node is not empty, routing during the overlap period is ambiguous because incoming frames could potentially follow two separate paths.

Band allocation. Given a set of band transmission requirements, the goal is to allocate bands that will fit the requirements while being synchronized and not interfering with each other. The band allocation is thus concerned in finding two parameters for each band: initiation time within the clock cycle and size. The allocation changes over time given new demands computed from past history.

These problems (and their generalizations) could per se become a very intricate study. They are not the main subject of this thesis and more elaborate solutions are left for future work. This chapter gives a preliminary solution to each problem.

4.1.2 Main Results

Tree allocation. The solution proposed is an exhaustive search. This solution is acceptable in this case because the tree allocation protocol is executed only when the net-

work configuration changes.

Band synchronization. The proposed solution operates by manipulating the link propagation delays in the network. It involves a special device placed at each switch outgoing port, that is, at the start of each network link. The device can be set to increase the link delay. A distributed protocol computes each link delay and then increases it using the special device to make the final delay a multiple of the cycle period. Because of this manipulation, if a frame is sent at time t within the cycle from a given node, it reaches the next node in the path still at time t but many cycles later. This fact trivializes synchronization because all that is necessary in this scenario is to start and end each band exactly at the same moment within each cycle.

Band allocation. For the purpose of this protocol, bands are always synchronized using the band synchronization protocol. The main concern of the band allocation protocol is then to size the bands. The task is pursued using a dynamic algorithm based on the Least Recently Used (LRU) memory allocation policy. The protocol operates by shrinking band sizes for destinations that are not accessed frequently and then increasing band sizes of destinations that need higher demands.

All of the discussed protocols can be implemented off-line, that is, they may execute at slow speed relative to transmissions.

4.1.3 Chapter Organization

This chapter is organized as follows. Section 4.2 discusses the mathematical model that will be used to formally define each problem. The tree allocation problem, the band synchronization problem, and the band allocation problem are discussed in Sections 4.3,

4.4, and 4.5, respectively. Section 4.6 overviews future work in each of these problems. Finally, Section 4.7 concludes.

4.2 The Model

This section described the mathematical model used to formulate the problems in the next sections. A network is viewed as a directed graph [Behzad et al. 79] $G = \langle V_G, E_G \rangle$, where V_G is the set of nodes and E_G is the set of edges. Each edge e has positive real-valued capacity $c(e)$ and propagation delay $d(e)$. Edges are also denoted by an ordered pair $e = (e_s, e_d)$ where e_s is the source and e_d is the destination of the edge. The following property must hold in G: $\forall u, v \in V_G \cdot (u, v) \in E_G \Rightarrow (v, u) \in E_G$. That is, only edges in both directions connect pairs of nodes.

A spanning tree $T = \langle V_T, E_T \rangle$ is a connected acyclic subgraph of G where $V_T = V_G$, $E_T \subseteq E_G$. Of particular interest are spanning trees that have a distinguished node r that can be reached from all other nodes. Such tree is a *routing tree* and node r is the root or destination of the tree. The tree with root r is labeled T_r.

Another class of trees of interest is spanning trees with a distinguished source node *s* that has a path in the tree to all other nodes. These are the *multicast trees*.

A clock is associated with each node in the network. The clock cycle period ranges from 0 to C. A band for a set of disjoint trees Γ at node n is an interval $B_n(\Gamma) = [b_n(\Gamma), e_n(\Gamma)]$, where $0 \le b_n(\Gamma) \le e_n(\Gamma) \le C$.

