Multipath Routing for Video Unicast over Bandwidth-Limited Networks

Jiancong Chen S.-H. Gary Chan
Department of Computer Science
The Hong Kong University of Science and Technology
Hong Kong

Abstract—Video applications such as video-on-demand systems in general have bandwidth and delay constraints. Such QoS requirements can no longer be guaranteed when video is transmitted with the traditional shortest-path routing over a bandwidth-limited network such as the Internet. In this paper, we propose multipath routing algorithms for video unicast so as to meet a certain bandwidth requirement with minimum start-up delay (and hence low user buffer requirement). We first formulate the problem which in its most general form is difficult to solve. However, for the special case where the network links are of unit capacity, we present an exact solution using k-shortest-disjoint paths algorithm with video rescheduling in the source. For the most general case, we propose an efficient heuristic based on max-flow and shortest path algorithms. The complexity of such algorithm is only $O(|V|^3)$, where $|V|$ is the number of nodes in the network. Extensive simulation results show that the multipath approach meets the bandwidth requirement with a delay close to the shortest-path routing.

Keywords—Quality-of-Service, QoS routing, multipath routing, disjoint paths, video rescheduling algorithm

I. Introduction

In resent years, there has been growing interest in delivering digitized audio-visual information over local or wide area networks [1], [2], [3]. Such multimedia applications often have stringent quality-of-service (QoS) requirements in, for example, bandwidth and delay. Routing has to find paths with sufficient resources to meet the user requirements. In traditional data networks, routing is primarily concerned with connectivity. The protocols usually represent the network with a single metric such as hop-count or delay, and use single path (i.e., shortest-path) algorithms for path computation. However, in order to support the QoS requirements for realtime multimedia streaming, routing protocols often need to consider multiple metrics such as bandwidth and delay. Therefore, the shortest-path algorithm is no longer sufficient. Routing with multiple QoS requirements is called QoS routing, which has been much discussed recently.

For video applications, high end-to-end bandwidth and low startup delay have to be guaranteed in order to offer high user satisfaction. For example, in an MPEG-I video-on-demand system, a streaming bandwidth of 1.5 Mbps is required for quality transmission. In a bandwidth-limited network such as the Internet, such a high end-to-end bandwidth usually cannot be guaranteed with the traditional single shortest-path routing. Multipath approach can be adopted where the originating node may deliver the date to a particular destination via multiple paths. This offers data transfer rate higher than what is possible with any one path by bandwidth aggregation. In this paper, we consider the design of a multipath routing for video unicast which finds a multipath set P such that the following two goals are achieved:

1. The minimum aggregate bandwidth for the video applications is met;
2. The startup delay is minimized over all the feasible paths satisfying 1.

Since single-path routing is a special case of multipath routing with path number equals to 1, our multipath routing algorithms reduces to shortest-path algorithm under this special case.

As an example, we show in Fig. 1 a network with six routers labelled from v_0 to v_6 interconnected by a number of links. Each link has its link state indicated by a duplet (w, d), where w is the link bandwidth (e.g., in unit of 100 kbps) and d is the propagation delay (e.g., in unit of 10 ms). Suppose that a video stream with a bandwidth requirement of 1.5 Mbps is to be transmitted from node v_0 to node v_6. Note that for single path routing, the maximum bandwidth that can be provided is only 10 units (i.e., 1 Mbps). On the other hand, when multipath routing is used (indicated by paths $(v_0v_1v_4v_5v_6)$ and $(v_0v_2v_3v_5v_6)$), the end-to-end bandwidth requirement of 15 units (or 1.5 Mbps) can be met.

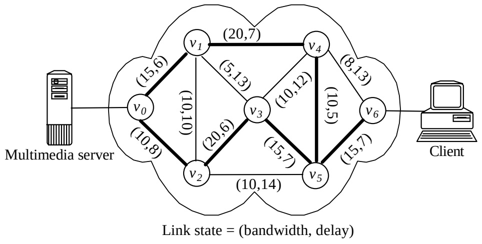


Fig. 1. Multipath routing provides high end-to-end bandwidth for multimedia applications.

