Distributed Interactive Application 을 위한 우선 순위 기반 오버레이 멀티캐스트

이형옥*, 남지승*
*전남대학교 전자컴퓨터공학과
e-mail:narcis99@nate.com

Priority-based Overlay Multicast for Distributed Interactive Application

Hyung-Ok Lee*, Ji-Seung Nam*
*Dept of Electronic-Computer Engineering Chonnam University

요 약

Applying Application-Level Multicast technology (ALM) to Distributed Interactive Applications (DIAs) becomes more and more popular. Especially for DIAs embedded priority that the sender forwards data to receivers due to their respective priorities. The priority-based directed minimum spanning tree (PST) algorithm was designed for these DIAs. However, the PST has no efficient priority selection and filtering mechanism. The system will consume a tremendous amount of resource for reconstructing distribution tree and becomes unstable and unscalable. In this paper, First, We propose a novel priority-based application level multicast algorithm: Predict-and-Quantize for Priority with directed minimum Spanning Tree (PQPST), which can efficiently predict efficient priorities for the receivers and quantize the predicted priorities to build the multicast distribution tree. Second, we propose Priority Discrepancy Heuristic Mechanism (PDHM), which sets different thresholds of priority discrepancy within the priority discrepancy interval to control the distribution tree construction can efficiently decrease the repeated distribution tree construction, and we get the best heuristic priority discrepancy interval by PQPST. According to the simulation results, the PQPST and PDHM can efficiently improve the performance of the PST algorithm.

1. Introduction

The emergence of new Internet-based applications --- Distributed Interactive Applications (DIAs) such as IPTV, Teleconferencing and NetGames generally require multicast capability for a successful operation, and the need for efficient support of one- to many and many – to –many applications, Moreover, DIAs allow a group of users connected via a computer network to communicate and collaborate in order to manipulate and accomplish a common task [1]. DIAs embedded priority is a special set of DIAs, because the sender node forwards data to receivers due to their respective priorities. In recent years, many Application Level Multicast protocols have been proposed [2]. Among them, ALMI [3] and Yoid [4] have been designed for multi-source application while Narada [5] and NICE [6] are for single source application However, most of them cannot be of significant use to these DIAs, whose receivers have chances of getting the packets with different priorities. Priority-based directed minimum spanning tree (PST) [7] is the original application-level multicast protocol for these DIAs. A typical example is the network multi-player games. Nevertheless, when the scale increases, due to lack of efficient priority selection mechanism, PST will consume undesired amount of resource enough to make a system unstable and unscalable.

2. Predict-and-quantize priority with MST(PQPST)

PQPST algorithm is based on the traditional PST algorithm. To enhance the performance of the existing PST applied in DIAs, the main work is that PQPST must get the priority that can make new distribution tree before implementing the application protocol. Hence, PQPST considers priority prediction and priority quantization to construct the multicast distribution tree.

In fact, not every priority changing from 0 to 1 can rebuild the significant distribution tree. In many cases, after rebuilding the distribution tree, because of priority changed, I find the new distribution tree is the same or little different from the previous one, and in some exceptional case lower priority can build more direct path. So, it is not worth to rebuild the distribution tree whenever application-level priorities change.

PST algorithm lacks efficient priority selection mechanism that can predict the efficient priority to be worth rebuilding the multicast distribution tree. It is, in addition, very costly to recalculate the distribution tree whenever application-level priorities change, especially when the size increases.

Our PQPST algorithm proposes an efficient priority selection mechanism as described in this section. Obviously,

when receiver's priority changes, the receiver node will change to different paths whose delay must contain the delay of k-shortest paths for the relative receiver. Therefore, we can use the respective delay of k shortest paths and apply Formula (1) to predict the priorities of the receivers for relative k shortest paths.

According to the PST, if receiver **j** in the long path wants to move to shorter path, it needs to increase the priority enough to make $w'(e_{ij}) > w'(e_{kj})$, where $w'(e_{kj})$ is the modified cost of the direct edge between receiver node **j** and the upper node **k** in the shorter path. That means the predicted priority is the minimum $p(j)$ that satisfies the Formula (1) below, so according to Formula (2) I can predict the priority.

$$w(e\_ij)+p(j)w(e\_sj)>w(e\_kj)+p(j)w(e\_sk) \quad (1)$$

$$p(j) = \frac{w(e_{si}) - w(e_{sk})}{w(e_{kj}) - w(e_{ij})} \quad (2)$$

3. Simulation and results analysis

To simulate the proposed algorithm, we have implemented a simple game simulator – Billiards based event to evaluate the performance of our PQPST algorithm compared to PST algorithm. The red ball is the shooting ball and it does not join the multicast session, and the white balls represent the source node and receiver nodes. In order to simplify the simulation, we adopt the single source model.

