A FRAMEWORK FOR ADMISSION MANAGEMENT OF DYNAMIC CONFERENCING CALLS UNDER A MULTI-POINT ATM ARCHITECTURE

Pratyush Moghe and Izhak Rubin
Department of Electrical Engineering, UCLA
{pmoghe,rubin}@ee.ucla.edu

ABSTRACT
The paper develops a framework for representing dynamic calls (i.e., calls with a variable number of clients) and the associated link-level connection (VCL) variability in an ATM multi-point architecture. A generic parameter vector is formulated for the dynamic call object. Under the selected network architecture, a tag-update algorithm isolates the impact of this global call object on the VCL layer. Each carried client is assumed to be active (transmit/receive) or inactive (receive). A simple two-state model captures the dependence of client status on the status of other clients in the same call. The activity (or active VCL) layer is assumed to be constrained (through a probabilistic threshold) by the lower ATM layer. We present a distributed call admission scheme that ensures this threshold is not exceeded on any directed link of the admitted call multi-cast tree.

1. INTRODUCTION
Future BISDN networks will require to provide services such as multi-point and multi-media dynamic calls, i.e., conferencing type or client-server type calls where the number of the communicating participants can vary during the life-time of the call [1][2]. The traditional approach that treats a call as a monolithic end-to-end object (used for fixed traffic type, using fixed number of channels or connections) is limiting in this context. In this paper, we redefine a call as a high-level distributed network object associated with a specified communication service. Refer to Figure 1. A call is identified with a unique call descriptor and is implemented through end-to-end logical connections (VCC or VPC pipes). The clients that share a call use the call descriptor to communicate via these connections. A view of a client is its call-context; generally, it is a multi-cast tree, rooted at the client, whose leaves comprise the recipient clients (called sink-clients). The set of sink-clients in a client's view forms the scope of the client. A client with complete scope is one whose view contains all carried clients in its call. An all-way conference is a typical application where all clients have complete scope.

The revised call concept introduces complexity in all aspects of management. This paper focuses on admission management of all-way conferencing calls with dynamic clients. We address two important issues. First, it is inefficient to undertake independent connection-admission procedures everytime a new client adds on to a dynamic call. Instead, we propose call parameters that will enable us to statistically reserve bandwidth for future client additions. Once a call is admitted, all conforming new clients will be set up without end-to-end admission. Second, we incorporate the dependence of client status (active/inactive) on the status of other carried clients within the call. This dependence affects the number of VCLs that are actively carried. Finally, we build a call-admission scheme that admits an incoming call if its active VCL-size distribution satisfies the underlying ATM-layer QOS.

The paper is organized as follows. Related work is discussed in Section 2. In Section 3.1 we propose a network architecture that implements the dynamic call object. Section 3.2 outlines traffic parameters for the call object. In Section 3.3 a simple model captures the dependence between activity status of the carried clients in a call. Section 3.4 summarizes the approach in deriving the active-VCL distribution. Analytical details are presented in Section 4. Section 5 outlines the call admission procedure. Results are presented in Section 6.

1.1. Related Work
Bandwidth partitioning is done between static and dynamic multi-cast calls on a single link in [4].

---

1This paper was supported by University of California MICRO and Pacific Bell Grant No. 94-107.

Footnotes:

[1] We assume all new clients are conforming to the call parameters.


Figure 1: Revised Call Concept, View, and Scope
loss is also introduced as a new connection level performance measure for dynamic calls. In [5], a static conference model is formulated and the bandwidth allocation problem is studied in the particular case of video traffic. The model limited in the sense that not more than one source can transmit at a time. [6] discusses strategies to reduce conference blocking probability with the clever positioning of a conference-center.

2. MODEL DEVELOPMENT

2.1. Network Architecture

Minimum multi-cast tree (MCT) construction for static multi-point calls is a hard problem [7]. Heuristics [8-9] have been suggested. We select a network architecture such that an MCT has to be constructed just once for each dynamic call. Refer to Figure 2. All nodes in the network are assumed to be reconfigurable multi-cast ATM switches. The network is classified into a set of access nodes and transit nodes. The access nodes are assumed to be Virtual Circuit switches, while the transit nodes are Virtual Path switches. An arriving call request declares the exhaustive list of access nodes (called member access nodes) required during its life-time. A minimum MCT is constructed between the member access node set. This involves static allocation of VPIs at all the switches. VCLs (equal in number to the carried clients) are dynamically allocated(deallocated) at the access nodes when clients enter(leave) the call. For multiple clients connected to an access node in receive mode, information is multi-cast at the access node.

