This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE ICC 2010 proceedings

Robust Scheduling of Video Streams in Network-Aware P2P Applications

Luca Abeni, Csaba Kiraly, Renato Lo Cigno
DISI - University of Trento, Povo, 38100 Trento, Italy

***Abstract*—P2P TV and video streaming are among the most bandwidth-hungry applications running over the Internet. One of the main reasons is that the scheduling of information transfer between peers is extremely aggressive and does not take network characteristics into account. Moreover, schedulers are not designed to be robust and configurable, so that their performance is greatly affected by networking conditions. This work first analyzes the impact of network heterogeneity on the streaming performance and then proposes a novel, robust, configurable, network-aware scheduler that outperforms the other schedulers in all networking scenarios.**

I. Background and Related Work

Multimedia streaming is one of the dominant component of Internet traffic. Focusing on popular applications like TV broadcasting, P2P overlay distribution systems seem to be the platform of election for new development. Commercial applications like TVAnts (http://www.tvants.com), Uusee (http://live.coolstreaming.us), PPLive (http://pplive.com), SopCast (http://www.sopcast.com) abounds and are stressing the network with their demands.

Most of these applications are based on downloading systems inspired to BitTorrent swarming: a client actively looks for the information it needs within a large neighborhood of other peers, and tries to download it as fast as its download capacity allows. This approach appears to be extremely successful and robust, however, it also results in extreme aggressiveness towards the network and a lot of resource wasting in case of congestion [1], [2].

As P2PTV systems[1] become more and more popular, researchers started a quest to find more efficient ways of distributing the information, trying to include in the scheduling algorithms the efficiency of network resource usage.

This work considers systems that are based on general mesh overlays, where peers are not organized according to a specific structure (the term "unstructured" is used in this paper). The stream is distributed based on fixed duration pieces of information (e.g., 1 s, or any number of frames) called chunks. Research in this area studied the streaming performance by considering asymptotic properties of distributed gossipping algorithms [3], [4], or by considering some kind of optimality properties for the scheduling algorithms. For example, if the fraction of the upload bandwidth of each peer dedicated to the distribution process is equal to the stream bitrate, then it is well known that the minimum time needed by a chunk to reach all the peers is $(\lceil \log_2(N) \rceil + 1)T$ where T is the chunk time. Hence, an algorithm can be said optimal if it achieves this bound; [5], [6] proved the existence of *centralised* schedulers that achieves it, while [7] proved the existence of an entire class of deadline-based *distributed* schedulers that achieve the bound above.

Other works tried to integrate network awareness in the streaming mechanism: for example, by explicitly consider bandwidth information in the peer scheduling algorithm [5], [9]. These works show that, when the network is bandwidth-heterogeneous it is impossible to obtain good performance without taking bandwidth into account.

The contribution of this paper is twofold. On the one hand, it analyzes the main performance metric of streaming systems, i.e., distribution delay, showing that based on the "type" of delay considered (average, maximum, some given percentile, ...) the relative performance of algorithms can be completely subverted. On the other hand, blending the optimal schedulers in [7] with network-aware information, it shows the possibility to obtain schedulers that are robust both to heterogeneity and to different networking conditions and perform consistently better than the others regardless of the performance metrics taken into account[2].

II. Of Schedulers, Performance and Distributions

An unstructured overlay is a set $\mathcal{S} = \{P_1, \dots P_N\}$ of N *peers*. In chunk based streaming, a *source* divides the media stream in M_c *chunks* (e.g., at fixed time intervals), and each peer P_i receives chunks C_j from other peers, and sends them out at a rate $s(P_i)$. The set of chunks already received by P_i at time t is $\mathcal{C}(P_i, t)$, and we say that P_i *owns* $\mathcal{C}(P_i, t)$.

The source generates the stream at a fixed rate of λ chunks/s, i.e., the source emits one copy of every chunk every $T_s = 1/\lambda$. All the rates are normalised w.r.t. λ, so that the generation time of C_j is $r_j = j$. Though not dimensionally correct we will sometimes call these rates *bandwidths* for the sake of simplicity.