Note that since the path lengths in multipath routing are different, the end host needs to reassemble the packets, therefore incurring a buffer requirement. For the same reason, the end-to-end delay is longer than the shortest-delay path. In order to reduce this delay (and hence the user buffer requirement), we propose in this paper a video rescheduling algorithm at the source based on the knowledge of the path lengths. In this algorithm, video data are segmented and multiplexed in a specific manner over different paths so that the end-host can assemble the data and play back the video at the earliest time. Using the above example with video rescheduling, the start-up delay can be reduced from 280 ms to 260 ms (the shortest-path delay is 250 ms) and user buffer is reduced from 30 kbits to 10 kbits.

The exact solution for the multipath problem proves to be very difficult. Therefore, we present in this paper a heuristic. However, under the special case of a network with unit bandwidth in each link, multipath routing reduces to *edge-disjoint paths*, in which the flows do not share links (but can share nodes) [4], [5]. In such a case, with our video rescheduling algorithm at the source, an exact and efficient solution can be obtained to minimize the user delay (of complexity $O(|V| \times \log |V|)$, where $|V|$ is the number of routers in the network). Refer to Fig. 1 but with unit link bandwidth, two shortest-disjoint paths $(v_0v_1v_4v_6)$ and $(v_0v_2v_3v_5v_6)$ with delay 260 ms and 280 ms can be found. By applying video rescheduling, the user delay can be further reduced from 280 ms to 270 ms. Note that our algorithm does not assume that the network metrics has to be stable during the video delivery session. If the metrics change, the algorithm can be rerun to meet the bandwidth requirement.

Our contributions of this paper is hence three-fold:

1. We propose an efficient heuristic for multipath routing to meet the bandwidth requirement for video unicast while offering a low end-to-end user delay;
2. We present a video rescheduling algorithm to further reduce the user delay (and hence buffer requirement); and
3. We present an exact solution for multipath routing with video rescheduling for the special case of networks with unit link bandwidth.

We briefly present the related work as follows. Wang and Crowcroft studied a similar problem "bandwidth-delay-constrained path problem" and proposed two algorithms for it [6], [7], [8]. These algorithms, however, find a single path subject to bandwidth-delay constraints. The advantages of using multipath routing are discussed in [9]. Previous work on multipath routing algorithms mainly focuses on packet delay and reliability without considering bandwidth requirement of the applications

This work is supported, in part, by the Areas of Excellence (AoE) on Information Technology funded by the University Grant Council in Hong Kong (AoE 98/99.EG01), and by the Direct Allocation Grant of Hong Kong Research Grant Council (DAG99/00.EG25).

[10], [11]. Video rescheduling is proposed to minimize the reordering overhead at the end-host in [12]. Our work focuses on finding multiple paths with sufficient bandwidth and low delay for video delivery over bandwidth-limited networks.

This paper is organized as follows. We first formulate the multipath routing problem in Sect. II. In Sect. III, we present the video rescheduling algorithm and show an exact solution of the multipath routing problem for a network with unit link bandwidth. We present in Sect. IV a heuristic based on max-flow and shortest-path algorithms for general multipath problem. We finally conclude in Sect. V.

II. Problem Formulation

In this section, we formulate the multipath routing problem. We model a network as a graph $G = (V, E)$, where V is the set of nodes in the network and E is the set of links or edges. Each link $e = (v_i, v_j) \in E$ has two associated positive metrics:

- the available bandwidth, $w(e) \in Z^+$ in some appropriate units (say, in 100kbps).
- the propagation delay, $d(e) \in Z^+$ in some appropriate units (say, in 10ms).

A simple and loop-free path p in G is a list of nodes $(v_1, \ldots, v_n)$ such that $\forall i, 1 \le i < n$, $(v_i, v_{i+1}) \in E$ and no node appears more than once. Denote $d(p)$ and $w(p)$ the delay and bandwidth of such a path p, respectively. We clearly have

$$d(p) = \sum_{i=1}^{n-1} d(v_i, v_{i+1}), \tag{1}$$

and

$$w(p) = \min_{i \in [1..n-1]} \{w(v_i, v_{i+1})\}. \tag{2}$$

The bandwidth of a path is the minimum of the residual bandwidth of all links on the path, the so-called “bottleneck bandwidth”, Usually, delay of a path has two basic components: queuing delay and propagation delay. Note that the queuing delay is determined by the bottleneck bandwidth and traffic characteristics. Since the queuing delay is mostly captured in the bandwidth metric, we only need to consider propagation delay here. Therefore, we can assume that the two metrics are independent. However, our algorithm is general enough that it is independent of the relationship between $w(p)$ and $d(p)$.

