Multipath Rate Control for Real-Time Media

Balázs Kreith and Árpád Drozdy

Abstract—**In this paper, we present rate control algorithms designed for real-time media transmission over multiple paths. Our focus is on delivering media content simultaneously across multiple paths while maximizing the network traffic utilization on each path. The proposed algorithms ensure reduced network delay fluctuations, more consistent transmission rates for media content, and fairness to cross-traffic across all paths. To evaluate their performance, the solution is implemented and tested in an emulated networking environment under various test scenarios. The results demonstrate that applying our algorithms leads to reduced network delay fluctuations, improved structural similarity of the received media content, and enhanced fairness toward cross-traffic.**

Index Terms—**Multipath, RTP, Multimedia, Scheduling, Congestion Control, Real-Time Media**

I. INTRODUCTION

Web Real-time Communication (WebRTC) has enabled the widespread deployment of real-time multimedia communications on the Internet. Today, videoconferencing platforms (e.g., Google Meet, MS Teams, Discord) and cloud gaming services (e.g., GeForce NOW) have become integral to daily life. Most of these real-time communication (RTC) services rely on WebRTC [1].

The rapid growth of network traffic has driven efforts to improve transport protocols for multimedia systems, focusing on maximizing bandwidth utilization while avoiding packet losses caused by exceeding end-to-end path capacity.

Conversational multimedia communications impose strict latency limits to maintain the recommended mouth-to-ear delay of 150ms, ensuring fluent real-time conversations [2]. Unlike bulk data transfer, loss detection and retransmission may not be feasible in these scenarios, as delayed packets risk missing their playout deadlines at the receiver.

To address the challenges of reliable congestion control for real-time multimedia, the Internet Engineering Task Force (IETF) previously addressed this issue through the Real-Time Protocol Media Congestion Avoidance Technique (RMCAT) working group [3]. RMCAT introduced several congestion control algorithms [4]–[6], primarily designed for single-path delivery, with minimal focus on multipath scenarios.

Modern devices frequently feature multiple network interfaces, enabling endpoints to communicate via multiple, potentially disjoint paths. These paths can be leveraged for load balancing, capacity aggregation, or failover in case of path failures.

Balázs Kreith is with the Department of Telecommunications, University of Debrecen, Debrecen, Hungary. (E-mail: balazs@kreith.hu).
Árpád Drozdy is with Nokia, Budapest, Hungary. (E-mail: arpad.drozdy@nokia.com).

DOI: 10.36244/ICJ.2025.2.6

The Real-time Transport Protocol (RTP) [7], widely used for real-time media, is supported by the Real-time Transport Control Protocol (RTCP) for exchanging performance metrics. To extend RTP to multipath scenarios, Multipath RTP (MPRTP) [8] was introduced. While MPRTP provides mechanisms for shifting traffic between congested and non-congested paths, it does not fully explore or maximize the end-to-end path capacities. Achieving this requires individual congestion control algorithms for each path and a mechanism to coordinate them effectively.

In this paper, we propose a multipath rate control algorithm designed to fully exploit multiple paths for real-time media. Our solution achieves lower network delay fluctuations, improved media content similarity, and fairness for cross-traffic.

We introduce an architecture that builds on existing RMCAT congestion control algorithms enabling efficient multipath utilization. The novelty of our approach lies in coupling congestion control algorithms across paths to ensure fairness, prevent traffic fading during rebalancing, and optimize real-time media performance.

The remainder of this paper is organized as follows: Section II outlines our design goals for a multipath rate controller. Section III presents a high-level architecture compatible with existing congestion control algorithms. Section IV details our proposed algorithms, which are evaluated under various scenarios in Section V. Section VI discusses implementation considerations, Section VII reviews related work on congestion control in multipath communication, and Section VIII concludes the paper.

II. DESIGN GOALS

Our goal is to deliver real-time media over multiple paths simultaneously. To achieve this, we introduce a rate control mechanism that *1)* distributes media content across multiple paths simultaneously, and *2)* maximizes the network traffic sent over these paths.

