A Distributed Constraint Optimization Solution to the P2P Video Streaming Problem

Theodore Elhourani, Nathan Denny and **Michael Marefat**
Department of Electrical and Computer Engineering
University of Arizona
Tucson, AZ 85721
{telhoura,denny,marefat}@ece.arizona.edu

Abstract

The future success of application layer video multicast depends on the availability of video stream distribution methods that can scale in the number of stream senders and receivers. Previous work on the problem of application layer video streaming has not effectively addressed scalability in the number of receivers and senders. Therefore, new solutions that are amenable to analysis and can achieve scalable P2P video streaming are needed. In this work we propose the use of automated negotiation algorithms to construct video streaming trees at the application layer. We show that automated negotiation can effectively solve the problem of distributing a video stream to a large number of receivers.

Introduction

This work addresses the problem of scalable video streaming in a peer-to-peer (P2P) network. A P2P network is a virtual network of computing nodes and logical links that is constructed on top of the Internet. The purpose of a P2P network is to provide functionalities that are not available or poorly supported on the Internet. Video stream multicast is a notable example. We use distributed constraint optimization techniques to achieve scalable video streaming in a P2P network. To achieve scalability in the number of stream providers and receivers we model the video streaming system as a time bounded distributed constraint optimization problem.

A disadvantage of video download is that the video has to be downloaded in its entirety before it can be viewed. Streaming attempts to overcome this disadvantage of file download. However the design of a video streaming system is not trivial. There exists a wide range of video streaming applications: point-to-point video, multicast and broadcast video streaming are examples. In this paper we focus on the problem of providing a video stream to a large group of end users (video multicast). Three characteristics of a communication channel: bandwidth, delay, and packet loss affect video communication. Continuously adjusting the video bit rate in response to small bandwidth fluctuations alleviates the risk of congestion and packet. However, it does not solve the problem of reliably delivering a quality stream to a group of receivers. We assume that a playback buffer compensates for end-to-end delay ensuring a jitter free presentation. In addition, we assume an error free channel, and therefore an acceptable level of packet loss for video streaming.

Constraint satisfaction and optimization problems are a general framework for problem solving in artificial intelligence. In the multi-agent systems research field a major problem is to achieve consistency among agents. The distributed constraint optimization problem (DCOP) is a COP where the variables are distributed among a set of agents. Distributed algorithms assigning values to all variables such that inter-agent constraints are satisfied are procedures for achieving consistency among agents. A DCOP is stated as:

- $A = \{a_1, a_2, ..., a_m\}$ set of m agents
- $V = \{x_1, x_2, ..., x_n\}$ set of n variables
- $D = \{d_1, d_2, ..., d_n\}$ domains for variables in V
- $C_h = \{c_1, c_2, ..., c_k\}$ set of hard constraints on values of variables in V,where $c_i = p_i(x_{k1}, x_{k2}, ..., x_{ki})$ is a constraint predicate
- $C_s = \{F_{S_1}, F_{S_2}, ..., F_{S_l}\}$ set of soft constraints defined over subsets of V
- $F_g = \sum F_{Q_i}$ global objective function

Advantages and problems of P2P video streaming

There are several advantages in migrating multicast to the application layer. First, application layer nodes, also referred to as peers, have higher buffering, and intelligent routing capabilities. Second, since a dedicated infrastructure is no longer used we potentially can achieve scalability in the number of servers and clients. In addition, distributing the system tasks over a wider set of nodes makes the overall system more resilient to local failures.

All these advantages are leveraged in a video streaming network at the cost of complicating the design of the system. Allocating bandwidth within a group of end hosts becomes an issue that needs to be dealt with at the application layer. The fact that supply bandwidth varies among end-hosts makes the problem even more difficult. Node dynamics prescribe the use of an online routing solution taking into account bandwidth constraints.