4.3 Tree Allocation

The main concern of the tree allocation problem is to build spanning trees in the network satisfying some optimization criteria. As discussed in Section 4.1, one such criteria would be to minimize the distance between nodes. This is a useful approach especially in HSNs where the propagation delay plays a major role in the end-to-end delay. Minimizing the number of links or, alternatively, the physical distance between nodes by choosing minimal distance paths will significantly decrease the total end-to-end delay. For example, when transmissions occur at 2.4 Gb/s, an ATM cell takes only 177 ns to be transmitted. The direct United States cross-country propagation delay is about 30 ms. It is easy to see that, if tree allocations are such that cross-country communication crosses a few intermediate nodes possibly with some nodes not geographically located in the right direction, the end-to-end total delay may be increased by a few milliseconds. It is thus important to minimize the number of hops or the physical distance in this scenario. Trees can be allocated with respect to this optimization criterion using one of the standard techniques to find minimal distance routes in PS networks [Comer 91, Halsall 92].

This section exploits an interesting novel minimization criterion particular to Isochronets. According to this criterion, the total number of bands to cover all possible destinations is the variable to be minimized. The path to accomplish this minimization criterion is to fit as many trees as possible in the same band. Two trees can inhabit the same band as long as their paths do not interfere, that is, as long as they do not share a link in the same direction. Figure 3.6 (Chapter 3) illustrates an example of two such trees.

The formal definition of the tree allocation problem is given in Problem 4.1.

Problem 4.1: Tree allocation.

Given: A network G.

Find: A set Θ of $|V_G|$ routing trees.

Satisfying: $\forall n \in V_G \cdot \exists T_n \in \Theta$.

Minimizing: $r(\Theta)$, where $r(\Theta)$ is the minimal number of subsets of Θ such that two elements of Θ are in the same subset if and only if they do not share any link in the same direction and the union of all the subsets is equal to Θ.

Problem 4.1 states that, given a generic network, the goal is to find a routing tree for each node (all nodes are possible destinations). The set of routing trees found must satisfy the following minimization criterion. It must be possible to subdivide the set in subsets of disjoint trees, that is, trees that do not share any link in the same direction. Furthermore, the number of such subsets must be minimal and the union of the subsets must be equal to the original set of routing trees. Notice that each subset may be assigned to the same band. Consequently, by minimizing the number of such subsets, the number of bands is minimized.

This section proposes a solution that uses an exhaustive search algorithm. It is an acceptable solution in this case because the problem can be solved off-line, before the system starts operating. It is assumed that links or nodes fail rarely. In such cases, the tree allocations need to be re-computed. Another alternative is to pre-compute tree allocations for all possible configurations of node or link failures and store them in a table for future look-ups.

Another compelling argument for exhaustive search algorithms is that Isochronets

many times operate on networks with few nodes. If the network size is large, a cluster technique outlined in Chapter 8 is used to build Isochronets within and among clusters. The goal of the clustering technique is to create a hierarchical solution such that the number of clusters and the nodes within clusters is kept small.

Protocol 4.1: *Generates an optimal solution for Problem 4.1 using an exhaustive search approach. Given a network G, the protocol finds the minimal number of bands that will cover all destinations.*

Generate all routing trees in *G*, building the set *R* of all possible routing trees.

1. Generate all maximal subsets of *R* such that two trees in *R* are in the same subset if and only if they do not share any link in the same direction. The resulting set of maximal subsets of *R* is *S*.
2. Select a minimum number of elements in *S* such that the union of all selected elements has one routing tree per network node.
3. If the elements of *S* selected in Step 3 have more than one tree for any destination, eliminate redundant trees so that the final solution contains only one tree per destination.
4. Distributed the allocation among the nodes in the network.

The exhaustive search algorithm works as sketched in Protocol 4.1. The algorithm is executed at a central site where the allocation is computed based on the network topology. Later, the allocation is distributed among the nodes in the network. Another option for reliability is to replicate the algorithm in a few sites that can run it in tandem.

Step 1 generates all possible routing tree in *G*. Step 2 seeks to generate all possible

sets of routing trees that do not interfere, that is, that do not share a link in the same direction. The generated sets are maximal, that is, no routing tree can be added to the set without violating the non interference criteria. Notice that each maximal set contains trees that can be allocated in the same band. Step 3 finds the minimal number of maximal sets whose union contains a routing tree for each node in the network. In other words, Step 3 effectively finds the minimal number of bands that can cover all possible destinations. Step 4 removes redundant trees. The solution directly satisfies the minimization criteria in Problem 4.1.