A. Delivering media content over multiple paths simultaneously

In multi-path media transmission, packets are distributed across *subflows* (paths) as shown in Figure 1. These paths may have varying delays and jitter, requiring synchronization at the receiver. To manage this, we designed a **Multipath Packet Processing Module** (MPPM), which:

- Distributes packets based on allocated bitrates.
- Minimizes outbound traffic fluctuations to avoid self-inflicted congestion.
- Prevents traffic fading, where traffic exceeds a path's capacity due to unsynchronized bitrate changes.

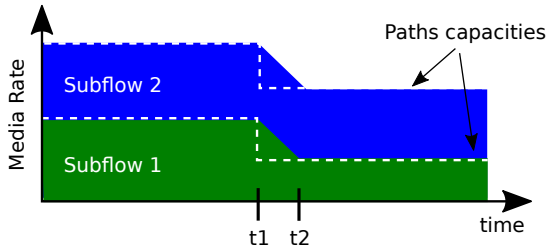

Fig. 1. Two subflows are shown. At $t1$, congestion is detected on $Subflow\ 1$, and the allocated bitrate is reduced. The media encoder adjusts to the new allocation, reaching the demanded bitrate at $t2$. Without synchronization, $Subflow\ 2$ would receive more traffic than its capacity between $t1$ and $t2$.

The MPPM updates packet distribution ratios promptly but must synchronize with the media encoder, which requires time to adjust. This synchronization avoids congestion or underutilization caused by temporary bitrate mismatches.

B. Maximizing the network traffic sent on paths

To maximize network utilization, a congestion control algorithm estimates and adjusts bitrates for each path. In single-path communication, the estimated bitrate equals the actual bitrate. For multi-path communication, separate estimations are needed for each path.

In our solution, the **Multipath Rate Control Module** (MRCM) ensures fairness across paths, following the fairness principles of MPTCP [9]. Fairness balances bandwidth across all paths and flows. To achieve this, the MRCM *1)* dynamically adjusts bitrates when congestion is detected, and *2)* increases bitrates when higher capacity is estimated, while maintaining fairness for cross-traffic.

Simulations and tests demonstrate how the MPPM avoids traffic fading and how the MRCM achieves fairness and efficiency in multi-path scenarios (see Sections V-A and V-B).

III. ARCHITECTURE

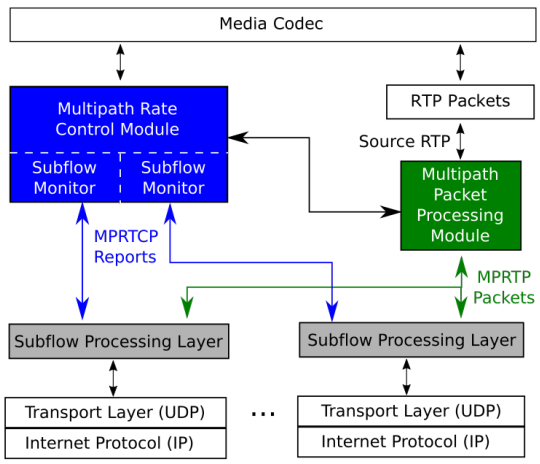

Fig. 2. Architectural scheme for Media Systems using Multipath Rate Control.

We propose a layered architectural model for a complex multimedia system (Figure 2), incorporating two modules to implement the multipath rate control mechanism: the Multipath Rate Control Module (MRCM) and the Multipath Packet Processing Module (MPPM). These modules operate between the layers responsible for media content and network transmissions. Additionally, Subflow Monitor (SM) submodules are applied to all subflows under MRCM's control. Note that the module implementations differ between the sender and receiver sides. For sending and receiving real-time media packets over multiple paths, we use Multipath Real-Time Protocol (MPRTP) [8], with subflow reports exchanged via Multipath Real-Time Control Protocol (MPRTCP).

To clarify the workings of the modules, we define the key types of sending bitrates used throughout the paper. The sending bitrate, representing the number of bits transmitted per second on a path, directly impacts media quality. In multi-path connections, the sender dynamically adjusts the sending bitrate for each path based on capacity estimations. The modules operate with the bitrate definitions presented in Table I as shown in Figure 3.

TABLE I
DEFINITIONS OF KEY BITRATE METRICS.

