Efficient Prioritized Congestion Management for Social Network Based Live Sharing

Yili Gong†, Wenjie Wang‡ and Chi Harold Liu‡
†School of Computer, Wuhan University, Wuhan, China
‡IBM Research - China, Beijing, China
†yiligong@whu.edu.cn, ‡{wenjwang, chiliu}@cn.ibm.com

***Abstract*—In this paper, we explicitly consider the scenario of supporting applications with high-bandwidth and low-latency requirements in mobile social networks (MSNs), by leveraging the existing social network sites. Our solution does not assume any centralized server who coordinates the data storage, access, and group management. We address the fundamental challenge supporting such applications, namely: the limited bandwidth availability and intense bandwidth requirement that may hinder the deployment of these applications. We propose a link-based congestion management (LCM) mechanism to adjust the source rate of each stream, and a prioritized flow management mechanism composed of a novel concept of "stress weight" and its associated utility function. Extensive simulation results show that our proposal achieves 50% less rate reduction under the severe congestions, and the achievable source rate of the proposed LCM scheme is over one order of magnitude higher than the alternative approach.**

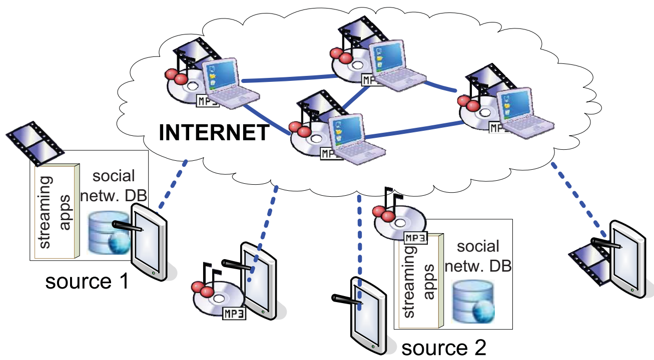

Fig. 1. An illustrative example of streaming media via MSNs, where two social groups are presented and one MP3 music and one video clip are shared.

I. Introduction

Mobile social networks (MSNs), leveraging the rapidly increasing amount of mobile phone users, especially smartphones [1], the social networking and media sharing sites (like Facebook, MySpace, YouTube), have exploded in popularity very recently. The traditional MSNs provide a variety of mechanisms for users to share rich sets of contextual data with their community group(s), such as mobile live streaming [2]. However, it becomes impractical for each application to build its own network from both business and user perspectives. A common approach is to rely on existing social networks to build extended applications, as existing social networks usually offer only social-based small games and opinion sharing for asynchronous interactions among users. Furthermore, with the rapid development of mobile Internet and social networks, there are emerging needs for applications to allow friends to participate in *direct* and *concurrent* interactions, such as game watching, video sharing, even virtual parties. In this paper, we focus on the social based applications that provides the realtime interactions among users, which usually requires high bandwidth and low latency supports.

Nevertheless, the current extended applications usually rely on a separate centralized server to manage the data subscriptions and deliveries among targeted interested group with certain privacy control. However, for high-bandwidth applications, they usually require significant infrastructure support for a large scale of MSNs. One alternative is to integrate with existing video website offerings such as YouTube and webcamnow [3], but at the expense of extra cost and usage hassle. Furthermore, privacy control, flexibility, and scalability concerns may complicate the design and result in the deteriorated user experience, and thus not appropriate for the purpose of mobile live sharing.

In this paper, we propose a *distributed* online live sharing mechanism for applications like streaming among mobile terminals, complemented by existing MSN services. We assume no centralized server managing and controlling the subscriptions. Supporting such applications over MSNs lively imposes a challenge: *how to manage the intensive bandwidth requirements to achieve the good perceived quality-of-service (QoS) for multiple flows from various friends.* High bandwidth applications requires end hosts to use the efficient congestion control mechanisms to avoid network overload, reduce delay and packet loss, especially with the mixture of mobile networks and fixed Internet. Additionally, it is critical to allocate bandwidth among multiple streams for different receivers to achieve the desired QoS.