In a P2P video streaming system all nodes simultaneously act as both clients and servers. Hence the problem is inherently distributed and DCOP algorithms are good candidates for a solution. However, to guarantee jitter free playback, the nodes must be able to reconstruct the streaming trees before the video playback buffers become empty. This introduces a new challenge to the design and choice of DCOP algorithms and motivates the development of a bounded DCOP model.

Related work

Several distributed algorithms have been developed for solving DCOPs (Yokoo & Hirayama 2000). We use DBA, distributed breakout algorithm (Yokoo & Hirayama 1995) chiefly because it is an iterative improvement algorithm. This property is essential since the algorithm may be stopped before it completely solves the problem and a near-optimal solution can be used. For a more information on the completeness of DBA as well as the application of constraint satisfaction in communications consult (Eisenberg & Faltings 2003; Frei & Faltings 1999; Elhourani 2006).

Three major architectures for application layer multicast have been proposed (Ganjam & Zhang 2005). In the dedicated-infrastructure model routers performs all the routing functions. In this model, although the number of receivers can potentially be large, the number of senders is constrained. The dedicated-infrastructure model is appropriate for a small number of large receiver groups. Akamai and Real Networks are prominent examples of this model. The application-endpoint model uses only end-hosts to perform multicast functions. It is self-scaling, however the system is heavily dependent on the bandwidth of participating nodes. It is appropriate for a large number of small to medium sized groups. Supply bandwidth is the biggest challenge in the application-endpoint model, as the contributed bandwidth from peers (end-hosts) is unknown at any given point. Other challenges of the application-endpoint model include: sender/receiver scalability and network dynamics. Sender/receiver scalability is achieved in a fully distributed system, such as the DCOP-based system we propose here. Network dynamics pose a great challenge to an application-endpoint video streaming system. As we will show later, this problem will be alleviated by implementing the bounded negotiation technique. In the "waypoint model" end hosts perform multicast functions, however, dedicated nodes can be dynamically invoked to contribute bandwidth to underserved receivers. Each of the three models of application layer multicast presents several challenges. In our work we develop an application-endpoint model.

Application layer multicast (Chu *et al.* 2002; Jannotti *et al.* 2000) was suggested as an alternative to IP multicast. In application layer multicast end-hosts play the role of both senders and receivers. The goal of the application layer multicast is to provide multicast functionality over the Internet. All the multicast functionalities are embedded in the application layer, and routers only perform unicast forwarding.

Zhang et al. (Zhang *et al.* 2005) propose CoolStreaming/DONet, a data-driven P2P network for efficient live media streaming. In DONet the availability of data, a video for instance, guides the flow directions. No specific network structure, such as streaming trees, is maintained. It is not clear how this choice of a provider node can lead to load reduction. In our work we employ negotiation to achieve a near-optimal load distribution among a group of senders and receivers. P2Cast (Guo *et al.* 2003) is an application layer architecture for cooperatively streaming video using a video patching technique. It assumes that only 10% of the stream's initial portion is cached at every node. Therefore arriving nodes must get patches from currently streaming nodes, and then elect a new parent node. In our work we assume that a node caches 100% of the stream's initial portion. Our definition of a video streaming service is hence slightly different. For a more detailed review of earlier systems see (Elhourani 2006).

Definitions and problem statement

We first give formal definitions of the entities involved in a P2P video streaming network.

Definition 1: The video streaming network is a graph $G = (A, E)$, where A is the set of nodes in the network, E is the set of edges (A_i, A_j) connecting nodes A_i, $A_j \in A$. If two nodes A_i and A_j have the addresses of each other, and own portions of the same video, then they are connected with an edge (A_i, A_j).

Definition 2: A node $A_i \in A$ is a 6-tuple $(A_{p_i}, A_{n_i}, U_i, U_{i_{max}}, L_i, B_i)$ where A_{p_i} is the node that is providing the video stream to node A_i, $A_{n_i} \subset A$ is the set of neighboring nodes, U_i is the current upstream bandwidth usage of node A_i, $U_{i_{max}}$ is the maximum available upstream bandwidth for A_i, L_i is the length of the portion of the video that has already been streamed at node A_i, and B_i is the current playback buffer level.