Let $\mathcal{D}_j(t)$ be the set of nodes owning chunk C_j at time t, the diffusion delay f_j of chunk C_j (i.e., the time required for C_j

This work is supported by the European Commission through the NAPA-WINE Project (Network-Aware P2P-TV Application over Wise Network – www.napa-wine.eu), ICT Call 1 FP7-ICT-2007-1, 1.5 Networked Media, grant No. 214412

[1]We use the term P2PTV in this paper for all systems that require live streaming to a large number of peers, so that the delay in delivering the information to peers is the dominant parameter to evaluate the performance.

[2]More results on this topic are included in the Technical Report [10] and in [11].

978-1-4244-6404-3/10/$26.00 ©2010 IEEE

to reach every peer) is $f_j = \min\{\delta : \mathcal{D}_j(\delta) = \mathcal{S}\}$. According to this definition, a generic peer P_i will receive chunk C_j at time $f_j(i)$ such that $j + 1 \leq f_j(i) \leq j + f_j$.

The stochastic distribution of both $f_j(i)$ and f_j depends on the *scheduling logic* of the peers, which is composed by two *schedulers*: the *chunk scheduler*, and the *peer scheduler*. We consider only distributed schedulers where the buffermaps (i.e., the sets $\mathcal{C}(P_i, t)$) are known, and the scheduling decision is taken by the transmitter: chunks are *pushed* downstream.

The peer scheduling algorithm selects a target peer $P_k \in \mathcal{N}_i$, where $\mathcal{N}_i$ is the set of its neighbours (the neighbourhood). Given an `X` chunk scheduler and a `Y` peer scheduler, applying first `X` and then `Y` one obtains a *chunk-first* `X/Y` scheduler; the other way around one obtains a *peer-first* `Y/X` scheduler.

A. Schedulers Definition

In the following, we briefly recall the most relevant scheduling algorithms.

1) Chunk Schedulers:

Random Useful (RUc): P_i randomly selects a chunk $C_j \in \mathcal{C}(P_i, t)$ needed by some of the peers in $\mathcal{N}_i$.

Latest Useful (LUc): P_i selects the most recent chunk $C_j \in \mathcal{C}(P_i, t)$ (that is, the one having the largest generation time) that is needed by some of the peers in $\mathcal{N}_i$.

Deadline-based scheduler (DLc): P_i selects the chunk $C_j \in \mathcal{C}(P_i, t)$ having the earliest scheduling deadline and needed by some of the peers in $\mathcal{N}_i$. Scheduling deadlines are assigned to the various chunks instances present in the system, and each scheduling deadline is postponed by a fixed amount when sending a chunk. The amount of deadline postponing (δ) is a parameter of the algorithm. This algorithm [7] combined with ELp (see below) has been proved to be optimal in full meshes (like LUc), and to generally provide good performance when the neighborhood size is reduced (thus claimed to be robust, unlike LUc, which fails if the topology is not a full mesh).

2) Peer Schedulers:

Random Useful Peer (RUp): P_i randomly selects a peer in $\mathcal{N}_i$ that needs the selected chunk (or a generic chunk $C_j \in \mathcal{C}(P_i, t)$ if the peer is selected first).

Most Deprived Peer (MDp): P_i selects the peer P_j in $\mathcal{N}_i$ which owns the smallest number of the chunks currently owned by P_i [8].

Earliest-Latest (ELp): P_i selects P_j in $\mathcal{N}_i$ such that the newest chunk in $\mathcal{C}(P_j, t)$ is the oldest among all neighbors [7].

Bandwidth Aware scheduler (BAp) P_i randomly selects a target peer $P_k \in \mathcal{N}_i$ (as in RUp); the probability of selecting P_k is proportional to its output bandwidth $s(P_k)$ [9].

