Improving Performance of ALM Systems with Bayesian Estimation of Peers Dynamics

Ihsan Ullah, Grégory Bonnet, Guillaume Doyen, and Dominique Gaïti

ERA/Institut Charles Delaunay – FRE CNRS 2848
Université de Technologie de Troyes
12 rue Marie Curie – 10000 TROYES – France
{ihsan.ullah,bonnet,doyen,gaiti}@utt.fr

Abstract. P2P-based Application Layer Multicast (ALM) systems have shown a great success for several group communication applications. But some performance problems still await a major breakthrough from these systems for critical services such as live video streaming. For these applications, one of the problems is the dynamics of users' presence since the unannounced departure of a peer causes an interruption in service for all dependent ones. In this paper, we address this issue and propose a probabilistic approach based on Bayesian inference to anticipate users' departures and let peers react proactively. Through simulations and experimental evaluation, we prove that our approach improves significantly the performance of ALM systems with a low overhead.

1 Introduction

Services such as Video-on-Demand (VoD) and live video streaming are becoming very popular due to the availability of broadband access. These services are highly bandwidth consuming and, for the unicast-oriented architecture of the Internet, it is not feasible to deploy them. Therefore to support these services, dedicated group communication mechanisms are required. IP multicast [1] is the natural solution for it but it can not be deployed at the Internet scale due to several reasons given in [2]. On the other hand, Content Delivery Networks (CDNs) [3] distribute the content diffusion load of an origin server to several other servers deployed at strategic geographic locations. Nevertheless they are very expensive and the number of servers must grow with the number of users.

P2P-based Application Layer Multicast (ALM) has emerged as a promising approach to enable group communication applications. ALM systems form an overlay network over the physical one by establishing direct links among end-hosts in a P2P manner. Multicast groups are created and maintained at the application level and content is disseminated without requiring any support from the physical network infrastructure. Hence these systems are easy to deploy and require low cost. Since ALM systems are overlay networks composed of nodes, which are owned and controlled by the end-users, they face problems due to both the overlay itself and end-users. Most of the overlay related issues are addressed in previous works but the problem of peer dynamics, which is user related, requires some more attention. It is not only a hurdle in the success of tree-based systems, but also impacts the performance of other P2P-based systems. In currently deployed systems, the impact of peer dynamics is attempted to be reduced indirectly through the use of buffers of a very large size. The buffer provides the continuity of service but induces a time range delay. In our work, we address this issue and propose a probabilistic approach based on a Bayesian inference which enables ALM nodes to analyze their past presence in the system, and estimate the current session duration. Each node cooperates with other nodes to provide this information to them on request. A node, knowing the estimated session duration of the node it is currently depending upon, can react proactively in such a way to minimize the impact of peer dynamics. Moreover, it improves the performance of ALM systems through reducing the required buffer size and thus delay. Our work does not consider the start-up delay because peer dynamics causes a disruption in service after once it is started.

The rest of the paper is organized as follows. Section 2 outlines the work related to the same issue. In section 3, we present our approach, its validation and a comparison with a previous one. Section 4 deals with the dynamics management algorithms for ALM systems. In section 5, we present the experimental validation we perfomed to evaluate our propopsal in real conditions. Finally, section 6 draws some conclusions and gives directions of the future work.

2 Related Work

Since ALM systems are overlays formed by end-hosts, they face performance problems due to both the overlay topology and the end-users. Topology related problems include heterogenous resources and overlay mismatching with the physical network. These issues are already addressed in the research community and solutions have been given in [4–6]. On the other side, user related problems cover two aspects, which are resource sharing and dynamics of user's presence. Although the former has been addressed in [7–9], the latter requires some more attention especially for content delivery services such as live video streaming and VoD. The related work to this problem mostly consists in studies over different P2P-based video streaming systems, Video-on-Demand systems and telco-managed IPTV systems. These studies provide useful insights towards understanding the user behavior and modeling it in the proper context. We discuss now three of them.