Fig. 1 shows as an illustrative example, where smartphone users upload their live media to the Internet via existing access networks like 3G, and at the sender side, each media is explicitly tagged with the preferred group of receivers, who are identified through the social circle, like Facebook. Then,

This work is supported by the Major National Science and Technology Special Projects (Grant No. 2010ZX03004-003-03) and the National Natural Science Foundation of China (Grant No. 60773058).

more powerful computing terminals or the *powered nodes*, like workstations and laptops, help forward the media to the group of receivers, by leveraging their relatively sufficient bandwidth resources. For privacy protections, these powered node are also restricted by the social circle. We also explicitly consider the case where multiple streams coexisting in the network, and formalize the bandwidth allocation among streams as a constrained optimization problem based on streams' QoS and capacity status.

Our contributions are four folds. First, we propose a *link-based congestion management* (LCM) algorithm for multicast, which carries out the congestion control on each individual multicast link within the cloud of the Internet, instead of on the whole route from the sender to multiple receivers. Second, to facilitate the stream forwarding with the aid of the online helpers to multiple receivers, we propose a novel concept of the "stress weight" to differentiate the stream priorities. Third, an *weighted flow control* (WFC) mechanism is proposed for each node to distributedly allocate the bandwidth among streams based on the defined stress weights in real-time. The optimality is achieved by a penalty-based utility function, and more importantly, a dynamic topology control scheme that renders the low priority streams to switch away under the severe congestion conditions. Fourth, we perform simulations on two real Internet topologies, indicating that the achievable source rate of the proposed LCM scheme is over one order of magnitude higher than the existing scheme like path-based congestion management (PCM) for a group of 256 nodes. Furthermore, our WFC scheme can effectively reduce the data rate of 50% more than the one without the WFC scheme, for both high priority and low priority streams under severe congestion conditions.

The rest of the paper is structured as follows. Section II highlights the related research activities. After introducing the bandwidth estimation scheme in Section III, we describe our LCM scheme in Section IV, and the WFC scheme in Section V. We then evaluate our proposed mechanisms in Section VI, and conclude in Section VII.

II. Related Work

Dissemination of dynamic content over a MSN was studied in [4], where service providers allocate their downlink rate to maximize the aggregated utility over all users. Adaptive peer-to-peer (P2P) content sharing between users of a MSN was proposed in [5], whose target environment is a disconnected, delay tolerant MANET with no help from the wired Internet. We have not found any system so far focusing on the sharing of live contents based on external social circles.

Application-wise, Stickam, Kyte, and many others [2], provide mobile live streaming services by leveraging the built-in GPS to auto-post your live video clips to multiple video sharing sites (YouTube and others) and microblogging services (Twitter, etc.). Unlike our proposal, these works utilize external servers for archival content delivery and separate networks for access control (if there is any).

Our proposed solution assumes the usage of a dynamic P2P-based multicast protocol. A large number of protocols [6], [7], [8] have been proposed to support the streaming media over the overlay networks. We leverage these protocols to construct and maintain the overlays, but control the membership of the multicast group with social data. Furthermore, we also estimate the congestion rate of the stream overlay tree, especially for the benefit of the wireless and mobile networks.

As one of the key components of our system, multicast congestion control have been vividly investigated [9], [10], [11]. These mechanisms are specifically designed for IP multicast, not suitable for application-level multicast. For application-level multicast, one can take advantage of the additional application level information for better adaptations [12], like the cooperation among the intermediate nodes and the awareness of the importance of payload contents. In this paper, we particularly utilize the collaboration among peers on each virtual link, and propose the novel approach to determine the stream priority by calculating its overall capacity stress based on the social data.

Different from many existing research regarding the support of multiple streams [13], [14], we explicitly explore the benefits of managing source rate for different streams in a distributed fashion based on an automatic prioritizing metric. We emphasize the preferential fairness according to the capacity situation facing by each stream to optimize the user-perceived performance, rather than ensuring the fairness in a fashion like TCP or TFRC.