B. Evaluating the Scheduling Performance

The main performance parameter in streaming is the diffusion delay $f_h(i)$ and f_h: in fact, if the diffusion delay of a chunk is larger than the playout delay D, the chunk is discarded (lost). Ideally, the best scheduling logic is the one that minimizes $f_h(i)$ for all h and i; however, this is, to the best of our knowledge, not feasible. Besides, such a stochastic minimization is even very difficult to measure with simulations or experiments given the size of the problem, so that often only the average or the maximum (over all h and i) are considered.

Moreover, the network upon which the overlay is built is neither ideal, nor homogeneous, so that a scheduling logic performing very well in one situation, for instance a homogeneous network, may fail in another, e.g., when the bandwidth distribution among peers is not uniform.

We have considered several scenarios with different characteristics of bandwidth heterogeneity (see [10] for details and a complete set of results); since they provide similar results, due to lack of space, we report here only the results corresponding to a scenario called *3-class*. The 3-class scenario is based on a distribution with three classes: low-bandwidth peers (with an upload bandwidth equal to $0.5\overline{B}$), mid-bandwidth peers (with bandwidth equal to $\overline{B}$), and high-bandwidth peers (bandwidth equal to $2\overline{B}$). The fraction of high-bandwidth peers in the system is $h/3$ (where h is an *heterogeneity factor*), the fraction of low-bandwidth peers is $2h/3$, and the fraction of mid-bandwidth peers is $1-h$; as a result, the average bandwidth is $2\overline{B}\cdot h/3 + \overline{B}\cdot(1-h) + 0.5\overline{B}\cdot 2h/3 = \overline{B}$. This scenario has been selected because it captures the most important properties of heterogeneous networks, and a similar setup has already been used in literature [4].

The results presented in this paper have been generated using the P2PTVSim simulation tool developed within the Napa-Wine EU project (www.napa-wine.eu), which allows describing bandwidths and delays between peers as well as different scheduling algorithms.

C. Missing Chunks Distributions

To exemplify the difficulty of finding a good scheduler, we consider, in Fig. 1, a 3-class scenario with moderate heterogeneity ($N = 1000$, $\overline{B} = 1$ and $h = 0.15$). The figure plots the probability that chunks miss the playout time D as a function of D itself expressed as multiple of $T_s = 1/\lambda$. The upper plot reports chunk-first scheduling logic combinations and the lower plot peer-first ones. Two observations are immediate: i) given a chunk and a peer scheduler, their combination in peer-first or chunk-first is not equivalent, and ii) how chaotic the curves look!

Some algorithms (such as LUc/MDp, LUc/ELp, and LUc/BAp) shows slightly smaller probability to miss a chunk than DLc/ELp with small playout delays, but have a non-null probability to miss a chunk even for large values of D.

Similar results, not reported here for the lack of space, show that changing the networking scenario or the heterogeneity parameters, the relative merits of the schedulers can be subverted, highlighting an intrinsic fragility of scheduling logic that are not designed specifically with robustness in mind.

The strong message coming from Fig. 1 is that to obtain a performing and robust P2P streaming system, one must consider the entire distribution of the diffusion delay, and also consider several different scenarios. The LUc based schedulers do not seem to be usable, because of the high worst-case diffusion delay they provide. On the other hand, the DLc based schedulers almost always perform as well as the corresponding