Metric	Notation	Description
Estimated bitrate	$SR_{estimated}$	Predicted bandwidth capacity of a link.
Actual bitrate	SR_{actual}	The bitrate currently transmitted on a path.
Allocated bitrate	$SR_{allocated}$	The bitrate calculated and assigned to a path.
Stable bitrate	SR_{stable}	The achieved and maintained bitrate by the media encoder.
Transient bitrate	$SR_{transient}$	A temporary bitrate during the encoders transition to a new allocation.

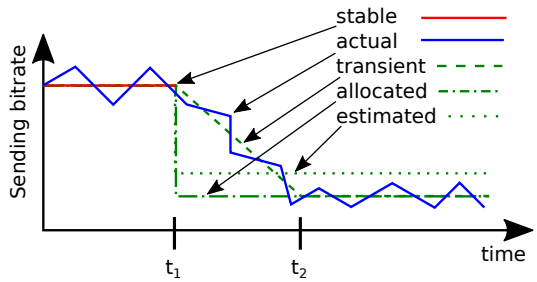

Fig. 3. Different types of sending bitrates during a reduction phase.

A. Subflow Monitor Module

Subflows can have varying characteristics, such as jitter, RTT, and bandwidth capacities, which may change during the connection. Therefore, each subflow is monitored by a *congestion control algorithm*, which evaluates the path characteristics and calculates the $SR_{estimated}$ for the monitored subflow. This information is forwarded to the MRCM on the sender side.

In addition to the $SR_{estimated}$, the SM submodule also informs the MRCM of the path state for each subflow. The path state can be *Congested*, *Stable*, or *Underused*. The MRCM uses this information to track the SR_{stable} when the stable state is reported. This state information is also utilized by the MPPM to avoid congested paths when possible. Based on the path state, the MPPM may assign packets to subflows, ensuring that, for example, I-Frame (reference points for decoding subsequent frames) packets are not sent on paths marked as *Congested*.

B. Multipath Rate Control Module

The MRCM operates only on the sender side. It sends $SR_{allocated}$, SR_{stable}, and path states to the MPPM. The MRCM calculates $SR_{allocated}$ from the subflows' $SR_{estimated}$ and SR_{stable} based on the following principles:

1) If $SR_{estimated} \leq SR_{stable}$ for a subflow, the algorithm assumes congestion and sets $SR_{allocated}$ equal to $SR_{estimated}$.
2) If $SR_{estimated} > SR_{stable}$ the algorithm assumes ramp-up and regulates $SR_{allocated}$ to ensure fairness, depending on how many subflows ramp-up simultaneously.

When a subflow reduces its $SR_{estimated}$, $SR_{allocated}$ is set to the new estimation to end congestion. Conversely, during ramp-up, $SR_{allocated}$ is less than or equal to $SR_{estimated}$, depending on the number of subflows ramping up. If only one subflow ramps-up, $SR_{allocated}$ equals $SR_{estimated}$. If multiple subflows ramp-up, $SR_{allocated}$ is reduced. If the total bitrate growth of all subflows does not exceed the maximum achievable growth of a single-path flow, fairness is ensured [9].

A key question is which subflows should determine the bitrate growth. The MRCM selects one subflow at a time and distributes its bitrate growth among all subflows. The MRCM executes the *fractional distribution algorithm* whenever a new $SR_{estimated}$ is reported, calculating the updated $SR_{allocated}$, which is sent to the MPPM with the SR_{stable} and path state.

C. Multipath Packet Processing Module

At the sender side, the MPPM distributes incoming traffic based on the information from the MRCM. This helps avoid traffic fading, as discussed in section II. Additionally, the MPPM considers real-time boundaries by selecting different paths for I- and B-Frames when possible. To assign a subflow to a packet, the MPPM runs the *packet scheduler algorithm* upon receiving a packet.

The *packet scheduler algorithm* converts RTP packets to MPRTP packets and assigns them to available subflows. It selects only subflows with a path state of *Stable* or *Underused*. Note that packet type information can be extracted only if the packet is not encrypted at the time of receipt, as discussed in section VI.