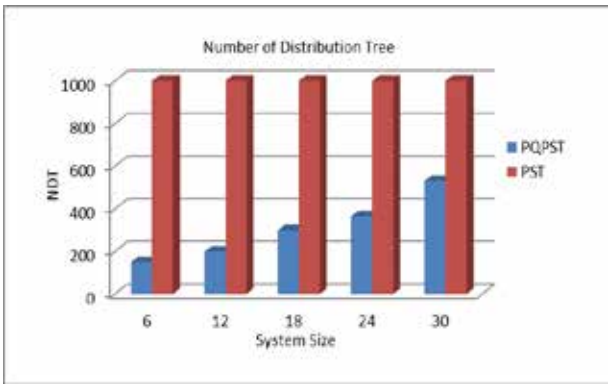
According to the position of relative receiver in the playing area, we calculate the relative priority for every receiver after each event happened in the playing area.

The priority p is calculated depending on their distance d(s,j) to the source node: p(j) = d(s, j)/R, where R is the maximum distance between source node and receiver in application. Furthermore, we use respective receiver's quantized priority groups to divide the circularity area into some concentric circles whose radiuses are the products of each efficient priority by R. Therefore, the player moving from one doughnut area to other doughnut area is equivalent to the priority changing from one group to other group. Obviously, it will greatly simplify the simulation

The metrics of evaluating the performance of the proposed PQPST and the existing PST are described as follows:

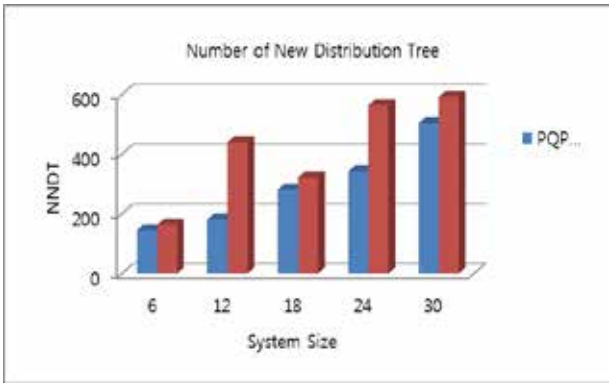
1. Number of distribution tree (NDT): NDT represents the total number of distribution trees which were built during the whole simulation.
2. Number of new distribution tree (NNDT): NNDT represents the total number of distribution trees which have different delay and were built adjacently during the whole simulation
3. New distribution tree reconstruction rate (NDTR): NDTR represents the quotient of NNDT by NDT.

From Fig.1, We can find that the NDT of PQPST is less than 50% of the PST's NDT in each system, especially, when the system size is small. This means PQPST decreases the large number of rebuilding distribution tree. Obviously, PQPST can efficiently decrease the cost required for reconstructing extra distribution trees. In fact, PQPST delete all the rebuilding distribution trees that are led by exceptional priority and non-efficient priority (cannot make new distribution tree).

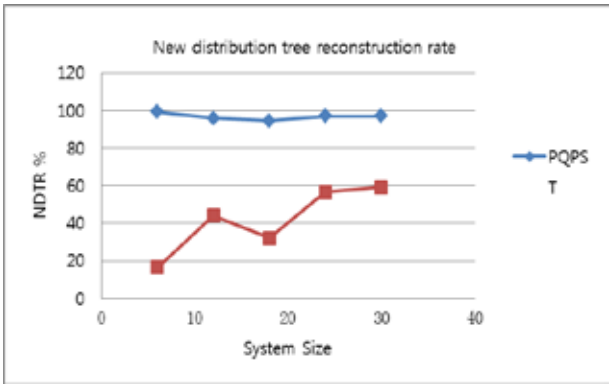


(Figure 1) Comparison of NDT in PQPST and PST

As shown in Fig.2, in each system size 6, 18 and 30 the NNDT of PQPST are almost the same with PST's. However, in system size 12 and 24 the NNDT of PQPST are obviously less than the PST's. There are two reasons. First, PQPST bases on kSP algorithm to predict the priority, and kSP algorithm doesn't consider all nodes for relative receiver node. So PQPST ignores a few priorities, and cannot make the entire new distribution tree. Second, NNDT of PST contains the distribution tree led by exceptional priority, and those distribution trees are bad for application.



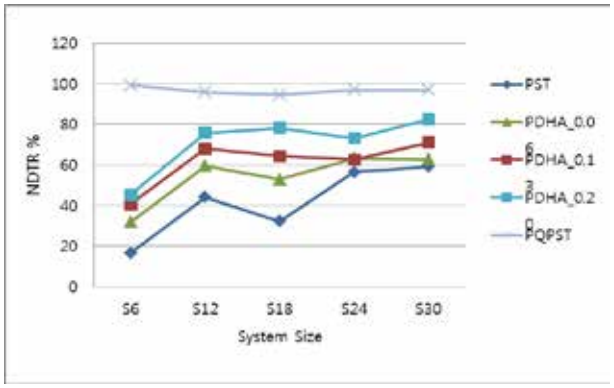
(Figure 2) Comparison of NNDT in PQPST and PST



(Figure 3) Comparison of NDTR in PQPST and PST

According to Fig.3, Most NDTR of the PQPST are almost reach 100%. Because PQPST algorithm can predict almost priority that can construct the new distribution trees (NDT), and reconstructs distribution trees only when the application priority match the predicted priority.