2.2. Call Parameters

An arriving conferencing call-request C declares the identity of the member access nodes. Define Ω to be the member access node set. Clients carried at call setup are called primary clients, while new additions are called secondary clients. For each member access node, C declares the number of primary clients (V) and the fraction of the call clients² (pD). C also specifies the global secondary client arrival rate (λC), i.e. the rate at which new client requests arrive to the call object during its life-time. Other parameters specified are the mean holding time of each client (1/µ), and the maximum simultaneously carried clients in the call (Kmax). If the number of carried clients in the call is equal to Kmax, arriving secondary clients are blocked.

A call terminates when all the clients leave. All the clients in the call are assumed to have complete scope.

2.3. Activity Layer

We assume that a carried client can be either active (i.e. transmitting and receiving) or inactive (receiving). A VCL used by an active client is called an active VCL or an activity on the particular link. Figure 3 illustrates the call, client (connection), and activity layers. We assume that the ATM layer (below the activity layer) presents a threshold M on the number of active VCLs. We require that M be exceeded with a probability no larger than ζ. This probabilistic threshold guarantees better than worst-case acceptable ATM layer QOS.

![Figure 3: Call, Client, and Activity Layers](image)

Figure 3: Call, Client, and Activity Layers

Figure 4 indicates the model used to capture the dependence between activity status of clients in a call. Carried clients are distributed between pool 1 (inactive pool) and pool 2 (active pool). Given k clients in pool 1(2), 1 ≤ k ≤ n, the rate of transition of any client to pool 2(1) is assumed to be α kΨ−1 (β kΨ−1). Constants α (α > 0) and β (β > 0) are transition rate parameters. Ψ (Ψ ≥ 1) is an activity dependence parameter. At Ψ = 1, the client activity durations are mutually independent; larger the Ψ, greater the dependence of a client transition from a pool on the other clients in the same pool.

![Figure 4: Activity Model](image)

Figure 4: Activity Model: R1 = α kΨ−1, R2 = β (n − k)Ψ−1

---

²pD is the fraction of the call C clients served during its entire life-time that originate at access node i.
2.4. Solution Methodology

(I) The first step is to derive the call C client-size distribution on any link of the multi-cast tree (MCT). This VCL-size distribution is derived in following steps. (1) A Markov model is developed for the global client-size process of C. (2) A secondary arrival parameter \( p^{SA} \) is defined per access node. \( p^{SA} \) is computed in terms of the known access node parameters \( (V, p^D) \) and the global parameters \( (\lambda_{sec}, \mu) \) of call C. An access node tag \( (V, p^D, p^{SA}) \) is formed. (3) A tag-update algorithm then builds a link-level tag set in terms of the access node tags of C. The link-level tag set and the global client-size process are jointly used to derive the VCL-size distribution.

(II) The second step is to derive the call C active VCL-size distribution on any directed link in MCT. By quasi-stationarity, we assume that the activity layer reaches steady-state between client-size changes within C. We also assume that all clients are homogeneous and require a single VCL each.

(III) Finally, a distributed call admission procedure is formulated in terms of the call C active VCL-size distribution and the aggregate active VCL-size distribution for all carried calls.

3. ANALYSIS

3.1. Global Client-size Process N

We artificially generate a client size process \( N = \{N_t, t \geq 0\} \), where \( N_t \) is the number of clients (indexed by \( n \)) at time \( t \) in call \( C \). \( N_0 \) models the global client-size process of \( C \). The constructed process \( N \) begins each busy-period with an initial load of \( K = \sum_{i \in S} V_t \) primary clients. Secondary client arrivals during its busy period are assumed to occur at a Poisson rate of \( \lambda_{sec} \). These arrivals are admitted if the number of carried clients is less than \( K_{max} \), else they are blocked. The holding time of each client is assumed to be exponentially distributed with parameter \( \mu \). When all the clients in the call depart, we introduce an artificial idle period3 (exponentially distributed with parameter) \( N \) can thus be modeled as a continuous-time Markov chain over the state space \( S = \{n, n = 0, 2, 3, ..., K_{max}\} \). Note that there cannot be a lone client in the call.