In Yu et al [10], a statistical study of user behavior in a Video-on-Demand system is presented. They show that the number of users watching videos decreases during working hours of the day and increases in the breaks and in the evening times. Similarly, the number of users increases at the weekends, which shows that users watch more videos in their free times. The user arrival rate in this particular VoD matches a modified form of the Poisson distribution. The study of session lengths reveals that about 37% of users go offline within 5 minutes after arrival. On the other hand about 25% of users watch a video for more than 25 minutes. The later group of users is relatively stable and can be utilized to improve the performance. A second analysis study of an IPTV system [11] shows similar patterns as above, namely that users watch more TV during breaks and in the dining hours. But, interestingly, Friday and Saturday got less online users as compared to other days of the week. The study of the IPTV system finds that the number of online users increases gradually but it decreases sharply due to batch departures. Moreover, the arrival and departure processes follow exponential distribution at short timescales. According to a third study of a Peer-to-peer IPTV system [12], user participation shows diurnal patterns with two peaks, one at the mid day and other at the evening time. The number of users does not vary too much on the weekends and on the working days. The study of users' session durations shows the same phenomenon as identified in the work presented above, namely that most of the users stay for a short time.

These user behavior analysis show that a large portion of users stay in the system for a very short time. Hence, on the departure of these users, other depending ones will face a service disruption. On the other hand, another category of users stay for sufficiently long time. Adapting ALM nodes to rely more on the stable part of the peers can reduce the service disruption.

Apart from the analysis studies, a work very close to our's is given in [13, 14]. Authors first present an analysis which statistically reveals that a peer's remaining online duration is positively correlated with its elapsed online duration. Based on this finding they propose a mechanism which chooses a content provider node as the one which elapsed the longest time in the system. To support this function, a centralized authority is used to keep sessions' information of all peers in the system. As a conclusion, considering the elapsed time as a stability indication assumes all recently joining peers unlikely to stay longer. Moreover, a centralized authority is not a scalable and robust solution.

3 Peer's Dynamics Estimation

In this section, we first give a brief overview of a previous estimator we proposed in [15], based on Exponential Moving Average (EMA). Then we present a new estimator, which relies on a Bayesian estimation technique. Before comparing both techniques, we describe the associated decision making mechanism. In the sequel, we term the content providing node as a *provider* and the content receiving node as a *consumer*.

3.1 EMA-based Estimation

In our previous work [15], we used an EMA-based estimation. EMA is a statistical technique which estimates an average from a set of values by giving exponentially decreasing weights to older values. As given in (1a), ES_t is the current session duration, S_{t-1} is the actual duration of last session, ES_{t-1} is the length of the last estimated session and α is a weighting factor in $[0, 1]$. The chosen optimal value of α is 0.7 suggesting to give more weights to the recent session durations. A node having no session history sets ES_0 at its elapsed time as the estimated current session duration. We define two enhacements for the estimator. Firstly, we decrease the estimated session duration by 20%. Such approach is called EMA_{20}. Secondly, we add the trend of the session length in the estimation as given in (1b) and (1c). β is a weighting factor in $[0, 1]$ and we choose 0.6 which gives overall good results. Such an approach is called EMA_{T} and is combined with EMA_{20} in EMA_{T20}.

$$ES_t = \alpha \times S_{t-1} + (1 - \alpha) \times ES_{t-1} \tag{1a}$$

$$T_t = (1 - \beta) \times T_{t-1} + \beta \times (ES_t - E_{t-1}) \tag{1b}$$

$$ES_t = \alpha \times S_{t-1} + (1 - \alpha) \times ES_{t-1} + T_t \tag{1c}$$

3.2 Bayesian Estimation

We propose another estimation technique: a probabilistic approach based on Bayesian inference, which enables an ALM peer to estimate its current session duration in the presence or absence of past sessions' history. In classical statistics, computing a probability distribution requires a set of observations to be infered. In our problem, a peer joining the system for the first time has no previous observations of the user behavior: this very limited number of observations is not appropriate to use classical statistics. Bayesian inference considers a prior probability distribution in the absence of any observation. Moreover, when a new observation arrives, this distribution is updated into posterior probability distribution accordingly. We assume that the prior probability distribution is a uniform one. Hence, we do not make any strong assumption on the user's behavior. Thus the Bayesian model we use is as follows.