Further define multiple paths $P = \{p_1, p_2, \ldots, p_K\}$, where K is the number of paths taken. The aggregate bandwidth, W, of the K paths is therefore

$$W(P) = \sum_{i=1}^{K} w(p_i). \tag{3}$$

When the video stream flows through the K paths, only after the data from the longest one arrive can the end-host begin to playback the video. So, the startup delay is equal to $D(P)$, where

$$D(P) = \max_{p_i \in P} \{d(p_i)\}, \tag{4}$$

In general, those data received before playback have to be buffered and the corresponding buffer requirement R is given by

$$R = \sum_{p_i \in P} w(p_i)(D - d(p_i)). \tag{5}$$

The multipath routing problem can now be stated as follows:

Bandwidth-constrained Delay-optimization Multipath Problem (BDMP): *Consider a network represented by a graph $G = (V, E)$ and a bandwidth constraint B, find a multipath set P from a source node s to a destination node t such that:*

1. *$W(P) \ge B$, and*
2. *$D(P)$ is minimized over all feasible paths satisfying 1.*

In the following, we first present an exact solution to the above problem for the special case that the links are of unit capacity, and then a heuristic for the general case that the link capacities are not equal.

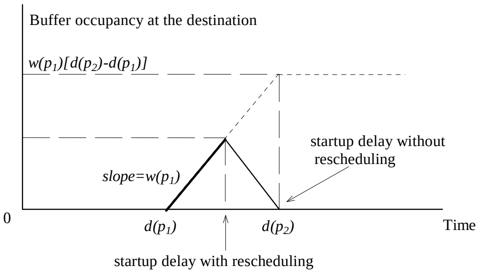


Fig. 2. Buffer occupancy versus time for multipath transmission with and without video rescheduling.

III. Disjoint Multipath Routing with video Rescheduling

In this section, we first propose a video rescheduling algorithm. By using the algorithm under the special case that each link in the network is of unit capacity, an exact solution for the multipath routing problem can be obtained. Clearly, if the video application requires more than one unit of bandwidth, multiple paths have to be found in order to offer sufficient bandwidth. For example, if the bandwidth requirement is k units, k paths is needed to deliver the video. Since the links are of unit bandwidth, the paths found are disjoint. Therefore, the problem is reduced to finding a set of edge-disjoint paths with minimum startup delay. In the following, we will show that combining with our video rescheduling algorithm, the *k-shortest-disjoint paths (k-SDP)* is the optimal solution for the BDMP problem (k-shortest-disjoint paths are k disjoint paths with the shortest total path length). The algorithm of finding the k-shortest-disjoint paths is well known in literature. Interested readers may refer to [11] for more detail.

A. Video Rescheduling Algorithm

In multipath routing, the start-up delay of the user is equal to the longest path if video transmission is not rescheduled. Furthermore, the end-host has to buffer the data due to the path difference before it can playback the video. As a result, if the difference between the path length is large, the buffer requirement is correspondingly large.

For example, consider a simple case with two paths p_1 and p_2 with path length $d(p_1) < d(p_2)$ and bandwidth $w(p_1)$, $w(p_2)$, respectively ($w(p_i)$ does not need to be equal). Clearly, the start-up delay is $d(p_2)$ and the buffer required in this case is given by

$$w(p_1)(d(p_2) - d(p_1)). \tag{6}$$

As $(d(p_2) - d(p_1))$ may be long (or $w(p_1)$ may be large), the buffer requirement is high.

We propose a video rescheduling algorithm at the source given the knowledge of the path lengths to decrease the start-up delay (and hence the buffer requirement). The video rescheduling algorithm is illustrated in Fig. 2, in which we plot the buffer occupancy at the destination with respect to time. Video is transmitted at the source at time 0. The dashed line corresponds to the case, in which there is no rescheduling. It rises with slope $w(p_1)$ at time $d(p_1)$ until time $d(p_2)$. Clearly, the user start-up delay and buffer requirement are as stated above.