3.2. Construction of Access Node Tag

We have assumed that call C declares two access node parameters: \( V \), the primary client size and \( p^D \), the fraction of call traffic. We now define a third parameter \( p^{SA} \), to be the fraction of global secondary clients that arrive at an access node. \( p^{SA} \) can be readily derived in terms of the global client-size process and the known parameters, \( V \) and \( p^D \).

Let \( U \) be the discrete-time Markov chain underlying \( Z \). The mean number of clients served per busy-period, \( E(N_{BF}) = \sum_{n=3}^{K_{max}} N_{BF} R_U(n) + \frac{U(2)}{U(0)} \cdot R_U(2, 0) \cdot 2, \) where \( U \) is the stationary distribution of \( U \) and \( R_U(i, j) \) is the \( (i, j) \)th entry of \( R_U \), the transition probability matrix of \( U \). Then, \( p^{SA}(i) = \frac{p^D(i) \cdot E(N_{BF}) - V(i)}{\sum_{i \in \mathbb{Z}} p^D(i) \cdot E(N_{BF}) - V(i)} \), \( \forall i \in \mathbb{Z} \).

The derived \( p^{SA} \) values and parameters \( V \) and \( p^D \) are now included in an access node tag, \( ANT \). For \( i \in \mathbb{Z}, ANT_i = (V, p^D, p^{SA}) \). The next section develops a link-level tag set that permits an isolated description of the call C VCL-size process on any directed link in the call MCT.

3.3. Link-level Tag Set Algorithm

Assume that all the access nodes have been tagged as described before. Our objective is to derive a tag set (left and right tag) for each undirected link. For a link \( (i, j) \), the left tag will determine the VCL size on the directed link \((i, j)\), and the right tag for the directed link \((j, i)\). The tag set is synthesized through the following algorithm.

1. With each access node \( i \ (i \in \mathbb{Z}) \), associate a field \( F_l \triangleq (ANT_i, 0) \). In general, \( F = (ANT, H) \), where the \( H \) is a non-negative integer.

2. Identify the multi-cast tree MCT associated with call C. For each node \( j \) in MCT, construct a field-table \( FTPAB \) with \( |\mathbb{Z}| \) empty field entries. The \( i \)th entry of this table \( F^J_l \) is denoted as \( (ANT, H) \).

3. Select, in any order, a node \( w \) from the member access node set as the root node of the MCT. (The corresponding view of \( w \) is MCT_\( w \).)

(ii) Flood MCT_\( w \) with field \( F_w \)

(a) For every node \( j \) that the cell encounters in MCT_\( w \), update the \( i \)th entry of FTPAB: \( ANT_{\mathbb{Z}} = ANT_{\mathbb{Z}}, H_{\mathbb{Z}} = H_{\mathbb{Z}} + Hop.Count(w, J) \)

(ii) Select, in any order, a node \( w \) from the member access node set as the root node of the MCT. (The corresponding view of \( w \) is MCT_\( w \).)

(i) Flood MCT_\( w \) with field \( F_w \)

(ii) For every node \( j \) that the cell encounters in MCT_\( w \), update the \( i \)th entry of FTPAB: \( ANT_{\mathbb{Z}} = ANT_{\mathbb{Z}}, H_{\mathbb{Z}} = H_{\mathbb{Z}} + Hop.Count(w, J) \)

(iii) Select, in any order, a node \( w \) from the member access node set as the root node of the MCT. (The corresponding view of \( w \) is MCT_\( w \).)

4. For an undirected link \( (i, j) \) in MCT, compute the left tag: \( T_l(i, J) = \sum_{k} T_{k}(i, J), ANT_{\mathbb{Z}} \), where \( T_{\mathbb{Z}}(i, J) = \{k | F_k \in FTPAB, H_k < H_k' \} \), and the right tag: \( T_r(i, J) = \sum_{k} T_{k}(i, J), ANT_{\mathbb{Z}} \), where \( T_{\mathbb{Z}}(i, J) = \{k | F_k \in FTPAB, H_k < H_k' \} \).

5. Denote the contents of \( T_l \) and \( T_r \) for link \( IJ \) as \( (V_L, p^{SA}_L, p^{SA}_R) \) and \( (V_R, p^{SA}_L, p^{SA}_R) \) respectively. If \( min(V_L, V_R) = 0 \), set \( V_L = V_R = 0 \).