III. System Model

Consider a P2P-based live sharing network that comprises a set of mobile handsets and a set of powered nodes, where mobile nodes are handhold devices that are connected to the Internet via mobile network such as 3G, and powered nodes are PCs and laptops with traditional broadband connections to the Internet. Source node (or the mobile handset) generates the stream $q \in \mathcal{Q}$ to a set of intended receivers $\mathcal{S}_q^r \in \mathcal{S}$, where $\mathcal{S}$ denotes the overall node set that includes both the mobile handsets and the powered nodes, and $\mathcal{Q}$ is the collection of concurrent streams being serviced. Furthermore, let $\mathcal{S}_q^p \in \mathcal{S}, \forall q \in \mathcal{Q}$ denote the set of powered nodes, served as the helpers of stream q based on the obtained social information. For each node $i \in \mathcal{S}$, let k_q^i be the number of down-stream friends it forwards content to for stream q. Finally, on each virtual link $(i, j) \in \mathcal{S}$, we periodically measure its associated packet delay and packet loss rate information as D_{ij} and L_{ij} to compute the supported throughput $T_{ij}, \forall(i, j) \in \mathcal{S}$ (see Section IV-B).

We next summarize the proposed prioritized congestion management approach. In general, we allow the online friends $\mathcal{S}_q^p$ of the source node of a stream q to help forward the stream by special status update posted to the online social network. In fact, the availability of the stream and the initial powered nodes are bootstrapped with such mechanism. More specifically, for stream q, the source node first measures the bottleneck throughput among all routes from the source to a set of intended receivers, and use it as the sending rate to perform the rate control. Then, when each powered node receives the traffic from its upstream node, it runs a utility-based optimization problem locally to optimally allocate the amount of the bandwidth for each stream if there are multiple concurrent streams. The utility function is based on a novel concept of the "stress weight" of each stream, which measures its capability of forwarding the streams based on the number of intended mobile receivers and helpers within the Internet. Detailed descriptions will be introduced in Section V-A. Finally, under the severe congestion conditions, we introduce a way to prioritize the streams based on their computed stress weights, see Section V-B.

IV. Link-based Congestion Management

With our proposed distributed design, a powered node on the Internet may forward multiple streams for his/her friends who share the common interests. Nevertheless, the node is limited in the uplink bandwidth due to the asymmetric nature of the dominant broadband solutions, such as ADSL and cable modems. Therefore, an efficient congestion control mechanism is desired to allow the source node (or the mobile handset) to adjust its sending rate accordingly.

To facilitate the accurate estimation of the network conditions, we allow the source node to leverage the feedback from the outcome of the used congestion control mechanism. These measurements can help avoid packet queuing in the powered nodes that may eventually lead to buffer overload or packet dropping. Furthermore, for applications supporting live streaming and user interactions, queuing in the powered nodes would result in dissatisfactory user experience like unrecognizable video or audio signals. In the mean time, the nature of such media-based applications allows the adjustment of its own bitrate without losing much of the user experience, e.g. the video resolutions and voice sampling rate. In this section, we first discuss the challenges in the P2P-based multicast congestion control in general, and then move to the description of our proposed LCM scheme.

A. Analysis

As discussed earlier, previous studies mostly focus on the path-based multicast congestion control [15], [9], [10], where a receiver measures the network condition of a root path and estimates its available bandwidth along the root path. Then, the source node uses the minimum reported rates from all receivers as its congestion rate. Different from all these, we adopt a link-based multicast congestion control (LCM) approach primarily for the following reasons. At the application level, the increased delay and packet loss rate on all virtual links significantly lower the perceived available bandwidth on the end-to-end root path, however critical for the streaming applications. Additionally, unlike PCM, which is designed for IP-multicast where applications do not have access to the powered nodes, our system allows the interactions with all powered nodes, the source, and receivers. Therefore, our approach provides additional flexibility to the design of the congestion control protocol, by leveraging the available link-level information.