Algorithm 4.1: *Generates all possible routing trees of a network G. The number of nodes in G is n.*

1. Generate all possible sets of n-1 links in G. The resulting set is L.
2. Generate a new set L' from L such that L' is a subset of L and all elements of L' constitute a valid routing tree. L' contains all possible routing trees of G.

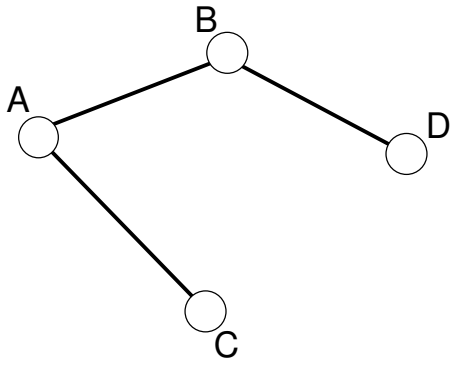

Figure 4.1: Example network topology

The implementation of Protocol 4.1 should not present difficulties, except for Step 1 that is solved in Algorithm 4.1. It operates by generating all possible sets of n-1 links of G

and then filtering the ones that do not constitute a valid routing tree.

Example 4.1: *Allocates routing trees in the example four node network depicted in Figure 4.1. The routing trees allocated minimize the number of necessary bands to cover all possible bands in the network.*

Consider the network *G* in Figure 4.1. Let us begin by generating all possible routing trees according to Algorithm 4.1. Notice that the links are (A,B), (B,A), (A,C), (C,A), (B,D), (D,B). Step 1 generates $\binom{6}{3} = 20$ sets of 3 links in *G*. For example, {(C,A), (A,B), (B,D)} is a valid tree while {(A,B), (A,C), (B,D)} is not. The following are the valid routing trees after the filtering phase in Step 2:

- {(A,B), (C,A), (B,D)};
- {(A,B), (C,A), (D,B)};
- {(B,A), (A,C), (D,B)}; and
- {(B,A), (C,A), (D,B)}.

Protocol 4.1 generates in Step 2 the following maximal subsets:

- {{(A,B), (C,A), (B,D)}, {(B,A), (A,C), (D,B)}};
- {{(A,B), (C,A), (D,B)}}; and
- {{(B,A), (C,A), (D,B)}}

Usually many possibilities will exist in Step 3, unlike the situation in this example in which the network contains only one path from each source to each destination. The solution for the example is to assign one band per maximal set, resulting in tree bands.

Example 4.1 illustrates how these protocols can be used to allocate routing trees in a

four node network.

There are a few directions that can be used to generalize Protocol 4.1. The first one is to allocate not only routing trees, but also multicast trees according to demands. The most complete scenario would be to allocate both a routing and a multicast tree per node. Toward this goal, Step 1 in Protocol 4.1 should generate all possible routing and multicast trees. Step 2 in Algorithm 4.1 can be tuned to generate all multicast trees. Step 2 in Protocol 4.1 needs no modification. Steps 3 and 4 need to be extended to cover all necessary routing and multicast trees.

Another interesting generalization of Protocol 4.1 is to fit trees with similar loads in the same band. That is, two trees are placed in the same band if their loads are similar. Otherwise, some of the trees would not fully use the band. Allocating bands taking into account the load of the trees would minimize bandwidth wasted due to load imbalances among trees sharing the same band. This problem is left for future investigation.

The complexity of the protocols discussed increases rapidly with the number of bi-directional links in the network. Algorithm 4.1 in Step 1 generates $\binom{2l}{n-1}$ possibilities, where $n-1 \le l \le n^2$ is the number of links in the network. Step 2 in Protocol 4.1 is in the worst case of size $\binom{2l}{n}^{\binom{2l}{n}}$. The other steps can be executed within this complexity.