Definition 3: A neighborhood $N = (A_N, E_N)$ is a clique within the network $G = (A, E)$. N is a set of nodes which have the addresses of each other and all possess parts of the same video.

A node stores a complete initial portion of the video locally. It is also assumed that once the streaming is completed the video is stored locally and may be streamed to neighboring nodes. However, a node can choose to leave the network at anytime.

As nodes join and leave the video streaming network the stream distribution trees must be reconstructed such that no nodes are left without a stream provider and the distribution trees are balanced. By balanced trees we refer to video stream distribution trees where upstream bandwidth usage percents are roughly equal at all nodes. Assigning a provider node $A_i \in N$ to every node $A_j \in N$ is a typical combinatorial optimization problem with hard constraints. It can also be formulated as the problem of constructing a spanning tree subject to the given constraints. Effectively, the constraints reduce the neighborhood N to a directed graph where each edge represents a legitimate point-to-point streaming path. Subsequently, a minimum spanning tree can then be easily constructed. However, due to the inherent distribution of our problem setting we would like to construct this tree in a decentralized fashion.

In addition parent nodes belonging to the same neighborhood must have roughly equal percent channel usage. We

hence add a soft constraint and minimize its associated objective function. The process of constructing a streaming tree subject to all the aforementioned constraints is itself temporally bounded. Nodes that are looking for a parent have a limited buffer capacity. To avoid discontinuity in the playback of the video the buffer must never be allowed to go empty. Therefore a parent must be assigned before the buffer is depleted to avoid jitter.

Given $N = (A_N, E_N, L_N, U_N, U_{N_{max}}, B_N)$, find a spanning tree $T = (A_T, E_T)$ such that:

$\sum_{e \in E_T} d_e \; is \; minimized, \; where$

$$d_e = |U_i/U_{i_{max}} - U_j/U_{j_{max}}|$$

$\sum_{e \in E_T} s_e = 0, where:$

$$s_{e_{(i,j)}} = \begin{cases} 1, & e_{i,j} \in E_T \; and \; e_{j,i} \in E_T \\ 0, & otherwise \end{cases}$$

$\sum_{e \in E_T} l_e = 0, where:$

$$l_{e_{(i,j)}} = \begin{cases} 1, & e_{i,j} \in E_T \; and \; L_i < L_j \\ 0, & otherwise \end{cases}$$

$\sum_{A_i \in A_T} u_{A_i} = 0, where:$

$$u_{A_i} = \begin{cases} 1, & U_{A_i} > U_{A_{i_{max}}} \\ 0, & otherwise \end{cases}$$

$\forall A_i \in N \; . \; B_i > 0$

We have formulated the problem as a special kind of distributed minimum spanning tree problem (Gallager, Humblet, & Spira 1983). Our problem formulation differs from the distributed minimum spanning tree problem in several ways. First, we have two independent weights defined on edges: s_e, l_e. These weights are not constants, but rather functional components. Second, there are two constraints defined over nodes: $\sum_{A_i \in A_T} u_{A_i} = 0$, and $B_i > 0$. The distributed minimum spanning tree problem formulation does not include any such constraints. And finally we also augment the model with a global soft constraint min $\sum_{e \in E_T} d_e$, also not in a distributed minimum spanning tree problem formulation. Modifying the distributed minimum spanning tree problem formulation to account for these constraints is not straightforward when compared to our proposed solution.