B. Proposed LCM Design

In our proposed LCM mechanism, the congestion control is performed on each virtual link (i, j), instead of the whole root path. As mentioned earlier, each node $i \in \mathcal{S}_q^p$ in the multicast tree (who are also identified as helpers of stream q) measures its upstream packet delay D_{ij} and packet loss rate L_{ij} associated with each outgoing virtual link to its upstream node. Then, based on the measured delay and loss information, node i calculates its link-based available bandwidth as [15]:

$$T_{ij} = \frac{\sigma}{D_{ij}\sqrt{\frac{2L_{ij}}{3}} + 3\sqrt{\frac{3L_{ij}}{8}}L_{ij}\left(1+32L_{ij}^2\right)\tau_{ij}}, \forall i, j \in \mathcal{S}, \quad (1)$$

where σ, τ_{ij} denote the packet size (as the constant) and the the timeout value over the virtual link (i, j), and let D_{ij}, L_{ij} be the measured delay and packet loss rate over (i, j), respectively. Then, T_{ij} is reported back to the upstream node. After collecting all available bandwidth information from node i's children, the upstream node forwards this information to the root node along the multicast tree. We use such a level-by-level aggregation mechanism to eliminate the feedback implosion. The root node, i.e., the source node, then takes the minimum value among all reported available bandwidth as its sending rate, as:

$$T_q = \min_{i,j \in \mathcal{S}_q^p} T_{ij}, \quad \forall q \in \mathcal{Q}. \quad (2)$$

We notice that the derived T_q does not aggressively seek out the available bandwidth, but it increases the sending rate gradually in response to a decrease in packet loss rate L_{ij}. Furthermore, it does not halve sending rate upon single loss event, but do so in response to several successive loss events. We finally derive the per-node throughput $T_i = \sum_j T_{ij}, \forall i \in \mathcal{S}$, if j denotes the upstream nodes of i.

To summarize, LCM helps lower down the increase of packet delay and packet loss rate by adjusting the sending rate with the knowledge of the quality of individual virtual link along the root path. Therefore, the available bandwidth seen by the end-to-end congestion control protocol increases, and its benefits are shown in the simulation results in Section VI.

V. Prioritized Flow Management

In the section, we propose an *weighted flow control* (WFC) mechanism for each powered node to *iteratively* and *distributedly* achieve the optimal bandwidth allocation among streams by dynamically changing the multicast tree topology and minimizing the defined penalty utilities. The difficulties of performing efficient bandwidth allocation for each node not only comes from the concurrent streams with different bandwidth requirements, but also the number of helpers within the Internet and the number of intended mobile receivers. Therefore, under the severe congestion conditions, although existing mechanisms reduce the data rates of all concurrent streams proportionally, or to provide the "soft" QoS requirements, they all lack of a clear notion to correlate and distinguish stream priorities and their associated social information. In other words, in a distributed solution, more friends could potentially lower down the priority of competing the bandwidth resources for each node on average. Therefore, a better flow management scheme should be able to overcome these shortcomings and optimally allocate the limited available resources during congestion period, to streams with higher priority, while leaving less resources for the low priority ones.

The overall proposed approach is composed of five design elements. First is the proposal of the stress weight, which defines the "priority" of each stream considering both the number of helpers and the number of receivers simultaneously. Second is the proposal of the penalty-based utility function to achieve certain fairness among streams. Third is the constrained optimization problem for optimal bandwidth allocation. Lastly is the distributed and dynamic topology control with which the powered node is able to change switch connection to the multicast tree to alleviate the congestion conditions. And finally is an iterative design approach where for each node the calculated stress weight and utility are used as the inputs to the optimization problem, and if the bandwidth of any stream is not satisfactory as an output, then the distributed topology control of that node is performed.

A. Stress Weight and Utility Function

To facilitate the efficient bandwidth allocation based on the associated social and stream-related information, we first introduce a unique metric, the "stress weight" of stream $q \in \mathcal{Q}$, denoted as ω_q as the ratio between $|\mathcal{S}_q^r|$ and $|\mathcal{S}_q^p|$, i.e.,

$$\omega_q = \frac{|\mathcal{S}_q^r|}{|\mathcal{S}_q^p|}, \quad \forall q \in \mathcal{Q}. \tag{3}$$