Note: By flow conservation, \( p^{SA}_L = 1-p^{SA}_R \) and \( p^{SA}_L = 1-p^{SA}_R \).

3.4. VCL-size Distribution Z of a Call

Define:

\( XL = \{X_L, t \geq 0\} \), where \( X_L \) is the number of call \( C \) VCLs (indexed by \( l \) ) at time \( t \) on a directed link \( IJ \),

\( XR = \{X_R, t \geq 0\} \), where \( XR \) is the number of call \( C \) VCLs (indexed by \( r \) ) at time \( t \) on a directed link \( IJ \).

\( Z \triangleq (N, XL) \)
CLAIM: Z is a time-homogeneous continuous-time Markov chain over state space \( S_Z = \{(n, l), n \in S; l = 0, 1, \ldots, n - 1\} \), with transition parameters completely described by those of \( N \) and the left tag.

We omit the proof and the derivation of the transition rates of \( Z \) for brevity. Define \( Z \) as the steady state distribution of \( Z(n, l) \triangleq \lim_{t \to \infty} P\{N_t = n, X_L_t = l\} \). Assuming \( Z \) is derived, we can compute \( Z(n, l) = \lim_{t \to \infty} P\{(N_t = n, X_L_t = l)\} \).

3.5. Active VCL-size Distribution of a Call

In this section, we use quasi-stationarity and the activity model in Figure 4 to derive the steady-state active VCL-size distribution conditioned on \( n \) clients in the call and \( l \) VCLs on the directed link \( IJ \).

Define:

- \( Y = \{Y_t, t \geq 0\} \), where \( Y_t \) is the number of call \( C \) clients in the active pool at time \( t \).
- \( A_L = \{A_L(t), t \geq 0\} \), where \( A_L(t) \) is the number of call \( C \) active VCLs on the directed link \( IJ \) at time \( t \).
- \( A_R = \{A_R(t), t \geq 0\} \), where \( A_R(t) \) is the number of call \( C \) active VCLs on the directed link \( JI \) at time \( t \).

Let \( Y, A_L, A_R \) be their respective steady-state distributions.

The second term \( (Y(y, l)) \) is the steady-state call \( C \) client distribution in the active pool, given \( n \) clients in the call. This can be derived in straightforward steps (using Figure 4 model):

\[
Y(n, l) = \frac{(\alpha/\beta)^y}{\sum_{y=0}^{n} (\alpha/\beta)^y} \quad y = 0, 1, \ldots, n.
\]

The first term \( (A_L(i, y, n, l)) \) is the conditional probability of an active client belonging to link \( IJ \). This is computed by apportioning active clients to \( IJ \) and \( JI \) with probabilities proportional to \( X_L \) and \( X_R \) respectively. The derivation is omitted. Assuming \( A_L \) is put together, \( A_R \) can be derived through reciprocity: \( A_R(j) = \sum_{y=0}^{\infty} A_L(i, y, n, l) Y(y, n, l) \).

4. CALL ADMISSION SCHEME

The active VCL parameters \( M \) (threshold on active VCLs on a directed link) and \( C \) (tolerable overflow probability) are assumed to be known from the ATM layer.

Call_Admisison_Procedure for \( C \):

1. Select a link \( IJ \in ARC \)
2. Compute \( A_L^{(IJ)}(i) \) and \( A_R^{(IJ)}(j) \) due to call \( C, \forall i, j = 0, 1, \ldots, K_{max} - 1 \) (Section 4)
3. Let \( A_L^{agg^{(IJ)}}(i) \) and \( A_R^{agg^{(IJ)}}(j) \) be the aggregate active VCL-size distributions of currently carried calls on \( IJ, i, j = 0, 1, \ldots, a_{max} \).