Convergence rates of DBA

P2P networks exhibit the characteristics of scale-free networks. Sarshar and Roychowdhury (Sarshar & Roychowdhury 2004) show that highly dynamic P2P networks have a scale-free structure with relatively high exponents, $a \approx 3$. A network whose vertex connectivity follows a power-law degree distribution is considered to be a scale-free network (Barabasi & Albert 1999). Several methods for generating scale-free networks have been proposed. We use the generative model of Holme and Kim (Holme & Kim 2002) to construct scale-free networks with exponent values $a \approx 3$. An equally important parameter of a scale-free network is the clustering coefficient, γ, defined as follows. Consider a graph node n_0 and its adjacent nodes $n_1, n_2, ..., n_k$. Then,

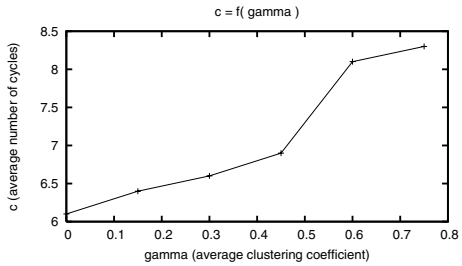


Figure 1: Number of DBA cycles as a function of γ

the clustering coefficient of node n_0 is the number of edges that link its adjacent nodes divided by the total number of edges that can possibly link its adjacent nodes. We note that the Holme and Kim model allows the tuning of γ.

We use the graph coloring problem to evaluate DBA in scale-free networks. We generate scale-free networks of size $N = 200$ for clustering coefficient values $0 \leq \gamma \leq 0.8$. In the graph coloring problem each node has one variable, the color variable. The constraint is that no two adjacent nodes can assume the same color. Intuitively, the higher the clustering coefficient the more difficult the problem. It is apparent from figure 1 that for values of $\gamma > 0.5$ a sharp rise in the number of DBA cycles occurs. Our result can be used to control the clustering coefficient of a P2P network. Hence, the average number of cycles required for the DBA algorithm to converge to an optimal solution can be bounded given that the power law exponent a is set to 3. To the best of our knowledge this is the first established mapping from a scale-free network parameter, γ in this case, to the average maximum number of cycles required for a DCOP class negotiation algorithm to converge.

Bounded negotiation

In this section we show how our new method for bounding negotiation cycles works. We assume that the number of negotiation cycles is bounded by the individual agents. A negotiation phase ends when at least one of the agents opts out. An agent can opt out for several reasons. In this work, the negotiation is temporally bounded. A negotiating node opts out when its video playback buffer is empty. At that point the negotiation has failed to reach an optimal solution. However, a near-optimal, lower quality solution, can still be used. We would like to know how the quality of a solution is affected by both the number of negotiation cycles and the clustering coefficient of a scale-free network. The clustering coefficient is the only parameter of the streaming network that we can control.

In this work we define "negotiation cycles" to be the number of messages exchanged between a set of nodes. We hence count the number of cycles between the initial state and when one of the nodes opts out or when the network reaches an optimal solution (all hard constraints are satisfied and the soft constraint is minimized). Solution quality combines the optimality of the soft constraint and the percentage of satisfied hard constraints. We define an aggregate measure of the solution quality as follows: $Q =$

$\frac{P_h \times 100 + (1-P_{F_g}) \times 100}{2}$, where $P_h = \frac{|C_{h_s}|}{|C_h|}$ is the percentage of satisfied hard constraints, with $|C_{h_s}|$ as the number of satisfied hard constraints and $|C_h|$ is the number of hard constraints. $P_{F_g} = 1 - \frac{F_g}{F_{g_{max}}}$ is the normalized soft constraint. F_g being the value of the soft constraint at the time the negotiation halts, and $F_{g_{max}}$ the maximum possible value of the soft constraint (worst value). Therefore, we treat the optimality of a soft constraint and the number of satisfied hard constraints as two separate measures that are equally contributing to the quality of a solution.