When subflows' $SR_{allocated}$ change, the media encoder responds with a time lag. This lag must be considered to avoid fluctuations that could lead to traffic fading. The scheduler calculates a $SR_{transient}$ for each subflow at every point in time when a packet is mapped, ensuring the distribution ratio is based on $SR_{transient}$ to allow seamless bitrate transitions. The packet scheduler algorithm is described in the next section.

Note that packets should only be sent through a subset of subflows if their SR_{actual} is smaller than $SR_{transient}$. Otherwise, fluctuations around $SR_{transient}$ are too high, increasing the risk of traffic fading.

At the receiver side, incoming MPRTP packets from different subflows are combined and played out to the Media Codec. A dejitter buffer designed for MPRTP packets from multiple subflows, as presented in [8], is used in our implementation.

IV. ALGORITHMS

To deliver media over multiple paths, algorithms are developed for the modules in section III. The MRCM uses the fractional distribution algorithm to allocate sending-bitrates for subflows, with each subflow monitored by a Subflow Monitor (SM) applying a congestion control algorithm. The MPPM uses the packet scheduler algorithm to map incoming packets to subflows based on allocated bitrates.

A. Congestion Control Algorithm

Each subflow is monitored by its own SM, which applies the FRACTaL [10] rate control algorithm. FRACTaL calculates the sending-bitrate for the path, which is forwarded as the $SR_{estimated}$ for the subflow. FRACTaL uses a state machine with REDUCE, KEEP, PROBE, and INCREASE states, mapped by the SM to path states defined in section III. If the inner state is REDUCE or INCREASE, the reported path state is *Congested* or *Underused*, respectively. Otherwise, the state is *Stable*, and the SR_{stable} is updated.

The SM sends the $SR_{estimated}$, SR_{stable}, and path state to the MRCM, which executes the fractional distribution algorithm upon receiving new information from any SM.

B. Fractional Distribution Algorithm

Algorithm 1 Fractional Distribution for Each Subflow

Require: The $total\_ramp\_up\_rate$ bitrate must be distributed among the subflows.
Require: An array of $weights$ contains a weight for each ramping-up subflow.
Require: The $total\_weight$, which is the sum of weights of all ramping-up subflows.

```
M̂R ← 0
subflows_total ← len(subflows)
for j = 1 → subflows_total do
    if ER[j] ¡ AR[j] then
        AR[j] ← ER[j]
    else
        ramp_up = total_ramp_up × weights[j]/total_weight
        if SR[j] ¡ AR[j] then
            ramp_up = MIN(AR[j] − SR[j], ramp_up)
        AR[j] ← SR[j] + ramp_up
    M̂R ← M̂R + AR[j]
```

The fractional distribution algorithm calculates $SR_{allocated}$ for each subflow, based on the principles

from section III-B. It takes $SR_{estimated}$, SR_{stable}, and path state from each subflow, reported by the Subflow Monitors.

If $SR_{estimated}$ is less than or equal to SR_{stable}, $SR_{allocated}$ is set to $SR_{estimated}$. Otherwise, let ΔSR be the positive difference between $SR_{estimated}$ and SR_{stable}. The algorithm selects the subflow with the highest ΔSR, and sets the $total\_ramp\_up\_rate$ to the highest ΔSR. The total rate increase of the ramping-up subflows should not exceed this rate to ensure fairness for cross-traffic, as described in section II.

A $weights$ array is computed, where each element is the ratio of the subflows ΔSR and Round Trip Time (RTT), allowing subflows with lower RTTs to ramp-up faster. $total\_weight$ is the sum of the $weights$. In practice, an encoder may not provide a new $SR_{allocated}$ immediately, leading to potential overlap of multiple ramp-ups. If a new $SR_{allocated}$ for a ramping-up subflow is smaller than the previous one, the smaller value is used.

The fractional distribution algorithm is shown in Algorithm 1. Note that fairness across multiple paths depends on each congestion control algorithm being fair on a single path, a topic beyond the scope of this paper. FRACTaL is a proactive congestion control algorithm that uses adaptive thresholds to detect congestion and compete with loss-based algorithms.