The main reason for the long delay is due to the fact that video packets are multiplexed into the two paths in a round-robin manner. If the beginning part of the video can be sent via the shorter path, clearly the end host can playback earlier, and its buffer is emptied at time $d(p_2)$, when the necessary video data arrive from the longer path. The delay (and buffer) can be reduced according to the solid line by means of rescheduling in Fig. 2. From the figure the video can be started at time

$$D' = d(p_2) - \frac{w(p_1)}{B}\left(d(p_2) - d(p_1)\right). \tag{7}$$

Clearly, the buffer requirement is reduced to

$$\begin{aligned} R' &= w(p_1)(D' - d(p_1)) \\ &= \frac{1}{B} w(p_1) w(p_2)(d(p_2) - d(p_1)). \end{aligned} \tag{8}$$

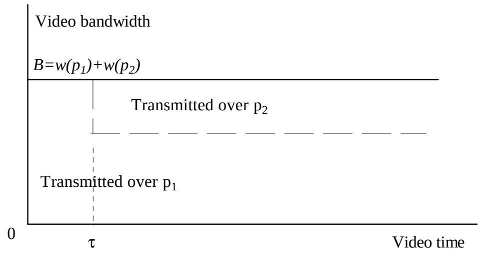


Fig. 3. Video transmission schedule with rescheduling for $K = 2$.

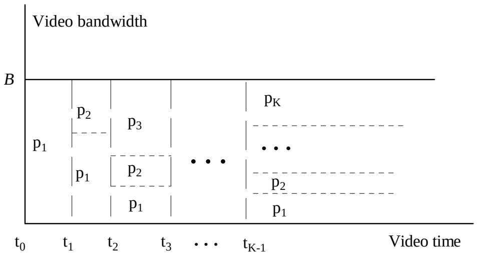


Fig. 4. Video transmission schedule with rescheduling for general K.

The rescheduling scheme is illustratived in Fig. 3, in which we show which part of the video should be sent through the two paths with total bandwidth B. First, the video is divided into two segments at time τ, where τ can be obtained as

$$\tau = \frac{w(p_1)(d(p_2) - d(p_1))}{B}. \tag{9}$$

From time τ onwards, the video is divided into 2 substreams ss_1 along path p_1 and ss_2 along path p_2 and multiplexed in a finegrained manner, respectively. We stream video segment $(0,\tau)$ into path p_1, and substream ss_2 in parallel. After finishing segment $(0,\tau)$ at time $d(p_2) - d(p_1)$, p_1 is used to transmit ss_1. In this way, the substreams would arrive at the destination at the same time such that they can be combined and played back immediately without buffering.

In general, given K paths $P = \{p_1, p_2, ..., p_K\}$ with $d(p_1) \leq d(p_2) \leq ... \leq d(p_K)$, the start-up delay without video rescheduling is $d(p_K)$. While the buffer requirement is given by

$$\begin{aligned} \hat{R} &= \sum_{i=1}^{K-1} w(p_i)(d(p_K) - d(p_i)) \\ &= B \times d(p_K) - \sum_{i=1}^{K} w(p_i)d(p_i). \end{aligned} \tag{10}$$

With video rescheduling, the video can be played back earlier by $\hat{R}/B$ seconds, i.e.,

$$\begin{aligned} \hat{D} &= d(p_K) - \frac{\hat{R}}{B} \\ &= \frac{1}{B}\sum_{i=1}^{K} w(p_i)d(p_i), \end{aligned} \tag{11}$$

after using Eq. 10.

To achieve the optimal minimum delay $\hat{D}$, we use multiplexing to transmit each video segment along the paths in parallel such that from time $\hat{D}$ onwards, all the video data at the receiver can be played back continuously.

Let t_i $(0 \leq i \leq K-1)$ be the division point of the video segments.

- *Step 1:* Segment the video at the beginning as

$$t_0 = 0; \tag{12}$$

and

$$t_i = \frac{\sum_{l=1}^{i} w(p_l)(d(p_{i+1}) - d(p_l))}{B}, \tag{13}$$

for $i = 1, 2, \ldots, K-1$.

- *Step 2:* Multiplex video segment (t_{i-1}, t_i) into i paths $p_1, p_2, \ldots, p_i$ in the ratio of $w(p_1) : w(p_2) : \ldots : w(p_i)$ as shown in Fig. 4, in which segment (t_0, t_1) is transmitted along path p_1, and segment (t_1, t_2) is transmitted along path p_1 and p_2 in the ratio of $w(p_1) : w(p_2)$, and so on.