This is primarily because for a multicast tree rooted at the source node, the total amount of bandwidth required from the powered nodes varies, not only depends on the number of powered nodes $|\mathcal{S}_q^p|$ who are able to help forward the stream q, but also the number of intended receivers $|\mathcal{S}_q^r|$. This stream-specific stress weight is calculated by the source node based on the knowledge of its social information and later piggybacked with the stream itself to all powered nodes as friends. Based on this observation, it is natural to use the stress weight to determine the *priority* of each stream q, i.e., streams with higher ω_q could be treated as the higher level of priority class when congestion occurs, or the bandwidth would be more favorable to be allocated to. Nevertheless, the lower value of ω_q implies that even if some of the powered nodes do not give out enough bandwidth, due to the relatively large number of existing helpers, others may help forward the stream instead. We next use this concept to propose a utility function for bandwidth allocations on each powered node $i \in \mathcal{S}$ among different streams to multiple receivers.

For a powered node i, the utility function of the running stream q is defined as:

$$U_q^i(b_q^i) = \frac{\omega_q}{2}\left[\left(T_q^i - b_q^i\right) + \sqrt{(T_q^i - b_q^i)^2 + 0.5}\right], \forall q, i, \tag{4}$$

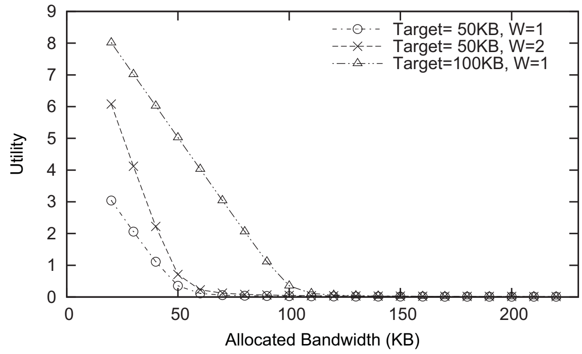

Fig. 2. The utility function for three streams with different requirements and associated stress weights.

where the input variables $\{b_q^i\}$ are a set of allocated bandwidth for each stream q on node i; and let target rate $T_q^i = k_q^i T_q$ where we recall k_q^i denotes the number of down-stream friends of stream q forwarded by node i. Fig. 2 demonstrates the profile of this utility function with respect to streams with different target rate T_q^i and the stress weight ω_q. We observe that no loss has been found in the utility value (shown as value 0) when the allocated bandwidth $b_q^i \geq T_q^i$; nevertheless, the utility grows linearly as b_q^i drops below the target T_q^i. Furthermore, for the streams with the same bandwidth requirements, the incurred utility values rise more quickly for the one with higher stress weight. The other unique characteristic of Equation 4 is the support of the preferential fairness based on defined priorities, i.e., for the same achieved utility levels, the one with highest stress weight will suffer the least rate reduction, but no streams will be starved.

Finally, we are able to define a *node*-based utility function aggregating all streams currently being serviced, as:

$$U^i = \sum_{q \in \mathcal{Q}_i} U_q^i, \quad \forall i \in \mathcal{S}, \tag{5}$$

where let $\mathcal{Q}_i$ be the set of streams currently being served at powered node i.

B. Iterative Management Approach

We introduce the following runtime optimization problem for each powered node $i \in \mathcal{S}$, where the objective is to minimize the sum of the overall utility values for the sum of all running streams:

$$\{b_q^{i,*}\} = \arg\min_{\{b_q^i\}} U^i = \arg\min_{\{b_q^i\}} \sum_{q \in \mathcal{Q}_i} U_q^i\left(b_q^i\right)$$
$$\text{subject to: } \sum_{q \in \mathcal{Q}_i} b_q^i \leq \mu_i, \tag{6}$$

where the inputs are the computed utilities for each stream q running on the node i, while the outputs are the set of allocated bandwidth achieving the optimum. The constraint is that the sum of all allocated bandwidth could not be greater than the allowed upper-bound bandwidth μ_i, which is usually specified by the user (like the amount of the bandwidth allowed for sharing purposes at the laptop). The solution of Equation 6

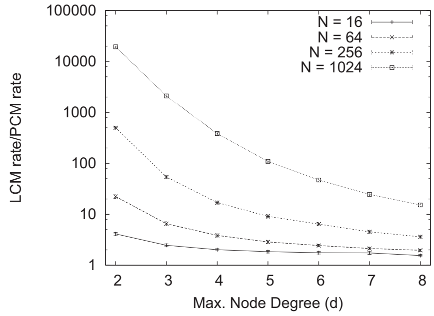

Fig. 3. Achieved source rate ratio between the proposed LCM and existing TFMCC schemes.