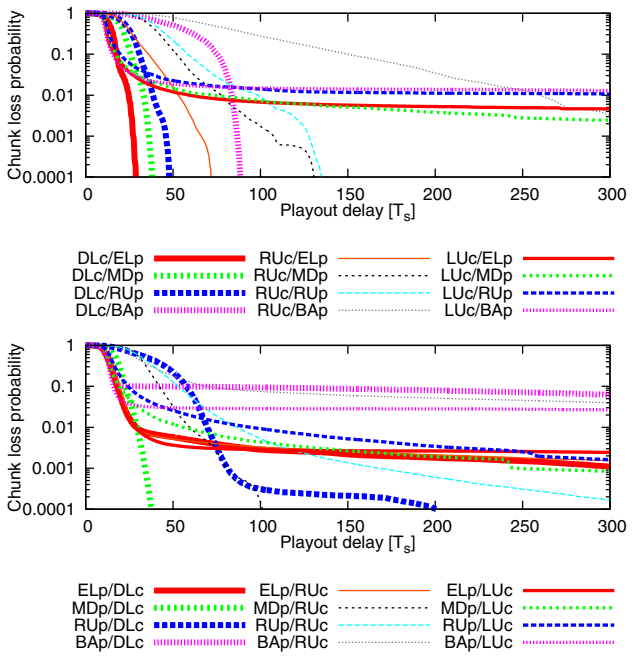

Fig. 1. Probability of missing chunks as a function of the playout delay (D) on a 3-class scenario with $N = 1000$, $M_c = 2000$, $\overline{B} = 1$ and $h = 0.15$; top plot: chunk first schedulers, bottom plot: peer first schedulers.

non-DLc schedulers (in particular, DLc/ELp is very close to be the best scheduler for all the possible metrics) and never exhibits bad behaviours. Hence, the DLc chunk scheduler will be used as a base for the robust scheduling strategy presented in this paper (DLc is also "configurable" by changing the deadline postponing parameter δ).

III. Bandwidth-Aware ELp scheduling

The analysis carried out in Sect. II highlighted the necessity for schedulers to be robust to different scenarios, and DLc/ELp proved to be the most promising combination in this direction. At the same time [5], [9] show that the peer scheduler, to be effective in a heterogeneous system, must consider the upload bandwidth of the target peers. This suggests to modify the ELp peer scheduler to select target peers based on the following multi-metric selection strategy at any given peer P_i:

$$P_j \leftarrow \max_k \{(t - L(P_k, t)) + \beta[s(P_k)/s(P_i)]\} \qquad (1)$$

where $L(P_j, t)$ is the generation time of the latest chunk at P_j at time t and β is an apportioning coefficient to blend the relative importance of the "earliest-latest" principle with the bandwidth awareness[3]. The use of the ratio $[s(P_k)/s(P_i)]$ instead of the absolute bandwidth $s(P_k)$, is simply a normalization to avoid that different stream rates (e.g., HDTV instead of standard definition) require the selection of different apportioning coefficients: (1) is normalized w.r.t. both the chunk emission time T_s an the stream bitrate.

[3]The ELp scheduler select the minimum $L(P_j, t)$ in the neighborhood, which is equivalent to maximizing $[t - L(P_j, t)]$