In the following we prove that using this rescheduling algorithm, data in the same segment transmitted along different paths arrives at the end-host at the same time, so that the video can be continuously played back. Since the segment (t_{i-1}, t_i) is transmitted along i paths, the transmission time τ_i for it is given by $(i = 1, 2, ..., K-1)$

$$\begin{aligned} \tau_i &= \frac{(t_i - t_{i-1}) \times B}{\sum_{l=1}^{i} w(p_i)} \\ &= d(p_i) - d(p_{i-1}). \end{aligned} \tag{14}$$

This segment arrives at the end-host along path p_m $(1 \leq m \leq i)$ at time

$$\begin{aligned} \sum_{l=m}^{i-1} \tau_l + d(p_m) &= (d(p_i) - d(p_m)) + d(p_m) \\ &= d(p_i). \end{aligned} \tag{15}$$

Therefore, using this rescheduling algorithm, the arrival time for data segment (t_{i-1}, t_i) is $d(p_i)$ independent of the path it is transmitted. In other words, the video can be played back continuously.

B. An Exact Solution: Disjoint-Path Routing with Video Rescheduling

Note that the start-up delay after applying the video rescheduling algorithm is given by $\sum_{i=1}^{K} \frac{w(p_i)}{B} d(p_i)$. Under the special case that the links are of unit capacity, the start-up delay is proportional to $\sum_{i=1}^{K} d(p_i)$. In the following, we prove that the *k-shortest-disjoint paths* is an optimal solution for the BDMP problem in such a network.

Theorem 1: The startup delay of the k-shortest-disjoint paths with video rescheduling is minimized.

Proof: Given a network G and k-shortest-disjoint paths $P = \{p_1, p_2, \ldots, p_k\}$, for all k-disjoint paths $P' = \{p'_1, p'_2, \ldots, p'_k\}$ in G, we have

$$\sum_{i=1}^{k} d(p_i) \leq \sum_{i=1}^{k} d'(p'_i). \tag{16}$$

The startup delay of the k-shortest-disjoint paths with video rescheduling is $\sum_{i=1}^{k} d(p_i)/k$. From Eq. (16), we have,

$$\frac{\sum_{i=1}^{k} d(p_i)}{k} \leq \frac{\sum_{i=1}^{k} d(p'_i)}{k} \leq D(P'), \tag{17}$$

i.e., the startup delay of the k-shortest-disjoint paths with video rescheduling is never larger than that of the other path sets with or without video rescheduling. This completes the proof.

C. Illustrative Simulation Results for Disjoint Path with Video Rescheduling

In this section, we study the performance of the multipath approach with video rescheduling with respect to: i) the video quality obtained, as given by the end-to-end transmission bandwidth; ii) the startup delay D; and iii) the success rate of finding the multiple paths needed.

To evaluate the performance of the multipath routing over the network, we generate a hierarchical network similar to the Internet based on the method used in [13]. We generate networks of different size ($N \times N$ grids) with different density, which is defined as the ratio of

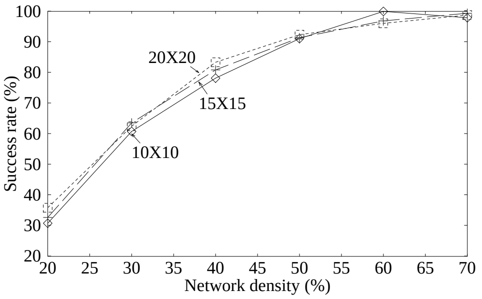


Fig. 5. Success rate of finding SDP versus network density given a network size.

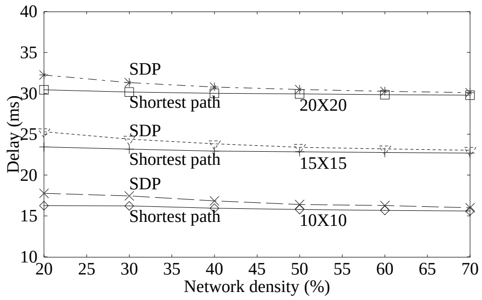


Fig. 6. Comparison of end-to-end delay versus network density between SDP and shortest-path given a network size.