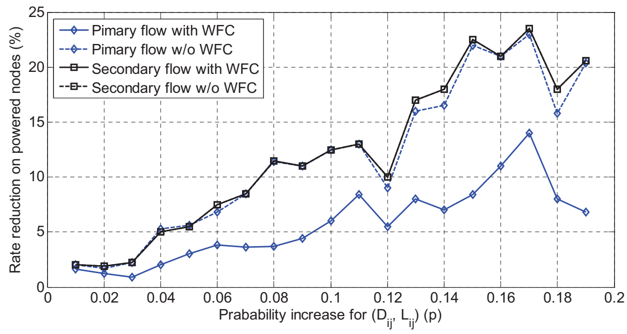

Fig. 4. Percentage of powered node suffer rate reduction under the "topo-isp" topology.

could be obtained through using Lagrangian multiplier by relaxing the constraint, and omitted here due to space limit.

Under the severe conditions, if $b_q^{i,*} < T_q^i, \forall q \in \mathcal{Q}_i$, then the dynamic topology management functionality is called and a new upstream node will be identified while the current one will be disconnected. We use the existing solutions like TMesh [7] to achieve such functionality, and integrate it into our WFC scheme. Therefore, for the stream with lower ω_q, the downstream peer is more likely to be disconnected, to reduce the consumption of the bottleneck uplink bandwidth. In extreme cases where the allocated bandwidth of all streams are cut proportionally but far below a target threshold θ_q, i.e., $b_q^{i,*} < \theta_q T_q^i$, a powered node would drop the stream with the lowest stress weight among all outgoing ones. We notice that the propose approach is completely distributed, not requiring any synchronization or information exchange among nodes; but the proposed utility function and dynamic topology management help achieve the optimality in the long run.

VI. Performance Evaluation

We assess the proposed LCM and WFC mechanisms on two real Internet topologies. One is a router-level topology, referred as "topo-isp", sampled from one of the major ISPs in North America, where the topology consists of 1426 nodes and 3587 links. Links in this topology are tagged with their latency and loss information. However, we are not attempting to use all 1426 nodes, but randomly select a subset of them as powered nodes in our scenario, on which a *complete* multicast tree is constructed with the maximum node degree d. To estimate the $D_{ij}, L_{ij}, \forall i, j \in \mathcal{S}$, on a virtual link, we use the Dijkstra's shortest path algorithm to calculate the shortest path between two powered nodes of the virtual link; then, the latency and loss rate of the calculated shortest path are used as the one for the virtual link.

The other simulation setting used is the "planetlab" [16] topology, where it consists of ten hosts, of which eight nodes are in North America, one in Western Europe, and one in East Asia. The planetlab topology also includes all virtual links between these ten planetlab hosts, but not the latency and packet loss rate information. To obtain the information for evaluations, we implement TMesh [7], an end-host multicast protocol, and deploy it on the planetlab nodes, where all nodes participate in the multicast group and form a TMesh overlay. We then feed data to one of the node to simulate media streaming. Each host collects the latency and packet loss rate information to the upstream node, and use these data to evaluate our proposed solution.

We first show that LCM offers significant improvement over PCM in terms of the maximum congestion rate, or T_{ij} in (1), and we implement TFMCC [17], as one of the protocols within the category of PCM for comparisons. Fig. 3 shows the ratio between the congestion rate of the LCM scheme and that of the TFMCC scheme. In the experiments, we use fixed node degree for each node, and vary the size of the topology from 16 to 1024. The results are averaged over 100 runs on the "topo-isp" topology. We observe that the proposed scheme can significantly improve the throughput for live sharing. This is primarily due to the accurate measurement of the end-to-end throughput over multiple virtual links on the root path; however, this benefit does not help the TFMCC scheme as much. Furthermore, it is also observed that the benefit of LCM is more significant for the powered nodes with lower node degrees. With lower node degree, the delivery path grows deeper and lengthens the receivers' root paths accordingly. In the TFMCC scheme, longer root paths would result in larger latency and higher loss rate observed by a receiver; and therefore, a lower estimated congestion rate. Our proposed LCM scheme overcomes this difficulty, where a receiver estimates its congestion rate based on its upstream virtual link only, and the rate is not affected by the lengthened root path. Finally, we observe similar results on the planetlab topology. For a maximum node degree of 4, the ratio between the congestion rate of the LCM scheme and that of the TFMCC scheme is 1.57, as not significant as shown in Fig. 3, due to the limited scale of the planetlab experiments.