4. Compute the activity overflow probabilities \( A_{of} \):
   \[A_{of,L} = \sum_{i=0}^{a_{max}} K_{max} - 1 A_L^{agg^{(IJ)}}(i),\]
   \[A_{of,R} = \sum_{i=0}^{a_{max}} K_{max} - 1 A_R^{agg^{(IJ)}}(j),\]
   If \( A_{of,L} \geq \zeta, \) or \( A_{of,R} \geq \zeta, \) REJECT \( C \), break;
5. \( ARC \leftarrow ARC \setminus IJ \)
   If \( |ARC| > 0 \), return to 1 and repeat.
6. ADMIT \( C \)

7. Restore \( ARC \) and \( VIJ \in ARC \), allocate resources:
   \( A_L^{agg^{(IJ)}} := A_L^{(IJ)}, A_R^{agg^{(IJ)}} := A_R^{(IJ)} \)

Call_Departure_Update_Procedure for \( C \):
1. Identify \( MCT \) and \( ARC \)
2. Select \( IJ \in ARC \), compute \( A_L^{(IJ)} \) and \( A_R^{(IJ)} \)
3. Reverse-convolve \( A_L^{agg^{(IJ)}} \) with \( A_L^{(IJ)} \) and \( A_R^{agg^{(IJ)}} \) with \( A_R^{(IJ)} \)
4. \( ARC \leftarrow ARC \setminus IJ \), If \( |ARC| > 0 \), go to 1, repeat.

5. NETWORK EXAMPLE

Consider the 9-node network in Figure 5. Table 1 lists the relevant parameters for the four carried calls.
the two pools, its duty-cycle (fractional time it spends in the active pool) is \( f/(1+f) \). The latter is defined as the *activity factor* \( af \), \( af \in [0,1) \). When \( af = 0.5 \), \( f = 1 \) and rates \( \alpha = \beta \) (half duty-cycle). In this particular case, \( A_{af} \) is responsive in the \( af \) range of \([0.5,0.9]\). At larger \( af \) values, the \( A_{af} \) values spread out for opposite directions of a link.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Calls C1, C2</th>
<th>Calls C3, C4</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \lambda_{sec} ) clients/min</td>
<td>8.00,5.00</td>
<td>13.00,0.1</td>
</tr>
<tr>
<td>( \mu ) VCLs/min</td>
<td>5.00,2.00</td>
<td>5.00,2.0</td>
</tr>
<tr>
<td>( K ) Tot Prim Clients</td>
<td>4.5</td>
<td>4.3</td>
</tr>
<tr>
<td>( K_{max} ) Max Clients</td>
<td>10,15</td>
<td>10,5</td>
</tr>
<tr>
<td>Member Access Nodes</td>
<td>(1,2,3), (2,3,4)</td>
<td>(4,1,3), (3,2)</td>
</tr>
<tr>
<td>( V ) (Primary Clients)</td>
<td>(1,1,2), (1,3,1)</td>
<td>(1,3,0), (1,2)</td>
</tr>
<tr>
<td>( p^{D} )</td>
<td>(0.3,0.4,0.3), (0.1,0.8,0.1)</td>
<td>(0.01,0.83,0.16)</td>
</tr>
<tr>
<td>( p^{DA} ) : Computed</td>
<td>(0.37,0.61,0.02), (0.03,0.94,0.03)</td>
<td>(0.09,0.91)</td>
</tr>
<tr>
<td>Activity Layer (( f, \Psi ))</td>
<td>(1.133,1.1)</td>
<td>(1.133,1.1)</td>
</tr>
<tr>
<td>Max. ( f_{sec} )</td>
<td>0.005</td>
<td>0.005</td>
</tr>
</tbody>
</table>

Table 1: Call Parameters for the Network Example in Figure 5.

Figure 6: Effect of Activity Factor \( af \) on \( A_{af} \).

Figure 7 examines the effect of activity-layer dependence parameter \( \Psi \) on \( A_{af} \). Recall \( \Psi \) was the exponent of the activity model in Figure 4. \( A_{af} \) decreases with increasing dependence; the drop is faster for heavily loaded links (link 83) when clients are even lightly dependent (\( \Psi \in [1,1.2] \)). Thus, admission management schemes can derive substantial benefit by utilizing the dependence effect between clients. The overall overflow probability can be reduced, and hence more calls can be admitted (increasing the call throughput).

### 6. CONCLUSIONS

A framework for representing dynamic-client conferencing calls has been developed. The conferencing application is described through a set of global parameters. A tag-update algorithm synthesizes the link-level parameters that are used to derive the VCL-size on any directed link of the call multi-cast tree. Each client in the call is assumed to be active (transmitting/receiving) or inactive (receiving). A distributed call admission scheme is developed, which ensures that the aggregate active VCLs on any directed link of the admitted call multi-cast tree are within the constraints imposed by the lower ATM layer QoS.

### 7. REFERENCES