Experiments

The anytime behavior is manifested for both the soft and hard constraints. As more DBA cycles are executed the percentage of satisfied hard constraints, P_h, increases and F_g, the soft constraint, converges to its optimal. P_h is a monotonously increasing function of the number of cycles. F_g is a monotonously decreasing function of the number of cycles. In the first experiment, results shown in figure 2 (a), the percentage of satisfied hard constraints P_h increases as more cycles are executed. As γ increases, the number of hard constraints c_{n_i,n_j} increases too, therefore it is expected that on average more cycles are required to solve the problem. For instance, for $\gamma = 0.75$ only 30% of hard constraints are satisfied after 10 cycles, while for $\gamma = 0.15$ 80% are satisfied at 10 cycles. It is hence beneficial to keep the clustering coefficient of the video streaming network as low as possible. This result can be used to decide at what number of cycles the percentage of satisfied hard constraints is acceptable, from there the negotiation can be bounded.

In the second experiment, results shown in figure 2 (b), the normalized soft constraint P_{F_g} decreases as more cycles are executed. One should not necessarily expect faster rates of convergence for lower values of γ as in the case of hard constraints. The soft constraint is defined over a set of nodes and is not directly related to the edges in the scale-free network. Nevertheless, the simulation results show that for higher values of the clustering coefficient the convergence of the soft constraint is slower. Having established this result, the designer must impose an upper bound on the clustering coefficient when the video streaming network forms. It can be observed that traces converge to different values of P_{F_g} this is due to the fact that each trace represents a unique set of randomly generated networks. For both hard and soft constraints results indicate that for values 0.3 and 0.45 of γ the rates of convergence are nearly identical (see figures 2 (a) and (b)).

Finally, the quality of solution is generated for cycles between 2 and 22 2 (c). The quality of solution is highly dependent upon the number of cycles and the clustering coefficient of the scale-free network. This confirming result is used to control the clustering coefficient and to bound the negotiation. To the best of our knowledge, this work is the first to confirm the anytime behavior of a DCOP negotiation algorithm, and to find a relation between a scale-free network parameter and the rate of convergence of a DCOP algorithm.

Distributed constraint optimization solution

Now we model the problem of P2P video streaming as a DCOP. The problem is that of constructing a video streaming tree for a given set of nodes such that hard and soft constraints are satisfied. Consider the following modeling of the solution:
$A = \{A_1, A_2, ..., A_m\}$, set of m peers
$V = \{P_{A_1}, P_{A_2}, ..., P_{A_m}\}$, set of provider variables for each peer
$D = \{D_{P_{A_1}}, D_{P_{A_2}}, ..., D_{P_{A_m}}\}$, set of domains, with $D_{PA_j} = N - A_j$ and N is the neighborhood to which A_j belongs
We define three hard constraints as follows:

1. $\forall A_i, A_j \in A, (P_{A_j} = A_i \rightarrow P_{A_i} \neq A_j) \wedge (P_{A_i} = A_j \rightarrow P_{A_j} \neq A_i)$, simultaneous download/upload constraint
2. $\forall A_i, A_j \in A, (P_{A_i} = A_j) \rightarrow (L_i < L_j)$, video length constraint
3. $\forall A_i, A_j \in A, P_{A_i} = A_j \rightarrow U_j + u_s < U_{j_{max}}$, upstream bandwidth constraint, where u_s is the bandwidth occupied by one stream
4. $\forall A_n \in A,\ B_n > 0$

We have a set of nodes A and a set of provider variables V. Each provider variable in V belongs to a node in A. Domains of variables in V are given by set D. The first hard constraint ensures that a client node (child) does not provide a video stream to its server (parent). In other words, a node A_i, providing the video stream to node A_j, cannot have node A_j as its stream provider. The second hard constraint ensures that the length of the video that has already been streamed at a parent node A_i is larger than the size of the same video already streamed at its client node A_j. A child node looks for a parent node that has already streamed a larger initial portion of the video. In addition, we would like to make sure that no node exceeds it maximum upstream channel bandwidth. This is captured in the third hard constraint. The last hard constraint concerns the buffer level, which must always be larger than 0. It is also desirable to reduce the average percent usage of a channel within the neighborhood. We therefore augment the model with a soft constraint F.