To demonstrate the effectiveness of the proposed WFC scheme, we simulate two concurrent streams: one primary stream with higher ω_q and one secondary stream with lower ω_q. We set up our simulator by a 256 powered nodes on the "topo-isp" topology with maximum node degree of 4. To simulate the severe network congestion, we increase the delay on each virtual link with p probability to $(1 + p_D)D_{ij}$, where we preset $p_D = 0.1$. Likewise, the packet loss rate is increased to $(1 + p_L)L_{ij}$ probabilistically, where $p_L = 0.05$. The obtained results are also averaged over 100 runs.

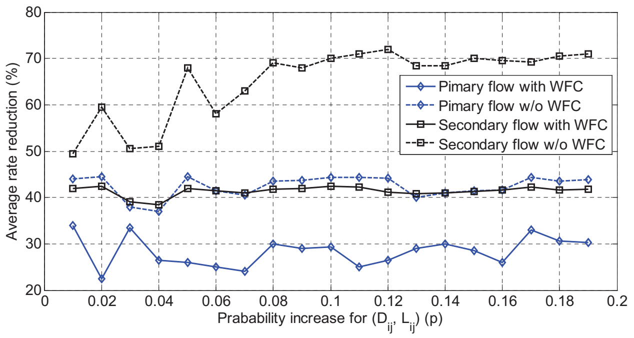

Fig. 5. Percentage of rate reduction under the “topo-isp” topology.

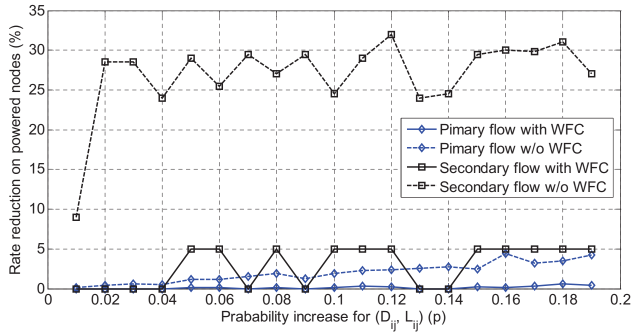

Fig. 6. Percentage of rate reduction under the planetlab topology.

Fig. 4 shows the percentage of powered nodes perform rate reduction on both the primary and secondary flows during the congestion period, with respect to different congestion parameter p. It is observed that our proposed WFC scheme successfully guarantees very low percentage of reduction compared with the one without WFC for both the primary and secondary streams. This is primarily because the stress weight accurately captures the inner structure of the Internet for stream forwarding, and the penalty-based utility functions provide a good degree of fairness among streams. Furthermore, the iterative approach with dynamic topology management allows seeking the optimal multicast tree within steps of iterations. All these benefits are not identified in the compared approach without WFC. On the other hand, with the increase of p, i.e., the higher probability for all virtual links to be congested, the increased number of powered node suffer rate reduction also increases.

Fig. 5 shows the actual percentage of rate reduction experienced by both streams on the powered nodes, compared with different congestion parameter p. Compared with the one without WFC scheme, the proposed scheme achieves about half of the reduced rate. It is interesting to observe that the proposed scheme is not affected by the congestion condition or the p parameter. This is achieved by our LCM congestion control, where the congestion rate is determined by the bottleneck congestion rate of all involved virtual links. Higher p value introduces more congested links but not necessarily the lower bottleneck one, since the congestion rate of individual virtual links are independent. We observe the similar trends on the planetlab topology, as shown in Fig. 6.