Let $T \in \mathbb{R}$ be the maximum possible session duration of a node A. Let us partition T in k time steps $\{t_1, t_2 \dots t_k\}$ such that $t_1 < t_2 \dots t_{k-1} < t_k$. For each time interval $[t_i, t_{i+1}[$ we define a binomial variable α_i. The prior probability of the event ϕ_j, meaning that the next session duration S_i will be at least equal to t_j, can be modeled by the Dirichlet density function as given in (2a) where α_j is the number of observations when $S_i \geqslant t_j$. The estimated prior probability of $(S_i \geqslant t_j)$ is given by (2b) where $|O|$ is the set of already observed session durations. Let us notice that the variables α are not independent. So we define a mechanism to update the $\langle \alpha \rangle$ vector. After observing a set O' of some new session durations, the posterior probability of a session duration S_i to be at least equal to t_j is computed through (2c) where $o_j \subseteq O$ and $|o_j|$ is the number of observations where $S_i \geqslant t_j$.

$$f(\phi_1 \cdots \phi_{k-1} \mid \alpha_1 \cdots \alpha_k) = \frac{\Gamma(\alpha_1 + \alpha_2 + \cdots + \alpha_k)}{\Gamma(\alpha_1)\Gamma(\alpha_2) \cdots \Gamma(\alpha_k)} (\phi^{\alpha_1 - 1} \phi^{\alpha_2 - 1} \cdots \phi^{\alpha_k - 1}) \tag{2a}$$

$$f(\phi_j) = \frac{\alpha_j + 1}{|O| + 2} \tag{2b}$$

$$f(\phi_j | \beta, O) = \frac{\alpha_j + |o_j| + 1}{|O| + |O'| + 2} \tag{2c}$$

Using this estimator each peer updates its posterior probability after each session and keeps a list of probabilities corresponding to each t_j.

3.3 Decision Making Mechanism

The *consumer* node requests the session estimation from the *provider*. If we use the EMA-based estimation, each node estimates the length of its current session and provides it to other nodes on request. If we use the Bayesian estimation, the *consumer* specifies a given probability threshold P_{Th} and the *provider* estimates the length t_i of the current session with (3).

$$t^* = \max_{t_j}(f(\phi_j) \geqslant P_{th}) \tag{3}$$

In both cases, the *provider* also sends its *join time* in order to compute its elapsed and remaining time durations. When the elapsed time duration of the *provider* is reaching to its estimated one, the *consumer* sends a request to all neighbor nodes for their estimated dynamics. The neighbor nodes respond with their estimated time durations, *join times* and capacity status. Capacity status shows the possibility of serving a new consumer. This parameter is important to control the load on stable nodes. On receiving responses from neighbor nodes, the *consumer* selects the node with a capacity to provide content to a new node and having the highest estimated remaining time duration as a new *provider* and leaves the previous one.

3.4 Estimator Evaluation

In order to compare both techniques, we first introduce the session generation model. Then we present a comparison between both approaches.

Sessions Generation. The analysis studies of ALM systems do not show the individual user behavior, instead they give insights of collectively all users in the systems. We choose one of these studies [10] and model the session durations as a lognormal distribution with parameters ($\mu = 2.2, \sigma = 1.5$). Since it shows a collective behavior (and not an individual one), we define two kinds of users: users whose behavior changes from session to session and have uncorrelated session durations and users having a persistent behavior whose sessions are auto-correlated. Firstly, we generate randomly uncorrelated sessions in lognormal form which show the first behavior. Then we apply (4) to make them correlated.

$$S_t = f \times L_t + (1 - f) \times S_{t-1} \tag{4}$$

Here, S_t is the adapted session duration, L_t is the session in lognormal form and S_{t-1} is the last session duration. f is called the auto-correlation factor having a value between 0 and 1. We have chosen the value 0.2 to generate correlated sessions and 1 to generate uncorrelated sessions.