However, the PST's in most system are less than 60%. Obviously, PQPST provides more efficient algorithm for reconstructing the distribution tree. Namely, Compared to PST, PQPST consumes much less system resource to build the distribution tree and finally keep the system more stable and scalable.



(Figure 4) Comparison of NDTR for each Dp

Fig.4 shows comparison of NDTR among PDHA, PST and PQPST. PQPST has the highest NDTR, and PDHA with three Dps (0.06, 0.13 and 0.2) all have higher NDTR than PST's. Even though PST can make the entire efficient distribution trees, it has the lowest of new tree construction rate, and meanwhile, Even though PQSPST can't make the entire efficient distribution trees, it has the highest of new distribution tree construction rate. It can prevent the exceptional priority and non-efficient priority from building the non-efficient tree that can make system unstable and unscalable. Otherwise, The NDTR of PDHA falls between PST's and PQPST's. Hence, PDHA is able to balance the NDTR and NNDT within the priority discrepancy interval. The same meaning is that PDHA can balance the PST and PQPST.

Consequently, my proposed PQPST and PDHM are all able to improve the performance of the PST algorithm in different aspect. To be worth mention these two methods have their own characteristics, and they can be adopted for different systems by respective requirements.

4. Conclusion and future work

In this paper, First, we proposed a novel algorithm named PQPST for Distribution Interactive Applications (DIAs). It uses both quantized priority and delay to construct multicast distribution trees. Second, we propose Priority Discrepancy Heuristic Mechanism (PDHM), which sets different thresholds of priority within the priority discrepancy interval to control the distribution tree construction. Even though the existing PST algorithm is the original protocol that uses priority to build distribution trees for DIAs embedded the priority, it lacks efficient priority selection mechanism. PST algorithm can build all efficient distribution trees, but it has very low new distribution tree reconstruction rate. This shortcoming leads PST to have heavy cost in recalculating the distribution tree and it is prone to make systems unstable and unscalable.

According to the simulation results presented, our proposed PQPST algorithm has the good ability of efficient priority prediction and can control distribution tree reconstruction efficiently. Moreover, PQPST has the best new distribution tree reconstruction rate. Hence, PQPST can solve this problem well, and at last keep system scalable and stable. However, PQPST cannot predict all the efficient priority. Yet, the PDHM can well balance the PST and PQPST within the priority discrepancy interval. Consequently, my proposed PQPST and PDHM are all able to improve the performance of the PST algorithm. Moreover, these two methods have their own advantages and disadvantages. They can be adopted for different systems by respective requirements.

In future, we first plan to improve the k-Shortest path algorithm to predict more efficient priority. Second, we want to adopt new heuristic approach to rank the priority efficiency. Third, we intend to improve the simulation by considering source-share model and by using more preference metrics, and consider other network properties (e.g. bandwidth, load balance etc.) to construct the distribution tree and improve the system performance. At last, we try to adopt fire-new algorithm to replace the PST algorithm and my proposed algorithm in this paper, because these

This research was supported by the MKE(The Ministry of Knowledge Economy), Korea, under the ITRC(Information Technology Research Center) support program (NIPA-2013-H0301-13-1006) supervised by the NIPA(National IT Industry Promotion Agency)

References

[1] U. M. Borghoff and J. H. Schlichter, Computer-Supported Cooperative Work. Springer, Berlin, Heidelberg, New York, 2000.

[2] Hosseini, D. T. Ahmed, S. Shirmohammadi, and N. D. Georganas. A survey of application-layer multicast protocols. IEEE Commun. Surveys and Tutiruals, 2007.

[3] D. Pendarakis, S. Shi, D. Verma, and M. Waldvogel, "ALMI: An application level multicast infrastructure," 3rd Usenix Symp. Int'l Technol. and Syst., Mar. 2001, pp. 49-60.

[4] P. Francis. "Yoid: Extending the Internet Multicast Architecture," unrefereed report, available at http://www.icir.org/yoid/docs/yoidArch.ps.gz, Apr. 2000.

[5] Y. Chu, S. Rao, and H. Zhang. "A Case For End-System Multicast," In Proc. ACM SIGMETRICS, Santa Clara, CA, USA, June 2000.

[6] D. A. Tran, K. A. Hua, and T. T. Do, "A peer-to-peer architecture for media streaming," IEEE J. Sel. Areas Commun., vol. 22, no. 1, Jan. 2004, pp. 121-133.

[7] J. Vogel, J. Widmer, D. Farin, M. Mauve, W. Effelsberg. "Priority Based Distribution Trees for Application Level Multicast," ACM NetGames, Redwood City, CA, 2003, pp. 148-157.