VII. Conclusions and Future Work

In this paper, we propose an efficient congestion management scheme for streams with different priorities under the scenario of P2P live sharing over mobile social networks. With increased packet loss and delay of virtual links, as well as the stringent resource provisioning on the intermediate nodes, existing multicast congestion control protocols become inefficient. We tackle the overall problem by two subproblems; one is the link-based congestion control algorithm to decide the source rate of the stream; the other problem is the per node-based flow management for prioritized streams. We proposed a novel concept of the stress weight of each stream for differential treatments, and an effective utility structure for optimal bandwidth allocation. Extensive simulation results on two real topologies show that we guarantee 50% less rate reduction under the servers congestions, and the achievable source rate of the proposed LCM scheme is over one order of magnitude higher than the existing approach. In the future, we are interested in providing real implementations on Android-based mobile phones for a testbed.

References

[1] N. D. Lane, E. Miluzzo, H. Lu, D. Peebles, T. Choudhury, and A. T. Campbell, “A survey of mobile phone sensing,” *IEEE Comm. Mag.*, vol. 48, no. 9, Sept. 2010.
[2] “Masternewmedia,” http://www.masternewmedia.org.
[3] “Webcamnow,” http://www.webcamnow.com.
[4] S. Ioannidis, A. Chaintreau, and L. Massoulie, “Optimal and scalable distribution of content updates over a mobile social network,” in *IEEE INFOCOM’09,*, april 2009, pp. 1422 –1430.
[5] B. Qureshi, M. Ilyas, G. Min, and D. Kouvatsos, “An adaptive content sharing protocol for p2p mobile social networks,” in *ACM MobiQuitous’09*, july 2009, pp. 1 –2.
[6] S. Banerjee, C. Kommareddy, K. Kar, B. Bhattacharjee, and S. Khuller, “Construction of an efficient overlay multicast infrastructure for real-time applications,” *IEEE INFOCOM 03*, 2003.
[7] W. Wang, D. Helder, S. Jamin, and L. Zhang, “Overlay optimizations for end-host multicast,” *Networked Group Communications*, 2002.
[8] M. Yang and Y. Yang, “Optimal overlay construction on heterogeneous live peer-to-peer streaming systems,” in *ICPP’10*, september 2010, pp. 580 –588.
[9] L. Rizzo, “pgmcc: a TCP-friendly single-rate multicast,” *ACM SIGCOMM ’00*, pp. 17–28, 2000.
[10] L. Vicisano, L. Rizzo, and J. Crowcroft, “TCP-like congestion control for layered multicast data transfer,” in *IEEE INFOCOM ’98*, 1998, pp. 996–1003.
[11] A. T. Nguyen, B. Li, and F. Eliassen, “Chameleon: Adaptive peer-to-peer streaming with network coding,” in *IEEE INFOCOM’10*, March 2010, pp. 1 –9.
[12] V. Padmanabhan, H. Wang, and P. Chou, “Supporting heterogeneity and congestion control in peer-to-peer multicast streaming,” *Peer-to-Peer Systems III*, pp. 54–63, 2005.
[13] D. Smilkov, H. Zhao, P. Dettori, J. Nogima, F. Schaffa, P. Westerink, and C. Wu, “Non-intrusive Adaptive Multi-Media Routing in peer-to-peer Multi-Party Video Conferencing,” *IEEE International Symposium on Multimedia (ISM 2010)*, Dec, 2010 2010.
[14] C. Luo, W. Wang, J. Tang, J. Sun, and J. Li, “A multiparty videoconferencing system over an application-level multicast protocol,” *Multimedia, IEEE Transactions on*, vol. 9, no. 8, pp. 1621–1632, 2007.
[15] S. Floyd, M. Handley, J. Padhye, and J. Widmer, “Equation-based congestion control for unicast applications,” *ACM SIGCOMM ’00*, pp. 43–56, Aug. 2000.
[16] “Planetlab,” url: http://www.planet-lab.org/.
[17] J. Widmer and M. Handley, “Extending Equation-Based congestion control to multicast applications,” *ACM SIGCOMM ’01*, 2001.