Comparison. To compare the estimators, we generate one thousand session durations and estimate each next session starting from the first one. By going forward the previous sessions are taken as a history. Firstly, we measure *success* which stands for the number of times our estimated session duration is less than or equal to the actual session duration. *Success* is important to react before the departure of the *provider* node. Secondly, we measure the *early reaction time. Early reaction time* is the difference between the actual session duration and estimated duration when the later is less than the former one. This measurement shows how optimal the reactions are. The lower the early reaction time, the lower is the overhead because *consumer* nodes stay over *provider nodes* as long as they can and they minimize the number of reactions. Thirdly, we measure the *error* that is the difference between the actual session duration and estimated session duration whatever the success. We made α, β varying on $]0,1[$ for EMA and $\mathrm{EMA_T}$ and the threshold varying on $]0,1[$ for Bayesian estimation. We kept the most interesting results. For Bayesian estimation, we kept three thresholds: 0.7, 0.8 and 0.9. We show our results in Table 1. In case of uncorrelated sessions, Bayesian estimation performs better on all criteria than all other estimators. By contrast, in case of correlated sessions, the $\mathrm{EMA_{20}}$ and $\mathrm{EMA_{T20}}$-based estimations perform better than Bayesian estimation. Thus the most interesting approach depends on the correlation factor that models the users.

	Uncorrelated sessions			Correlated sessions		
Estimator	Success	Early Reaction Time	Error	Success	Early Reaction Time	Error
EMA	41.7%	32.3576	26.9371	32.3%	7.6659	4.9135
$\mathrm{EMA_{20}}$	45.4%	31.8811	24.4678	**81.7%**	**5.5993**	**4.6996**
$\mathrm{EMA_{T20}}$	58.4%	28.4502	28.79	**87.3%**	**5.3565**	**4.9191**
BAYES_0.7	**69.1%**	**27.0214**	**19.3771**	66.7%	14.2141	10.6827
BAYES_0.8	**76.3%**	**25.6273**	**19.9028**	76.7%	14.3295	11.6206
BAYES_0.9	**83.7%**	**24.3352**	**20.5066**	87.7%	14.9109	13.3018

Table 1. Comparison between EMA-, $\mathrm{EMA_T}$-based and Bayesian estimation

In order to refine our comparison, we analyze the *ratio* of the *early reaction time* to the actual session duration of the *provider* node as well as the *evolution* of the *error*. Figure 1 depicts the frequency distribution of these ratios and the evolution of the error for $\mathrm{EMA_{20}}$, $\mathrm{EMA_{T20}}$ and Bayesian estimator. For Bayesian estimation, we set the threshold at 0.7 which is representative of the results. In case of uncorrelated sessions, the ratio distributions have the same structure for the three estimators. Thus, Bayesian estimation performs better than the others estimators due to its higher success. Concerning the evolution of the error, we can notice that Bayesian estimation does not overestimate the session compared to others estimators. Now in case of correlated sessions, ratio distributions have not the same structure. $\mathrm{EMA_{20}}$ and $\mathrm{EMA_{T20}}$ are better