As pointed out previously, Isochronets are designed to operate on networks with few nodes that are then interconnected to other networks. This fact and the fact that the tree allocation algorithm is run once before the network begins operating make the perform-

ance of this solution acceptable.

4.4 Synchronization

There are two kinds of synchronization necessary for Isochronets: clock synchronization and band synchronization. Any of the traditional protocols such as the Network Time Protocol (NTP) [Mills 91] may be used to solve the clock synchronization problem. This section defines and provides a solution for the band synchronization problem.

Band synchronization must ascertain the following constraints:

- ***Band Constraint.*** For each path from a source to a given destination, the corresponding band at each node must be strictly contained, when propagation delay is added, within the same band at the next node.
- ***Overlap Constraint.*** The bands of different trees on the same node do not intersect.

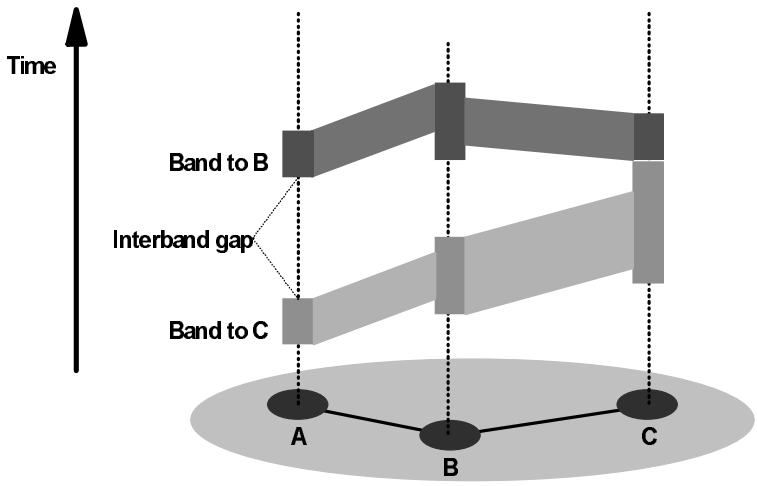

Figure 4.2: Band and overlap constraints

The goal of band synchronization is to establish band initiation values within the clock

cycle that satisfy both the band constraint and the overlap constraint at each node. Figure 4.2 illustrates both constraints. The graph above illustrates the band allocation (boxes) at each node in the network underneath. The Band Constraint is illustrated by the fact that each band is contained, when propagation delay is added, within the same band in the next node toward the destination. The Overlap Constraint means that boxes cannot intersect. Notice that this constraint forced the inter-band gap at nodes A and B.

The network links are assumed to contain special hardware devices or *delay element* that can be tuned to increase the link delay by a certain positive value between zero and the cycle period. The delay element can be tuned by the source switching node incident to the link. It can be implemented as a buffer at the node output, as will be explained in Chapter 6.

Formally, the propagation delay in each link is an element of a group [Suzuki 77]. The domain of the group is the set of all real number s (or *shifts*) in the interval $0 \le s \le C$, where C is the clock cycle size at each link. The group contains the operation sum modulo C that is denoted by "·". The reason why the propagation delay can be abstracted using this group is that the value of interest for the synchronization problem is at what point *within the cycle* a transmitted frame reaches the destination rather than the total propagation delay. Given an edge *e*, the shift on *e* is denoted by s_e. The "·" operation on bands is defined by $\left[b_n(\Gamma), e_n(\Gamma)\right] \cdot \left[s_{(n,m)}, s_{(n,m)}\right] = \left[b_n(\Gamma) \cdot s_{(n,m)}, e_n(\Gamma) \cdot s_{(n,m)}\right]$. The formal definition of the synchronization problem is the following.

Problem 4.2 states that, given a network, a collection Φ that contains sets of trees that participate in the same band (that is, trees that do not interfere), the goal is to find:

1. For each node in the network, and for each set of non-interfering trees, the initiation and termination times of the band.
2. Delays in each link in the network that participates in some tree.