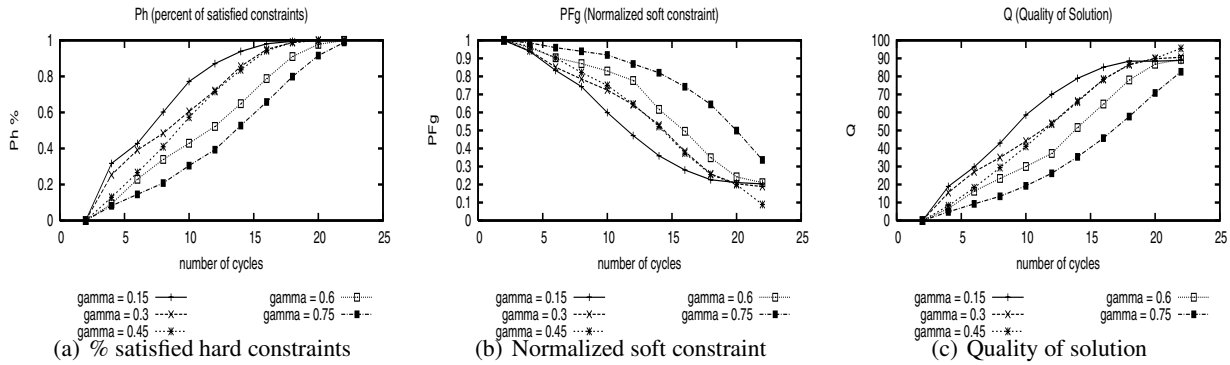
- $f_k = |U_1/U_{1_{max}} - U_2/U_{2_{max}}| + |U_1/U_{1_{max}} - U_3/U_{3_{max}}| + ... + |U_{k-1}/U_{k-1_{max}} - U_k/U_{k_{max}}|$

  $\min F = \sum f_i$

f_k is the sum of differences of the channel percent usage for every pair of nodes in the neighborhood N_k. Minimizing F has the effect of balancing the channel percent usage of all the nodes in the network. F is a global soft constraint, individual nodes are only aware of the soft constraint f_k of the neighborhood they belong to. A local soft constraint f_k is defined for each neighborhood N_k of the network.

The DBA algorithm and extensions

Two major changes are made to the DBA algorithm. First we modified the algorithm to take into account the soft constraint. Since it is impossible to know when the soft constraint is minimized, because the minimum is not known,



(a) % satisfied hard constraints (b) Normalized soft constraint (c) Quality of solution

Figure 2: Iterative improvement behavior of DBA in scale-free networks

we used a simple rule to decide when to stop the negotiation. If for four consequent cycles: 1-The soft constraint stays constant, 2-The hard constraints are all satisfied and 3-The algorithm has not entered the "quasi-local minimum state", we consider that an optimal solution was reached and halt the negotiation. The second modification concerns the buffer constraint. Each time a node checks if the current assignment is consistent, it also checks whether the buffer constraint is satisfied. In case the buffer is empty the node notifies its neighbors that the negotiation must be terminated.

We have used a dynamic scheme for the pricing of the soft and hard constraints. The soft constraint is priced to 0 until all hard constraints are satisfied. After the hard constraints are satisfied the negotiation continues with a 1 pricing of the soft constraint. Effectively, the hard constraints are given full priority until they are completely satisfied. It is important that a tree satisfying all hard constraints is found first, although the soft constraint (node bandwidth usage balancing) may not be optimal. Assuming that all hard constraints are represented by $c_h = c_{h_1} \wedge c_{h_2} \wedge ... \wedge c_{h_n}$, the weights for the constraints are as follows:

$$w_h = \begin{cases} 0 & c_h = True \\ 1 & otherwise \end{cases} \quad w_s = \begin{cases} 0 & c_s = False \\ 1 & otherwise \end{cases}$$

Experiments and results

We would like to measure rates of node rejections in a DBA-based video streaming system. This measure is important to us, since it effectively tells us what percentage of the nodes in the networks suffers playback discontinuity. When a node can not find a parent at the termination of a DCOP negotiation phase the video playback is discontinued.