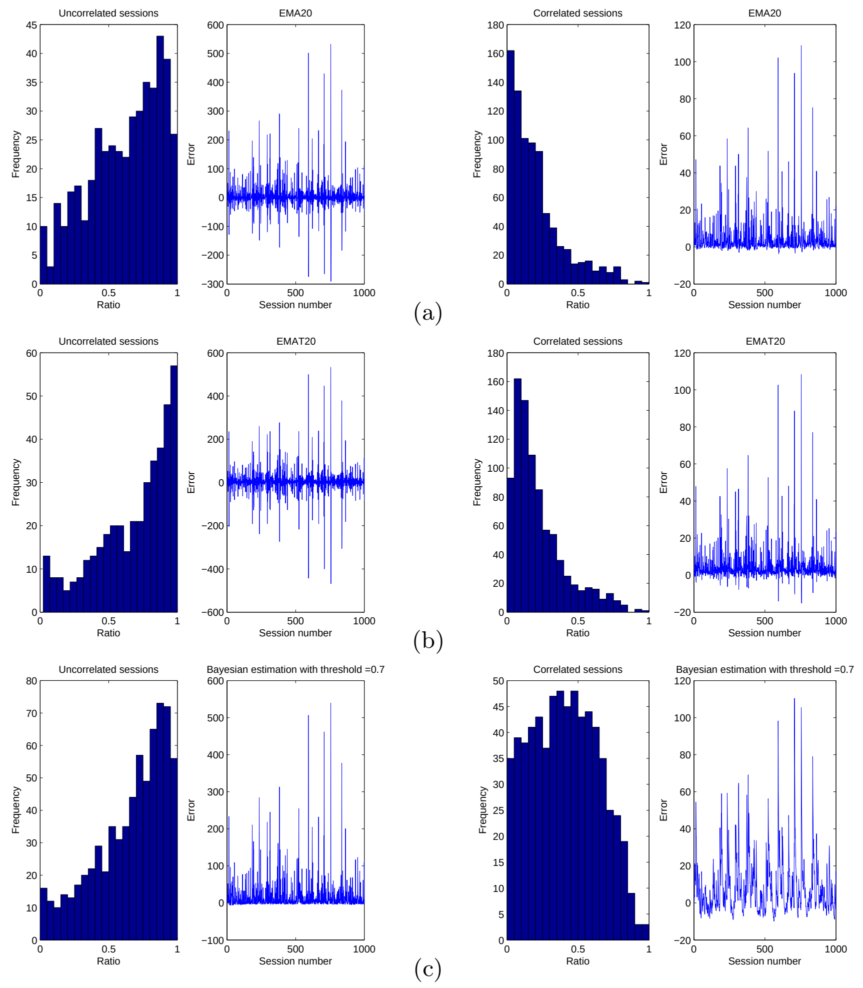

Fig. 1. Ratio distributions and evolution of the error for (a) EMA_{20}-based estimator ; (b) $\mathrm{EMA}_{\mathrm{T}20}$-based estimator ; (c) Bayesian estimator

than Bayesian estimator. EMA_{20} does not over-estimate sessions compare to the others estimators. $\mathrm{EMA}_{\mathrm{T}20}$ slightly overestimate due to the trend that might be incorrect. Bayesian estimation is very sensitive to sudden variations in the session durations and it has difficulty to converge. As previously, Bayesian estimator is better in uncorrelated cases whereas EMA_{20} and $\mathrm{EMA}_{\mathrm{T}20}$-based estimator are better in correlated cases.

To summarize, in case of uncorrelated session durations Bayesian approach does less underestimations and have less ratios than EMA-based approaches. Moreover they achieve more success than EMA-based approaches, therefore Bayesian approach is better than EMA-based approaches for uncorrelated session durations. On the other hand, for strongly correlated session durations, EMA-based approaches are better in terms of success, underestimation and ratio. Thus EMA-based approaches are better for strongly correlated session durations.

4 Dynamics Management Algorithm

In this section, we discuss our mechanism for making application layer multicast systems dynamics-aware. We explain this process through three algorithms. All these algorithms are run by the *consumer* node. Algorithm 1 describes investigation of the current *provider* and scheduling a move. Algorithm 2 allows a *consumer* node to find *potential providers* and request them for their dynamics. To choose a new *provider* node, the consumer runs Algorithm 3.

Algorithm 1 Investigating the provider node and scheduling a move

```
1: send DynamicReq(provider)
2: $providerElapsedTime \leftarrow currentTime - providerResponse.joinTime$
3: $providerRemTime \leftarrow providerResponse.estSession - providerElapsedTime$
4: scheduleMove(providerRemTime)
```

As a node joins the ALM system, it starts Algorithm 1. In the first step, the *consumer* sends a message to its *provider* inquiring for his estimated current session duration and *joinTime* as shown in line 1. The *provider* responds with the two values. After receiving the response, the *consumer* estimates the remaining online time of the *provider* as given in lines 2 and 3. Then it schedules a *move* after the remaining online time of the *provider* as given in line 4. The execution of move is shown in Algorithm 2.

Algorithm 2 Execution of scheduled move

```
1: potentialProviders=getPotentialProviders(groupId, k)
2: for each potentialProvider do
3:    send(DynamicsRequest, potentialProvider)
4: end for
5: for each DynamicsResponse dr do
6:    if (($dr.capacity\_status = true$) then
7:       List.add(dr.source,dr)
8:    end if
9: end for
```

As time scheduled in Algorithm 1 is up, Algorithm 2 starts. In the first step the *consumer* node queries the list of the neighbor nodes and chooses k nodes from it as shown in line 1. We term them the *potential providers.* After getting the *potential provider* nodes, it sends a request to each of them asking their current estimated session duration and *joinTime* as shown in lines 2 and 3. On receiving responses from the *potential providers* each response is analyzed in line 6. If the capacity status of a *potential provider* is true: the number of *consumer* nodes it is currently serving is not reached to the maximum limit, then the *consumer* stores the response in a list as given in line 7. Algorithm 3 uses this list for choosing a new *provider*.