Problem 4.2: *Band synchronization.*

Given: A network G and a collection Φ of sets Γ of non-interfering spanning trees of G.

Find: For each node $n \in V_G$ and each $\Gamma \in \Phi$, a band $B_n(\Gamma)$.

For each edge e that occurs in elements of Φ, a shift s_e.

Satisfying: For each $\Gamma \in \Phi$ and each directed edge (n,m) that occurs in elements of Γ, $\left[b_n(\Gamma), e_n(\Gamma)\right] \cdot \left[s_{(n,m)}, s_{(n,m)}\right] \subseteq \left[b_m(\Gamma), e_m(\Gamma)\right]$.

For each node $n \in V_G$, $\left[b_n(\Gamma), e_n(\Gamma)\right] \cap \left[b_n(\Gamma'), e_n(\Gamma')\right] = \varnothing$, if $\Gamma \neq \Gamma'$.

Minimizing:
$$L = \sum_{\Gamma \in \Phi,\,(n,m) \in E_T,\,T \in \Gamma} \left[\left(e_m(\Gamma) - e_n(\Gamma) \cdot s_{(n,m)}\right) + \left(b_n(\Gamma) \cdot s_{(n,m)} - b_m(\Gamma)\right)\right] + \sum_{n \in V_T} \left[C - \sum_{\Gamma \in \Phi} \left(e_n(\Gamma) - b_n(\Gamma)\right)\right].$$

Solutions are restricted so as to satisfy the band and overlap constraints. The minimization criterion is to avoid wasting bandwidth. It has two parts. The first computes how much bandwidth is wasted due to band shifting by link propagation delays. It is computed as the summation of the differences between the ending times and the difference between the starting times of the same band at the two nodes of each link. Ideally, there should be no such waste if the bands are synchronized. That is, each band should be allocated such that, when a given band at the start of a link is shifted by the delay in the link, it exactly matches the same band at the end of the link.

The second component adds the difference at each node between the cycle period and the sum of all bands. It computes how much of the cycle is not allocated to a band. The ideal allocation would make such waste 0.

This formula is a measure of bandwidth waste in terms of band offsets at the source and destination of each link and gaps between bands. The offset is illustrated in Figure 4.2. For example, the band to C at A and B are of different sizes and thus introduce some wasted bandwidth. The inter-gap between bands is also illustrated in the figure. The gap is sometimes necessary in order to satisfy the overlap constraint.

The following optimal solution is proposed: add delays to the links so as to make the resulting link delays equal to 0 modulo C and make each band initiation and termination times the same in all the nodes. It is easy to verify that, in this case, $L = 0$. Whenever a new band is allocated (see next section), the beginning and ending times for each band are set to the same value for all the nodes in the network.

The only open question is how to make the final delay 0 modulo C for all links in the network. Protocol 4.2 ensures this property. The idea is to use the group property of existence of an inverse element for each link shift. The inverse element is added to the link delay, making the total link delay become 0 modulo C.

In Step 1, node A sends a request to B to find out the delay in the link between them. In Step 2, B time-stamps the reply and then transmits it. In Step 3, A receives the reply and marks the arrival time. A is now in position to compute the offset between A and B. The offset is the propagation delay between the nodes modulo the cycle period. Using the computed offset, A then updates its delay element in the link between A and B in Step 4.

If the delay element was set with a value larger than the offset computed, the latter is simply subtracted from the contents of the delay element making the total end-to-end delay 0 modulo the cycle period. Otherwise, the contents of the delay elements are incremented with the value necessary to make the total delay between A and B a multiple of the cycle period.

Protocol 4.2: *Sets the delay at each link to 0. Given two nodes A and B, the protocol sets the delay in the link (A,B) to 0 modulo* C. *The delay introduced by the delay element at the output of A to B is d*(A,*B*).