the number of routers with respect to the network size. The network is generated as follows. First, a center router is placed in the center of the network. Then. the network plane is divided into 4 equal-size areas, the center of which is placed a area router as the second-layer router. The area routers are fully connected to the center router to form a backbone network. The probability that a second-layer router is connected to another one is:

$$p_l = \lambda \exp(-l(u,v)/\rho L), \tag{18}$$

where $l(u,v)$ is the distance between the router position u and v, while L is the maximum distance between any two nodes that may be connected by a link (i.e., $L = \frac{N}{2}\sqrt{2}$ in a network of size $N \times N$), λ is the link density and ρ controls the proportion of short links. We set $\lambda = 10.0$, and $\rho = 0.1$. After this, layer-3 routers and links are generated. Every such router is inserted randomly over the available grid positions. Link are added to connect them to the nearest area router. The layer-3 routers are connected according to the p_l given above with $\lambda = 0.7$, and $\rho = 0.2$. We assume the maximum capacity of the data links between the area routers is 100 Mbps and others 10 Mbps. In order to simulate the background traffic, at any time, the available bandwidth is modelled as uniformly distributed from zero to the maximum capacity.

Using the above network model, we study the performance of the k-shortest-disjoint paths with video rescheduling by considering that the links are of unit bandwidth, and 3 units of bandwidth are required, i.e., 3-shortest-disjoint paths are needed. Figure 5 plots the success rate of finding the shortest-disjoint paths versus network density given a certain network size. Clearly, as the network density increases, the success rate is getting higher, as it is more likely to find multiple paths to satisfy the bandwidth requirement. The success rate does not change much with respect to different network size. Therefore, network density is the main parameter that affects the success rate. We show in Fig. 6 the startup delay versus network density. The startup delay does not change much with respect to the network density, meaning that network density has little effect on startup delay in our model. When the network increases, the startup delay increases accordingly. However, the startup delay of the multipath approach and the shortest-path does not differ much. This result shows that the SDP algorithm with video rescheduling only trades off a little higher delay to offer a large bandwidth.

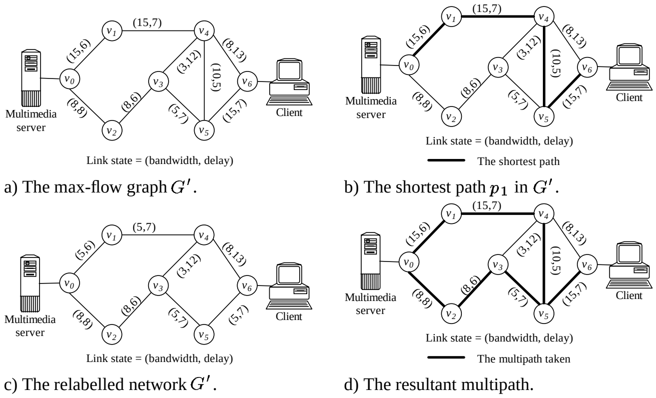
IV. Multipath Heuristic and Simulation Results

In this section, we present a multipath heuristic for BDMP with arbitrary bandwidth and delay. We first find out the maximum aggregate bandwidth from source s to destination t through the preflow-push algorithm. If such bandwidth is less than the video required bandwidth, then the video cannot be streamed over the network with its full quality. Otherwise, we can use the following multipath heuristic to find the path set P, which offers enough total end-to-end bandwidth for the video application with low delay.

A. Multipath Heuristic

In the following, we present the multipath heuristic followed by an illustrative example.

- *Step 1:* Let the multipath set P be empty initially, i.e., $P = \emptyset$



a) The max-flow graph G'.

b) The shortest path p_1 in G'.

c) The relabelled network G'.

d) The resultant multipath.

Fig. 7. A multipath computation example.

- *Step 2:* Run the preflow-push algorithm on network G to compute the max-flow M. We assume in the following that $M \geq B$. Add up all the augmenting paths to form the max-flow graph $G' = (V', E')$.
- *Step 3:* Take the shortest path p in G' and add p to P:

$$P \leftarrow P + \{p\}. \tag{19}$$

- *Step 4:* Subtract the bandwidth of the shortest path p from the available bandwidth of each link along p in G', i.e.,

$$w(e') \leftarrow w(e') - w(p), \quad \forall e' \in p. \tag{20}$$

- *Step 5:* Repeat *Step 3 & 4* until total bandwidth offered by P is sufficient for the video application, i.e.,

$$\sum_{p_i \in P} w(p_i) \geq B. \tag{21}$$

We now analyze the complexity of the heuristic. First of all, observe that the preflow-push algorithm in *Step 2* is of complexity $O(|V|^3)$ [14]. Given G', the shortest path algorithm in *Step 3* is of complexity $O(|V|\log|V|)$. Since it iterates $|P|$ times, where $|P|$ is the number of paths, the complexity of *Step 5* is given by $O(|V|\log|V| \times |P|)$. As a result, the complexity of our heuristic is given by $O(|V|^3 + |V|\log|V| \times |P|) = O(|V|^3)$.