We simulate a stream of node joins and leaves according to a Poisson distribution. We feed this stream to the network and measure the rates of node rejections for several means of the arrivals/departure Poisson distribution. The size of the network is not important since the clustering coefficient is constant. Results from previous simulations (section) indicate that the clustering coefficient must be chosen to be at its lowest possible value. We therefore use the lowest clustering coefficient values in the following simulations.

Since it is desired to reduce the video playback buffer size as much as possible, we set the maximum buffer size B_{max} to correspond to the maximum number of cycles required to find an optimal solution. For instance, if the video playback bit rate V_{br} = 10kbps and the maximum number of cycles required to find an optimal solution in a network is $c = 6s$, then the maximum buffer size is $B_{max} = V_{br} * c = 60kb$. We then run simulations for nodes with 25%, 50%, 75% and 100% of B_{max}. For each value of B, buffer size, we run discrete event simulations to find the node rejection probability at an increasing mean of the arrival/departure Poisson distribution. We also assume that node arrivals/departures follow a Poisson distribution with mean $1 \leq \mu_p$ (step/node) ≤ 20. As in chapters 5 and 6 we use a discrete event simulator to generate the results. For further details on the experimental setup consult (Elhourani 2006).

Our results show, as expected, decreasing trends in the rejection rates for higher arrival/departure intervals 3. When the rate at which nodes join and depart from the network is high less time is left to find optimal solutions to the negotiation problem. Therefore it is observed in the results of the simulations 3 that rejections are highest for μ_p = 1 step/node. While they drop to their lowest for μ_p = 20 step/node. In addition the size of the size of the buffer has a major effect on node rejection rates. It can be seen that when the buffer size is at its maximum (equivalent to the maximum required cycles) the rejections are in effect 0. Unfortunately the buffer may not be allowed to have the maximum size. Hence, the results show that the rejection probability increases as the size of the buffer is decreased.

We also compare the node rejection probability in our system to the reported node rejection probabilities of the P2CAST system (Guo *et al.* 2003). We consider rejection probabilities for equivalent buffer sizes and rate of arrivals. In (Guo *et al.* 2003) the buffer size is assumed to be 10% of the length of the video. We empirically know that in the worst case 25 cycles are required to construct an optimal streaming tree within a neighborhood. Therefore, assuming that a cycle takes 24ms to be completed, the total time to reach a solution is 0.6 s. If the nominal video playback rate is 1.5Mb/s then the buffer size for which the negotiation is not temporally constrained is 112Kb bytes. considering a

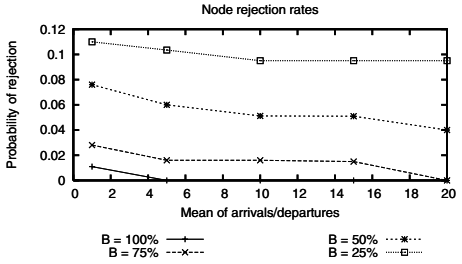


Figure 3: Rejection probability of a joining node

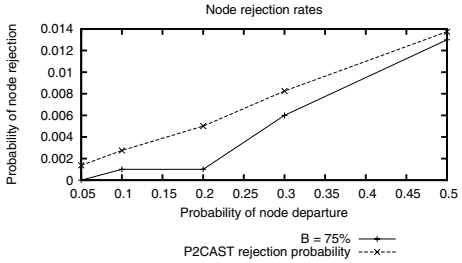


Figure 4: Rejection probability of a joining node compared to P2CAST

1Mb video, the buffer size in P2CAST would be 100Kb. We hence can consider simulations with 75% of the buffer size in our system.