After receiving responses from all the *potential providers*, the *consumer* starts choosing the most stable node. We show this process in Algorithm 3. It chooses a node with maximum remaining online time from the list by analyzing their responses as shown in lines 2 to 5. Then it sends a join request to the node with the longest remaining online time. If the request is accepted, it leaves the current *provider*, otherwise it restarts the process with the next node. All these steps are shown in lines 8 to 14.

Algorithm 3 Choosing a new provider

```
 1: $longestRemTime \leftarrow 0$
 2: for each List.dr do
 3:    if $dr.estSession > longestRemTime$ then
 4:       $logestRemTime \leftarrow dr.estSession$
 5:       $provider \leftarrow dr.source$
 6:    end if
 7: end for
 8: send(JoinRequest, newprovider)
 9: wait()
10: if JoinAckMessage is received then
11:    send(LeaveMessage, previousProvider)
12: else
13:    List.remove(newProvider.dr)
14:    goto line 1
15: end if
```

The overall process in these three algorithms requires ALM nodes to exchange $2(k + 2)$ messages for each *move*, where k is the number of potential provider nodes. The two other exchanges are for leaving the current provider node and joining the new one. This algorithm runs only when the estimated session duration of the current provider node is expired. Therefore, the overall overhead of the approach is low.

5 Experimental Evaluation

Apart from simulations we carry out experiments on an ALM system to validate our approach and notice the performance improvement. We choose Scribe [16] for our experiment, which is a tree-based ALM system that forms a tree for each multicast group enabling the content to be disseminated from the root towards leaf nodes. It is built upon Pastry [17], which is a structured P2P overlay. Scribe does not implement a buffering mechanism. Hence, the departure of a peer interrupts the availability of content to its descendent nodes until the next execution of maintenance process and the finding of a new parent node. Next we discuss the experimental setup and the results we have collected.

5.1 Experimental Setup

We carry out all experiments on a LAN environment consisting of 12 computers connected through a switch. We use the Scribe implementation built upon FreePastry[1]: an open source Java-based implementation of Pastry. In order to coordinate the 12 computers during the experiment, we use DELTA[2] [18], a generic environment for testing and measuring Java-based distributed applications. We launch several nodes on each machine all subscribing to the same Scribe group. The root node stays online throughout the experiment and also works as a source node. All other nodes stay in the system for specified session durations. To avoid the case of many nodes connecting with the stable peers we limit the out-degree of each node to 5 child nodes.

To simulate a user behavior, we base our experiment on the analysis study given in [10], where the session durations of collectively all users follow a log-normal distribution with parameters ($\mu = 2.2, \sigma = 1.5$). Therefore, we generate random session durations in lognormal form with parameters ($\mu = 2.2, \sigma = 1.5$), and assign each node a history of 10 sessions for the estimation of the next session. Similarly, to simulate nodes arrivals into the system, we create them according to the modified Poisson distribution [10] until the maximum limit of nodes on a machine is reached. Concerning the video flow, we use a video trace file of *Jurassic Park* encoded with $H.263$ format with a target bit rate of 256 *kbps* [19]. The trace file has the statistics for one-hour video, therefore each experiment lasts for one hour time. The source generates dummy frames with the sizes specified in the trace file and makes it available to the root node after time intervals synthesized in the trace file.

5.2 Results

Although our simulations consider two kinds of users' behavior (i.e. users with correlated session durations and uncorrelated session durations), for the validation experiments we choose the uncorrelated case which is more difficult to anticipate. We configure DELTA to collect measurement results from each node on each machine after each 5 seconds time. These results contain lost frames, number of attempted moves and number of nodes in the network. Lost frames are those lost by a node between the first frame it receives after its arrival and the last frame it receives before the departure. It shows the impact of peer dynamics on Scribe and the improvement of our proposed approach. An attempted