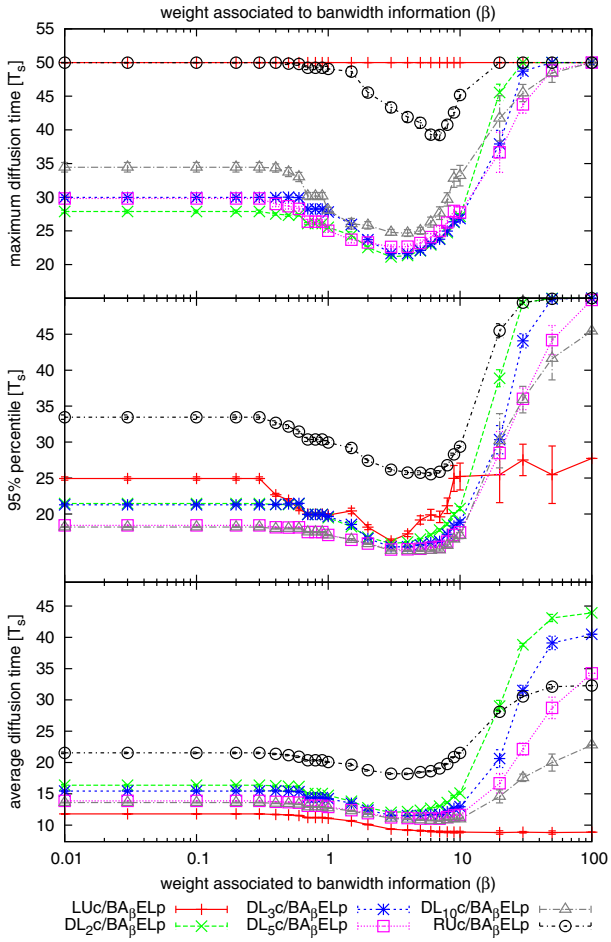

Fig. 2. BA$_\beta$ELp sensitivity to β: 3-class scenario with $h = 0.5$ and $\overline{B} = 1$, $N = 1000$, $M_c = 2000$, neighborhood size 20 and playout delay $50T_s$, different chunk schedulers

We named the scheduler (1) *Bandwidth Aware Earliest Latest* (BAELp), and its combination with the deadline based chunk scheduler DLc/BAELp. DLc/BAELp has two tuning parameters: the deadline postponing δ of DLc and the bandwidth apportioning β of BAELp. Sometimes they are explicitly indicated (possibly as numbers) with the notation DL$_\delta$c/BA$_\beta$ELp. Some preliminary properties of DLc/BAELp schedulers were presented in [11].

A. Sensitivity Analysis to β and δ

In this section, the effects of The tuning parameters of DL$_\delta$c/BA$_\beta$ELp are analysed.

Fig. 2 presents a sensitivity analysis of BA$_\beta$ELp for a scenario with medium heterogeneity ($h = 0.5$). Chunk schedulers are varied to verify if the behavior of BA$_\beta$ELp is strongly dependent on the associated chunk scheduler or not. We use four settings of DL$_\delta$c with different δ plus LUc and RUc. The number of peers is 1000, the video is 2000 chunks long, the neighborhood size is 20, and the playout delay fixed at $50T_s$. The upper plot reports the maximum delay, the middle one the 95-th percentile and the lower one the average delay. Points were obtained as an average of 100 simulation runs;

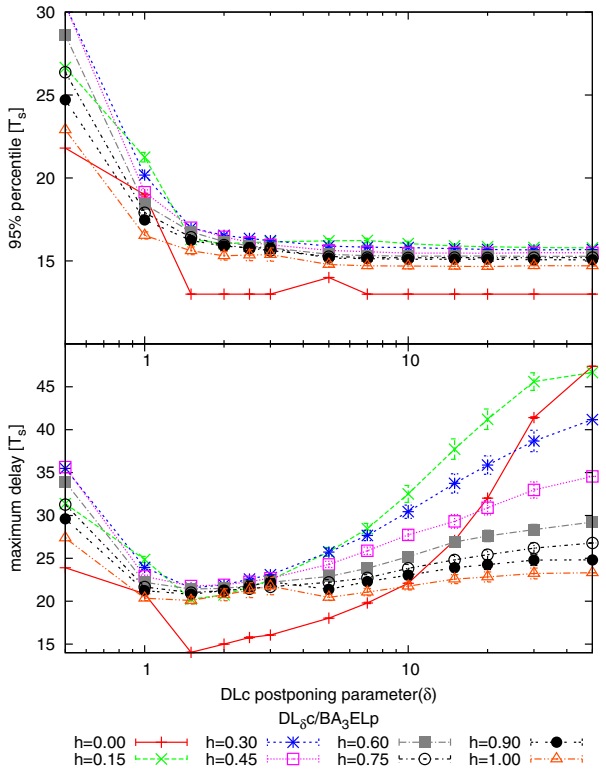

Fig. 3. DL$_\delta$c sensitivity to δ for different heterogeneity factors h: $\overline{B} = 1$, $N = 1000$, $M_c = 2000$, neighborhood size 20 and playout delay $50T_s$, BA$_3$ELp peer scheduler