1. A->B: Request For Delay (RFD) message for link (A,B).
2. B->A: Delay Response (DR); B marks time *T* at which DR is sent.
3. A marks arrival time *R* of DR. A measures the offset *O*=(*R*-*T*) modulo *C*.
4. If *d*(A,B) > *O*, set *d*(A,B) to *d*(A,B)-*O*. Otherwise, set *d*(A,B) to *d*(A,B)+*C*-*O*.

Example 4.2 illustrates some issues in Protocol 4.2.

Example 4.2: *Computes the incremental delay that needs to be added to the delay element in an example link (A,B). The propagation delay between the links is 10.732 ms and the cycle period is 125* μ*s.*

Any frame sent by A at time *t* will reach B at time *t*+107 μs (modulo the cycle period) 85 cycles later. The idea in Protocol 4.2 is to add 18 μs to the propagation delay between A and B so that now any frame sent by A at time *t* will reach B at time *t*, but 86 cycles later.

4.5 Band Allocation

The goal of band allocation protocols is to establish appropriate band duration given

minimal required tree loads. The loads are expressed in terms of band sizes. The allocation must satisfy the band and the overlap constraints. Band allocation is formally defined as follows.

Problem 4: *Band allocation.*

Given: A network G, a collection Φ of sets Γ of non-interfering spanning trees of G, and a tree load δ_Γ for each $\Gamma \in \Phi$.

Find: For each node $n \in V_G$ and each $\Gamma \in \Phi$, a band $B_n(\Gamma)$.

For each edge e that occurs in elements of Φ, a shift s_e.

Satisfying: For each $\Gamma \in \Phi$ and each directed edge (n,m) that occurs in elements of Γ, $\left[b_n(\Gamma), e_n(\Gamma)\right] \cdot \left[s_{(n,m)}, s_{(n,m)}\right] \subseteq \left[b_m(\Gamma), e_m(\Gamma)\right]$.

For each node $n \in V_G$, $\left[b_n(\Gamma), e_n(\Gamma)\right] \cap \left[b_n(\Gamma'), e_n(\Gamma')\right] = \varnothing$, if $\Gamma \neq \Gamma'$.

For each node $n \in V_G$ and each $\Gamma \in \Phi$, $e_n(\Gamma) - b_n(\Gamma) \geq \delta_\Gamma$.

Minimizing:
$$L = \sum_{\Gamma \in \Phi,\,(n,m) \in E_T,\,T \in \Gamma} \left[\left(e_m(\Gamma) - e_n(\Gamma) \cdot s_{(n,m)}\right) + \left(b_n(\Gamma) \cdot s_{(n,m)} - b_m(\Gamma)\right)\right] + \sum_{n \in V_T} \left[C - \sum_{\Gamma \in \Phi} \left(e_n(\Gamma) - b_n(\Gamma)\right)\right].$$

All the nodes must know what is the position of each band within the cycle because all the trees are spanning. Thus, in order to allocate bands, it is necessary to communicate the allocation to all the nodes in the network. Notice that the problem definition is similar to the synchronization problem, except that now band sizes must be at least as big as the required tree load.

The band allocation problem can be solved in a manner similar to band synchroniza-

tion. By setting the link delays to 0 modulo the cycle period and the band initiation and termination values to be the same at each node, $L = 0$. To complete band allocation, it is necessary to set what the band initiation and termination times should be. The value can be computed by allocating sizes that can fit minimal requested loads.

Problem 4.3 can be generalized to dynamic load demands as follows. The minimal band sizes now changed periodically based on source load demands. The synchronization solution can still be adopted in this case. Nevertheless, the process of sizing the bands becomes dynamic rather than static. A possible solution to the dynamic band sizing problem is to allocate bands according to a memory management algorithm such as Least Recently Used (LRU). Protocol 4.3 is an example of such an approach.