As a simple example, we consider the network in Fig. 1 with a bandwidth requirement B of 15 units (i.e., $B = 1.5$ Mbps):

- *Step 1:* The multipath set P is empty.
- *Step 2:* The max-flow graph G' of the network in Fig. 1 is shown in Fig. 7 a), in which the max-flow $M = 23$ units.
- *Step 4:* Subtract the bandwidth of the shortest path from the available bandwidth of each link along path p_1 in G'. The residual graph will then be as shown in Fig. 7 c).
- *Step 5:* From Fig. 7 b), the bandwidth offered by p_1 is 10 units which does not meet the bandwidth requirement B. Hence we

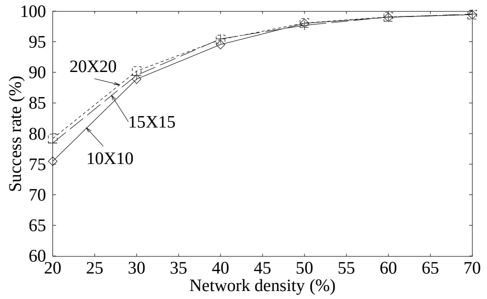


Fig. 8. Success rate versus network density for the multipath heuristic given a network size.

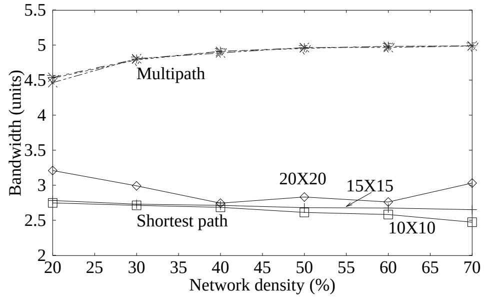


Fig. 9. Comparison of end-to-end bandwidth versus network density between the multipath heuristic and shortest-path.

repeat *Steps 3 & 4* to yield the shortest path $(v_0v_2v_3v_5v_6)$ in the relabelled G', which is shown in Fig. 7 c). Since the total bandwidth is equal to B, the algorithm terminates with the resultant multipath shown in Fig. 7 d). The total delay in this case is the longest path in set P, i.e., 28 units (or 280 ms). By applying the video rescheduling scheme, the startup delay can be reduced to 26 units (or 260 ms).

B. Simulation Results for Multipath Heuristic

Using the network model described in Sect. III-C, we compare the performance of the multipath routing with video rescheduling with the shortest-path algorithm in terms of bandwidth, delay and success rate of finding sufficient bandwidth for the application. In this study, we randomly generate thousands of networks with different size as mentioned before, and run the two routing algorithms on them to collect data. The bandwidth requirement of the application is 5 units.

We plot Fig. 8 the success rate of finding multipath with sufficient bandwidth versus network density. The success rate increases with the network density. However, even when the network density is low, say, 30%, the success rate is quite high (around 90%). Furthermore, the success rate is independent of the network size.

The multipath heuristic on average can achieve much larger bandwidth than the shortest-path routing. We plot the average bandwidth obtained by the two routing algorithms versus network density in Fig. 9. The bandwidth obtained by the multipath heuristic is very close to the requirement and increases when the network density increases, while that of the shortest-path remains low. We can conclude that the multipath heuristic can offer much higher bandwidth. We show in Fig. 10, which plots the corresponding average startup delay of the two algorithms versus network density. Clearly, the startup delay of the multipath heuristic is close to the shortest-path. The startup delay of the multipath heuristic is very slightly higher than than the shortest-path, while offering much higher end-to-end bandwidth.

V. Conclusions

In order to offer high video quality, a certain end-to-end bandwidth has to be guaranteed. In bandwidth-limited networks, a single path with such a bandwidth is hard to be realized. To address this problem, a multipath approach can be used, in which the video data is transmitted over different paths to achieve the aggregate end-to-end bandwidth requirement. Since video data has different delay along different paths and has to be reassembled at the end host, there is a need to design a multipath routing algorithm with low user delay. In this paper, we study this issue by first formulating the routing problem. Under the general case where link bandwidth and delay are arbitrary, we present an efficient heuristic based on max-flow and shortest path algorithms. The complexity of the algorithm is only $O(|V|^3)$, where $|V|$ is the number of nodes in the network. Our results show that the multipath heuristic offers sufficient bandwidth with very high success rate while the delay is close to the shortest-path.