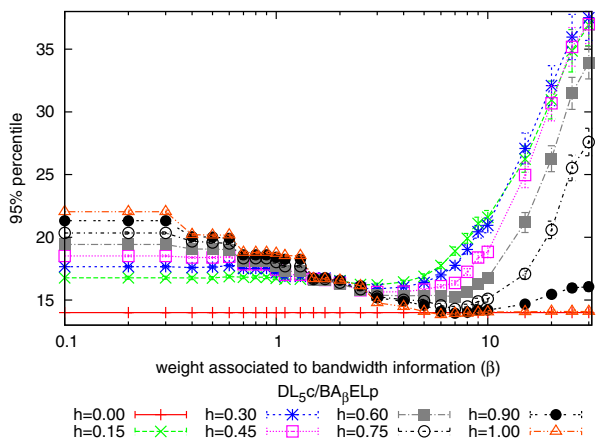

Fig. 4. BA$_\beta$ELp sensitivity to β, for different heterogeneity h; $\overline{B} = 1$, $N = 1000$, $M_c = 2000$, neighborhood size 20 and playout delay $50T_s$

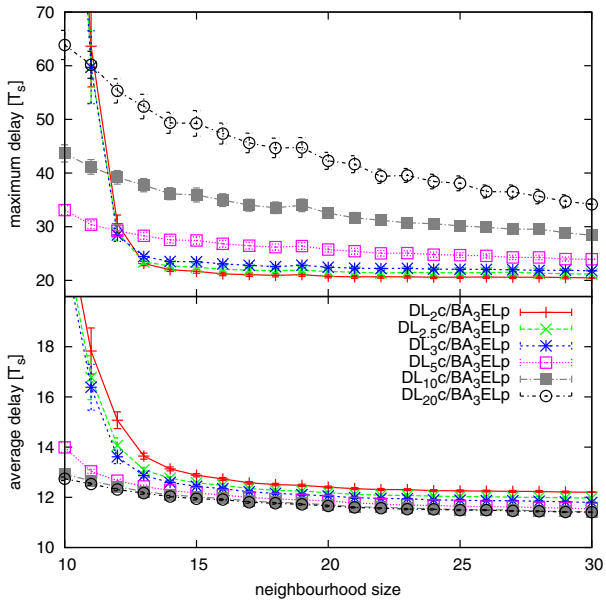

Fig. 5. Maximum and average diffusion delay. 3-class scenario with $N = 1000$, $M_c = 2000$, $\overline{B} = 1$ and $h = 0.15$

bars show 95% confidence intervals. The behavior of BA$_\beta$ELp as a function of β is practically independent from the chunk scheduler, and it shows, for this specific h, a rather flat minimum around $\beta = 2$–4. The minimum is most pronounced for the maximum delay, which is more sensitive, and minimal for the average delay. Notice once more the fragility of the LUc scheduler, which cannot guarantee lossless delivery with a deadline of $50T_s$ so that the maximum delay is constant at 50.

Different values of δ seems to have a different influence depending on the percentile observed. Moreover, small β values are a good choice for small heterogeneity scenarios (remember ELp is optimal in homogeneous networks), while very large β values force the scheduler to select peers based only on their available resources, so for scenarios with extreme heterogeneity high values of β yields better performance.

Fig. 3 reports the maximum and 95-th percentile of the delay of DL$_\delta$c/BA$_3$ELp schedulers as a function of δ for different heterogeneity factors h. Based on the analysis of β in Fig. 2 we fixed $\beta = 3$ as a reasonable and robust value. In [7] it was proved that DL$_\delta$c requires $\delta > 1$ in homogeneous networks with full connectivity. The behavior of the 95-th percentiles, and indeed also of the average and other percentiles smaller than 95 not shown here for the lack of space, is minimally sensitive to the increase of δ independently of h. Another important observation si that using large postponing values δ for networks with small heterogeneity may result in large values for the maximum delay, while using $2 < \delta < 10$ always result in good performance.