[1] http://freepastry.org
[2] Distributed Environment for Large-scale Tests of Applications

move is counted when a *consumer* tries to find another *provider* node. Since the overhead of our approach is caused by a move attempt which involves exchange of messages among the nodes, therefore we show them in our results. Number of nodes gives an idea of the network size. We perform three experiments: (1) Scribe without any dynamics anticipation mechanism; (2) with EMA_{20}; (3) with our statistical Bayesian approach. In Figure 2 (a), we show the number of active nodes in the system. We can notice that the number of nodes vary in a similar way and the three experiments follow a similar shape which indicates a similar experimentation behavior. We depict the accumulated frames loss in Figure 2 (b). Here we can see that without any estimation Scribe looses the highest number of frames with a value equal to $411,581$. On the other hand, EMA_{20} performs better since it reduces the frame loss by 33%. Finally, the Bayesian approach reduces the Scribe frames loss by 40%. It clearly shows a significant performance improvement of the Bayesian estimation in Scribe. Now, concerning the overhead, we show the number of attempted moves for both EMA_{20} and Bayesian-based approach. We consider moves attempts instead of the successful moves because all the attempted moves are not successful in Scribe due to three reasons: (1) there is no *potential provider* which fulfils the Pastry's prefix routing constraint (2) the capacity of all the available *potential providers* is reached to the maximum limit; (3) the estimated remaining session durations of available potential providers are reached to zero. As shown in Figure 2 (c), EMA_{20} attempts 5.8 moves, while Bayesian-based approach attempts 7.8 moves every minute in the overall community. The average moves per node in one hour are 3.74 for Bayesian approach. although it is more than the EMA_{20}-based approach but still it is a low overhead.

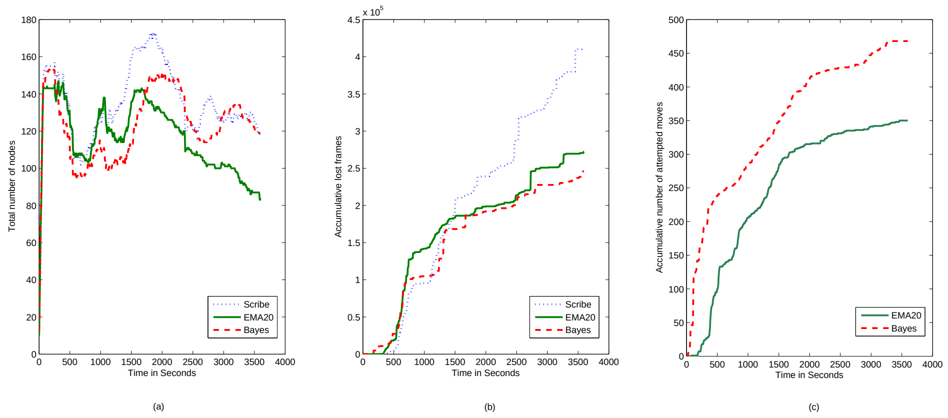

Fig. 2. Experimental results for uncorrelated sessions. (a) Number of nodes ; (b) Frames loss; (c) Number of attempted moves

6 Conclusion and Future Work

In this paper we address the problem of peer dynamics in P2P-based ALM systems. We propose a statistical approach based on a Bayesian inference to anticipate the departure of a peer by considering its online presence in the past. We also compare it with previously proposed EMA-based approaches. Moreover we proposed a generic algorithm to make ALM systems dynamics-aware. We validated our proposal with both simulations and experimentation. Simulations validate the accuracy of the estimator. Experimentations demonstrate the performance improvement of an ALM system in a context very close to what happens in a concrete P2P broadcasting system. Especially, our experimental results show that the performance of ALM systems can be improved by reducing significantly the number of frames lost by peers during a movie broadcast. Thus, benefits of our approach are twofold: (1) it minimizes the impact of peer dynamics on the performance of ALM systems; (2) it reduces the size of the required buffer in P2P-base video streaming systems, which will decrease the delay. Our approach is not limited to ALM systems. It can also be applied to other P2P systems where dynamics has an impact on the performance of the system.

Concerning short time future work, we will perform the same experiment for the users with correlated session durations and will test our approach on other types of ALM systems (e.g. mesh-based). We will also work on integration of other impacting parameters like type of application and time of the day with a Bayesian network. After that, we will consider other performance issues like heterogeneity of resources and overlay mismatching together with the peer dynamics. Some of performance parameters related to these issues are conflicting in nature and addressing them altogether will be interesting.