Protocol 4.3: *Adapts the band sizes according to the LRU memory management algorithm. Given b bands, each of size $s[i]$, $1 \le i \le b$, and a cycle of size C, the protocol adapts the band sizes according to the traffic demands $d[i]$, $1 \le i \le b$, where $d[i]$ is the highest demand of all trees sharing the band. The resulting new band sizes are $s'[i]$.*

1. Find all bands i such that $d[i] < s[i]$. Make $s'[i] = d[i]$ for those bands and compute *LOB,* the LeftOver Bandwidth (that is, summation of $s[i] - d[i]$ for all bands such that $d[i] < s[i]$).
2. Find all bands i such that $d[i] > s[i]$. Compute *NB*, the Needed Bandwidth (that is, summation of $d[i] - s[i]$ for all bands such that $d[i] > s[i]$).
3. Distribute *LOB* among the bands i such that $d[i] > s[i]$. Make $s'[i] = s[i] + LOB*(d[i] - s[i])/NB$.
4. Distribute the band allocation among all nodes in the network.

Step 1 searches all bands such that the traffic demand is smaller than the allocated band size. These are shrunk into the size requested and the left over band is computed to be later distributed among the more demanding destinations. Step 2 searches for the bands that have more demand than the allocated bandwidth. The total needed band is computed. Step 3 distributes the left over band computed in Step 1 among the needing destinations according to their demands.

Protocol 4.4 can be extended to include filters, as in traditional LRU algorithms, to avoid oscillations in band allocations.

4.6 Future Work

As stated in the introduction, a complete study of these problems can be quite complex and is left for future endeavors. The goal of this section is to state directions that can be pursued.

4.6.1 Tree Allocation

The algorithm for tree allocation is an exhaustive search solution. Potential work in the direction of improving this initial solution would greatly improve the performance of the protocol presented. Simple optimization could be pursued depending on how the network will be used. For example, if only routing trees are needed, Algorithm 4.1 can generate trees more efficiently by not trying to include links originating at nodes already in the tree.

Additionally, novel approximate solutions using heuristics could be of great potential value. For example, trying to generate only trees with minimal degree at each node will tend to use fewer links per node and thus potentially leave links for other non-interfering

trees.

4.6.2 Synchronization

One interesting extension is to study synchronization in the absence of delay elements. For example, in an all-optical implementation it may be difficult to implement delay elements that can be finely tuned. Another example is a multiprocessor system connected by Isochronets. In this case, the end-to-end delay in the network is a critical parameter and may not be increased artificially. It would be interesting to find synchronization solutions for such systems.

4.6.3 Band Allocation

Probably the richest opportunity for future work in Isochronet mechanisms is to find new techniques to solve the band allocation problem. The solution presented in this chapter is but one possible dynamic band sizing protocol. Many others need to be exploited and analyzed to compare their performance values.

Another rich domain is the study of priority band distribution among nodes in the network. This distribution has the potential of trading resource use for QoS demands.

4.7 Summary

The mechanisms necessary for Isochronet operations are the solutions to three problems: (1) tree allocation, (2) band synchronization, and (3) band allocation. Each of these can lead to intricate studies. This chapter defines formally each of the problems and gives initial solutions to them.

The tree allocation problem is concerned with allocation routing and multicast trees in

the network. It is solved using an exhaustive search approach. This initial solution is acceptable because the allocation only happens upon network reconfiguration.

The band synchronization problem is to allocate bands so that frames transmitted from a node within a particular band reaches the next node in the path within the same band. The synchronization must also avoid overlapping different bands at a node. The proposed solution to this problem is to increase or decrease the link delays so as to make the final delay a multiple of the cycle period. By using this technique, frames transmitted at time *t* within a cycle reach each node in its path at the same time *t* some cycles later. Finally, one can simply allocate exactly the same bands in all nodes and they will be automatically synchronized.

The band allocation problem is concerned with sizing bands so as to accommodate the loads in each tree. A possible technique to solve this problem is to use an LRU approach similar to the one used in memory allocation policies. In such solution, bands from idle trees are allocated to busy trees in order to balance traffic demands.

All of these solutions are but a first cut through the problems. Future research has ample opportunities in each of these domains, especially in the band allocation problem.