Fig. 4 reports DL$_5$c/BA$_\beta$ELp sensitivity to β of the 95-th percentile for different h. As expected the larger h the larger β should be to ensure good performance (remember that β represents the weight given to bandwidth information - see Equation 1), while a homogeneous scenario is completely insensitive to β since all peer resources are identical.

IV. Performance Evaluation

Before providing a comparison between a properly tuned DL$_\delta$c/BA$_\beta$ELp and other schedulers, we investigate the effects of the neighbourhood size. A scenario with $h = 0.15$ and 1000 peers is considered, varying the neighbourhood size from 10 to 30. DL$_\delta$c is used with different δ parameters

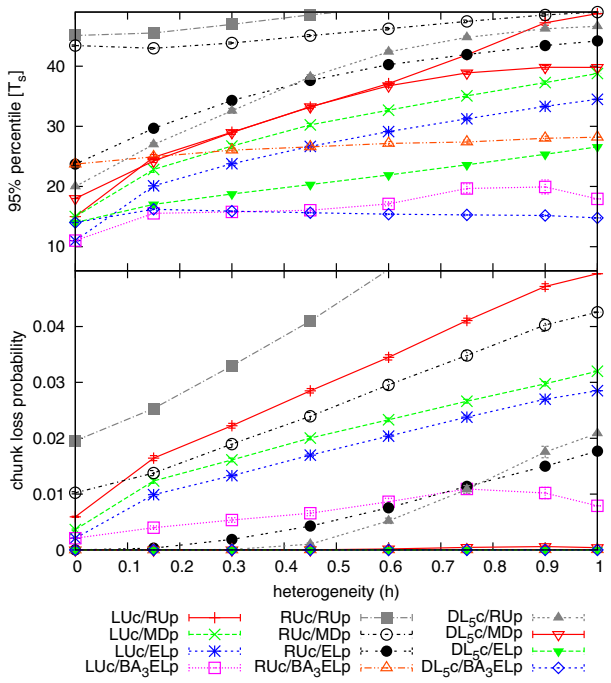

Fig. 6. Maximum and average diffusion delay. 3-class scenario with $N = 1000$, $M_c = 2000$, $\overline{B} = 1$ and $N_N = 20$

ranging from 2 to 20. Because of space constraints we only show average and maximum delay in Figure 5, but losses and delay percentiles only confirm these results. The first important observation is that the minimum neighbourhood size needed to achieve reasonable performance is larger than in the uniform case [7], which is $\lceil \log_2(N) \rceil = 10$ for 1000 peers. This result is understandable, since a randomly generated small neighbourhood has a high possibility of not containing high-bandwidth peers, defeating the bandwidth aware behaviour of BAELp. A second interesting effect is that larger values of δ help when the neighbourhood is too small: when it is near to 10, the diffusion delays for $\delta = 2$, 2.5, or 3 increase to unreasonably large values (leading to lost chunks), whereas the diffusion delays for larger values of δ are still reasonable. Hence, larger values of δ can improve the robustness of the chunk scheduler at the cost of increasing the worst-case diffusion delay when the neighborhood is large enough.

Finally, it is worth noting that increasing δ lead to better average diffusion times both for small and for large neighborhoods. The sum of this effect with the one previously noticed make the worst-case curve particularly interesting for small values of the neighborhood: for very small values, the effects of the neighbourhood are predominant, and larger δ values have to be preferred even for the worst-case statistics. When the neighbourhood is large enough, increasing δ helps in reducing the average diffusion delay (or small percentiles) and decreasing δ helps in reducing the maximum diffusion delay (or large percentiles). Again, it is worth noting how $\delta = 5$ tends to behave reasonably in all the situations and thus can be taken as average, robust setting.