Figure 4 shows that the rejection rates of our system are slightly lower than the rates reported in (Guo *et al.* 2003). In P2CAST the probability of rejection is generated for only the case when nodes prematurely quit the network. In our simulations we assume that rejections occur when nodes leave the network and when new nodes attempt to join the network. This is more realistic since the joining nodes may also destabilize the network, increasing the workload on already existing node and potentially forcing some nodes to leave the network. This promising result shows that our method is capable of outperforming previous approaches. We are effectively reducing the probability of rejection while adding robustness and scalability by using a fully-decentralized system.

Conclusion

The contributions of this work are: modeling and solving the problem of P2P video streaming using distributed constraint optimization, determining the relationship between the clustering coefficient of a scale-free network and the number of cycles expended by DBA, determining the relation between quality of solution and number of negotiation cycles for DBA. Previous approaches rely on dedicated nodes for constructing and maintaining streaming trees. This centralization of decision making reduces the robustness and scalability of the system. Through full-decentralization our approach overcomes this disadvantage of previous architectures. The second contribution of this work is to relate the efficiency of DBA to the number of negotiation cycles and to the clustering coefficient network parameter. Our results can be used in the design and control of distributed constraint optimization based negotiation algorithms in networks exhibiting scale-free topologies. The third contribution is the bounded negotiation model where we introduce additional local constraint representing the size of the video playback buffer.

References

Barabasi, A.-L., and Albert, R. 1999. Emergence of scaling in random networks. *Science* 286:509.

Chu, Y.; Rao, S.; Seshan, S.; and Zhang, H. 2002. A case for end system multicast. *IEEE Journal on selected areas in communications* 20(8):1456–1471.

Eisenberg, C., and Faltings, B. 2003. Making the breakout algorithm complete using systematic search. In *Eighteenth International Joint Conference on Artificial Intelligence (IJCAI-2003)*, 1374–1375.

Elhourani, T. 2006. A distributed constraint optimization solution for the application layer video streaming problem. Master's thesis, University of Arizona.

Frei, C., and Faltings, B. 1999. Bandwidth allocation planning in communication networks. In *Symposium on High Speed Networks, IEEE Globecom*, 1473–1477.

Gallager, R. G.; Humblet, P. A.; and Spira, P. M. 1983. A distributed algorithm for minimum-weight spanning trees. *ACM TOPLAS* 5(1):pp. 66–77.

Ganjam, A., and Zhang, H. 2005. Internet multicast video delivery. *Proceedings of the IEEE* 93(1):159–170.

Guo, Y.; Suh, K.; Kurose, J.; and Towsley, D. 2003. P2cast: peer-to-peer patching scheme for vod service. In *WWW '03: Proceedings of the 12th international conference on World Wide Web*, 301–309. ACM Press.

Holme, P., and Kim, B. J. 2002. Growing scale-free networks with tunable clustering. *Physical Review E* 65:026107.

Jannotti, J.; Gifford, D. K.; Johnson, K. L.; and Kaashoek, F. M. a. 2000. Overcast: Reliable Multicasting with an Overlay Network. In *Proceedings of the Fourth USENIX Symposium on Operating Systems Design and Implementation (OSDI 2000)*. Berkeley, CA: USENIX Assoc.

Sarshar, N., and Roychowdhury, V. 2004. Scale-free and stable structures in complex ad hoc networks. *Physical Review E* 69:026101.

Yokoo, M., and Hirayama, K. 1995. Distributed breakout algorithm for solving distributed constraint satisfaction problems. In Lesser, V., ed., *Proceedings of the First International Conference on Multi–Agent Systems*. MIT Press.

Yokoo, M., and Hirayama, K. 2000. Algorithms for distributed constraint satisfaction: A review. *Autonomous Agents and Multi-Agent Systems* 3(2):198–212.

Zhang, X.; Liu, J.; Li, B.; and Yum, Y. S. P. 2005. Coolstreaming/donet: a data-driven overlay network for peer-to-peer live media streaming. In *INFOCOM 2005*, 2102–2111. IEEE.