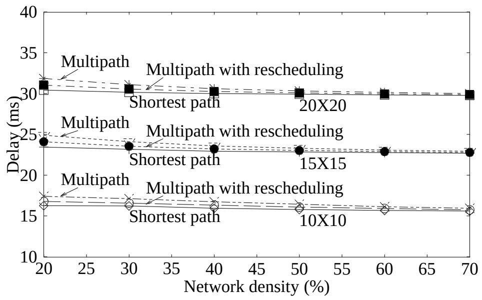


Fig. 10. end-to-end delay versus network density for multipath, multipath with video rescheduling, and shortest-path given a network size.

The multipath problem has an exact solution with a video rescheduling algorithm under the special case that each link in the network is of unit bandwidth. In this algorithm, the source partitions the video length into segments and transmits different segments along different paths in a manner such that the end-host can play back the video at the earliest time. Our results show that such algorithm is able to reduce the delay of multipath to be close to that of shortest-path.

In the future, we will study distributed algorithm for multipath routing, bandwidth and delay trade-off in the protocol design , and multicast routing for multi-tree transmission.

References

[1] B. Girod, K. Stuhlmuller, M. Link, and U. Horn, “Packet loss resilient Internet video streaming,” *Visual Communications and Image Processing ’99. San Jose, CA, USA*, June 1999.

[2] S. Sen, D. Towsley, Z.-L. Zhang, and JK. Dey, “Optimal multicast smoothing of streaming video over an internetwork,” in *Proceedings of Conference on Computer Communications*, pp. 455–463, IEEE, Sept. 1999.

[3] S. Satchell and HBJ. Clifford, “Internet video: small isn’t beautiful,” *BYTE*, vol. 21, p. 69, June 1996.

[4] J. W. Suurballe, “Disjoint paths in a network,” *NETWORKS*, vol. 4, no. 2, pp. 125–145, 1974.

[5] R. G. Ogier, V. Rutenburg, and N. Shacham, “Distributed algorithms for computing shortest pairs of disjoint paths,” *IEEE Transactions on Information Theory*, vol. 39, pp. 443–455, March 1993.

[6] Z. Wang and J. Crowcroft, “Qos routing for supporting multimedia applications,” *IEEE Journal on Selected Areas in Communications*, vol. 14, pp. 1188–1234, September 1996.

[7] W. Zhao, T. Seth, M. Kim, and M. Willebeek-LeMair, “Optimal bandwidth/delay tradeoff for feasible-region-based scalable multimedia scheduling,” in *Proceedings of IEEE INFOCOM ’98*, (New York, NY, USA), pp. 1131–1138, IEEE, March 1998.

[8] T. Kormaz and M. Krunz, “A randomized algorithm for finding a path subject to multiple QOS constrains,” in *IEEE Globecom’99 (Global Internet: Application and Technology)*, pp. 1694–1698, 1999.

[9] I. Cidon, R. Rom, and Y. Shavitt, “Analysis of multi-path routing,” *IEEE/ACM Transactions on Networking*, vol. 7, pp. 885–896, Dec. 1999.

[10] J. Garcia-Luna-Aceves, S. Vutukury, and W. T. Zaymen, “A practical approach to minimizing delays in Internet routing,” in *1999 IEEE International Conference on Communications*, (Piscataway, NJ, USA), pp. 479–483, 1999.

[11] B. Ramesh, *Survivable networks : algorithms for diverse routing*. The Kluwer international series in engineering and computer science, Boston: Kluwer Academic Publishers, 1999.

[12] K.-C. Leung and V. O. K. Li, “Flow assignment and packet scheduling for multipath networks,” in *Proceedings of IEEE INFOCOM*, 1999.

[13] W. Wen, B. Mukherjee, D. Ghosal, and S.-H. G. Chan, “uppercaseLVMSR-an efficient algorithm to multicast layered video,” 2000.

[14] T. H. Cormen, C. E. Leiserson, and R. L. Rivest, *Introducion to Algorithms*. The MIT electrical engineering and computer science series, MIT Press, 1998.