References

1. Deering, S., Cheriton, D.: Multicast routing in datagram internetworks and extended lans. In: ACM Transactions on Computer Systems. (1990) 85–110
2. Chu, Y., Seshan, S., Zhang, H.: A case for end system multicast. IEEE Journal on Selected Areas in Communication (JSAC) **20**(8) (2002)
3. Dilley, J., Maggs, B., Parikh, J., Prokop, H., Sitaraman, R., Weihl, B.: Globally distributed content delivery. In: Internet Computing, IEEE (2002) 50 – 58
4. Kostic, D., Rodriguez, A., Albrecht, J., Vahdat, A.: Bullet: high bandwidth data dissemination using an overlay mesh. In: Proceedings of the Nineteenth ACM Symposium on Operating Systems Principles, ACM Press (2003) 282–297
5. Xiao, L., Liu, Y., Ni, L.M.: Improving unstructured peer-to-peer systems by adaptive connection establishment. IEEE Trans. Comput. **54**(9) (2005) 1091–1103
6. Zhao, J., Lu, J.: Solving overlay mismatching of unstructured p2p networks using physical locality information. In: Proceedings of the 6th International Conference on P2P Computing, IEEE (2006)
7. Castro, M., Druschel, P., Rowstron, A., Kermarrec, A., Singh, A., Nandi, A.: Splitstream: high-bandwidth multicast in cooperative environments. In: 19th ACM Symposium on Operating Systems Principles. (2003)

8. Ngan, T., Wallach, D., Druschel, P.: Incentives-compatible peer-to-peer multicast. In: Proceedings of the Second Workshop on the Economics of Peer-to-Peer Systems. (2004)
9. Liuy, Z., Shen, Y., Panwar, S.S., Ross, K.W., Wang, Y.: Using layered video to provide incentives in p2p live streaming. In: Proceedings of the 2007 workshop on Peer-to-peer streaming and IP-TV, ACM (2007) 311–316
10. Yu, H., Zheng, D., Zhao, B.Y., Zheng, W.: Understanding user behavior in large-scale video-on-demand systems. SIGOPS Oper. Syst. Rev. **40**(4) (2006) 333–344
11. Cha, M., Rodriguez, P., Crowcroft, J., Moon, S., Amatriain, X.: Watching television over an ip network. In: Proceedings of the 8th ACM SIGCOMM Conference on Internet Measurement, ACM (2008) 71–84
12. Hei, X., Liang, C., Liang, J., Liu, Y., Ross, K.W.: A measurement study of a large-scale P2P IPTV system. IEEE Transactions on Multimedia (2007)
13. Tang, Y., Sun, L., Luo, J.G., Zhong, Y.: Characterizing user behavior to improve quality of streaming service over P2P networks. In: Advances in Multimedia Information Processing. (2006) 175–184
14. Tang, Y., Sun, L., Luo, J.G., Yang, S., Q., Zhong, Y.: Improving quality of live streaming service over P2P networks with user behavior model. In: Advances in Multimedia Modeling, Springer (2007) 333–342
15. Ullah, I., Doyen, G., Khatoun, R., Gaîti, D.: A decentralized approach to make application layer multicast systems dynamics-aware. In: 10èmes journées doctorales en informatique et réseaux, UTBM (2009) 43–48
16. Castro, M., Rowstron, A., Kermarrec, A.M., Druschel, P.: Scribe: a large-scale and decentralised application-level multicast infrastructure. Journal on Selected Areas in Communication **20**(8) (2002)
17. Rowstron, A., Druschel, P.: Pastry: scalable, distributed object location and routing for large-scale peer-to-peer systems. In: IFIP/ACM International Conference on Distributed Systems Platforms. (2001) 329–350
18. Doyen, G., Ploix, A., Lemercier, M., Khatoun, R.: Towards a generic environment for the large-scale evaluation of peer-to-peer protocols. In: Networking and Electronic Commerce Research Conference 2008. (2008)
19. Fitzek, F.H.P., Reisslein, M.: MPEG-4 and H.263 video traces for network performance evaluation. IEEE Network **15**(6) (2001) 40–54