The last experiment reported compares a properly tuned DL$_\delta$c/BA$_\beta$ELp with other scheduling algorithms. The results (chunk loss and 95-th percentile of the diffusion delay) are shown in Figure 6. Only four of the twelve combinations are performing well enough to avoid chunk losses all over the heterogeneity range: DL$_5$c/MDp, DL$_5$c/ELp, DL$_5$c/BA$_3$ELp and RUc/BA$_3$ELp. Of these four DL$_5$c/BA$_3$ELp provides the lowest delays. Only in the homogeneous case LUc/BA$_3$ELp and LUc/ELp have lower delays, but they have chunk losses. Note also that with any chunk scheduler, BA$_3$ELp provides the best performance for the whole heterogeneity range, both in terms of chunk losses and delays. ELp ranks second and MDp third independent of the chunk scheduler.

V. Discussion and Conclusions

This paper compared a large number of scheduling strategies for P2P streaming systems in presence of network heterogeneity (peers having different upload bandwidths). Since none of the existing peer schedulers seems to perform well in all the conditions, a new peer scheduling algorithm, named BA$_\beta$ELp has been developed and compared with the others. BAELp combined with the DL$_\delta$c chunk scheduler is robust to heterogeneity and outperforms all the other scheduling algorithms in a large number of different conditions and scenarios.

The proposed algorithm is characterised by two tunables DL$_\delta$c and BA$_\beta$ELp whose optimal value depend on the network conditions. Hence, measurement based network-awareness and self-adaptation can be added to achieve dynamically optimal performance.

References

[1] D. Ciullo, M. Mellia, M. Meo, and E. Leonardi, "Understanding P2P-TV Systems through Real Measurements," *In Proc. IEEE GLOBECOM 2008*, Nov. 30 – Dec. 4, 2008.

[2] E. Alessandria, M. Gallo, E. Leonardi, M. Mellia, and M. Meo, "P2P-TV Systems under Adverse Network Conditions: A Measurement Study," *In Proc. Infocom 2009*, April 19–25, 2009.

[3] S. Sanghavi, B. Hajek, and L. Massoulié, "Gossiping with multiple messages," *In Proc. IEEE INFOCOM 2007*, May 6–12, 2007.

[4] T. Bonald, L. Massoulié, F. Mathieu, D. Perino, and A. Twigg, "Epidemic live streaming: optimal performance trade-offs." *In Proc. ACM SIGMETRICS 2008*, June 2–6, 2008.

[5] Y. Liu, "On the minimum delay peer-to-peer video streaming: how realtime can it be?" *In Proc. ACM MULTIMEDIA 2007*, Sept. 24–29, 2007.

[6] J. Mundinger, R. Weber, and G. Weiss, "Optimal scheduling of peer-to-peer file dissemination," *J. of Scheduling*, vol. 11, no. 2, pp. 105–120, 2008.

[7] L. Abeni, C. Kiraly, and R. Lo Cigno, "On the optimal scheduling of streaming applications in unstructured meshes," *In Proc. IFIP Networking 2009*, May 11–15 2009.

[8] L. Massoulié, A. Twigg, C. Gkantsidis, and P. Rodriguez, "Randomized decentralized broadcasting algorithms," *In Proc. IEEE INFOCOM 2007*, May 6–12, 2007.

[9] A. Couto da Silva, E. Leonardi, M. Mellia, and M. Meo, "A bandwidth-aware scheduling strategy for p2p-tv systems," *In Proc. P2P 2008)*, Sept. 8–11 2008.

[10] L. Abeni, C. Kiraly, and R. Lo Cigno, "Achieving Performance and Robustness in P2P Streaming Systems," University of Trento, T.R. DISI-09-041, 2009 (http://disi.unitn.it/locigno/preprints/TR-DISI-09-041.pdf).

[11] L. Abeni, C. Kiraly, and R. Lo Cigno, "Scheduling P2P Multimedia Streams: Can We Achieve Performance and Robustness?", *In Proc. IMSAA-09*, Bangalore, India, Dec. 9-11, 2009.