C. Packet Scheduler Algorithm

Algorithm 2 Map a Subflow to a Packet

Require: The $target^{(off)}$ value
Require: The $selected\_subflows$ array
  $selected \leftarrow NULL$
  $selected\_diff \leftarrow 0$
  for $j = 1 \rightarrow selected\_subflows_{total}$ **do**
    $target \leftarrow AR[j] \times target^{(off)} + SR[j] \times (1 - target^{(off)})$
    $diff \leftarrow target - SentBitrate[j]$
    if $selected$ **is** $NULL$ **OR** $selected\_diff$ ¡ $diff$ **then**
      $selected \leftarrow subflow$
      $selected\_diff \leftarrow diff$
  return $selected$

The MPPM executes the packet scheduler algorithm each time it receives a packet. It selects a subflow based on the packet size, $SR_{allocated}$, SR_{stable}, and the path state of each subflow. To avoid traffic fading, a $SR_{transient}$ is calculated as a weighted average of $SR_{allocated}$ and SR_{stable}, where the weight, $target^{(off)}$, is determined by how close MR_{actual} is to $MR_{demanded}$:

$$target^{(off)} = 1 - \left| \frac{MR_{demanded} - MR_{actual}}{MR_{demanded} - \sum SR_{stable}} \right| \quad (1)$$

When $target^{(off)}$ is 0, MR_{actual} has reached $SR_{allocated}$. The packet scheduler maps the subflow to a packet with the largest difference between its $SR_{transient}$ and SR_{actual}. The packet scheduler algorithm is presented in Algorithm 2.

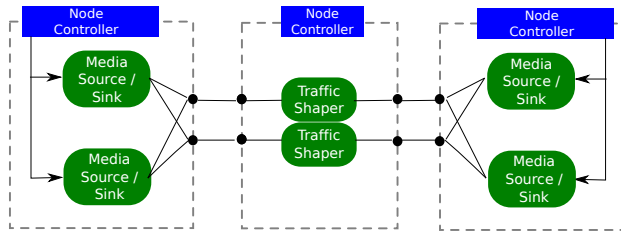

Fig. 4. Performance Evaluation Setup.

V. PERFORMANCE EVALUATION

In this section, we evaluate the performance of our proposal using multiple paths for real-time multimedia communications. Our algorithms are implemented with Gstreamer [11] and can be used with OpenWebRTC [12]. We test the algorithms effectiveness in avoiding traffic fading and ensuring fairness to cross-traffic.

Environment: The tests are conducted in a controlled environment (Figure 4), where the sender and receiver run on separate Linux machines connected via Ethernet. Network conditions are emulated with Netem [13] and tbf [14]. Traffic shapers set a 300 ms queue length with a drop-tail policy, and network delays in bottleneck links are 100 ms. We use the KristenAndSara [15] video sequence with VP8 [16], simulating a camera source at a clock rate of 90000 and a frame rate of 25 fps. Full details are available online[1].

Metrics: We measure Goodput (GP), Packet Loss Rate (LR), and Average Bandwidth Utilization (ABU). Queue Median Delay (QMD) is updated every 100 ms for packets sent in the last second. We also define Rate Error (RE) to evaluate traffic fading avoidance and use Jains fairness index [17] to assess fairness. The RE is calculated as:

$$\frac{SR_{actual} - SR_{allocated}}{SR_{allocated}} \quad (2)$$

Jains fairness index is normalized by the total throughput across all paths. We also measure video quality using SSIM [18] and MS-SSIM [19] over 2000 rendered frames.

A. Micro benchmark

We perform micro-benchmarks to assess throughput aggregation efficiency and the ability to avoid traffic fading.

Throughput Aggregations: In this scenario, we use one, two, and five subflows across three sub-scenarios. The paths mapped to subflows are disjoint, each with a 500 kbps bottleneck capacity. The test lasts for 120 s, and the subflows are joined at the start. The measurement summary is shown in Table II. FRACTaL starts conservatively, causing subflows to ramp up slowly. The fractional distribution algorithm manages the overall ramp-up, meaning that the media source using more subflows reaches the bottleneck capacity later than one with fewer subflows, resulting in a lower ABU in the short term for scenarios with more subflows.

[1] https://github.com/balazskreith/docker-gst-mprtp

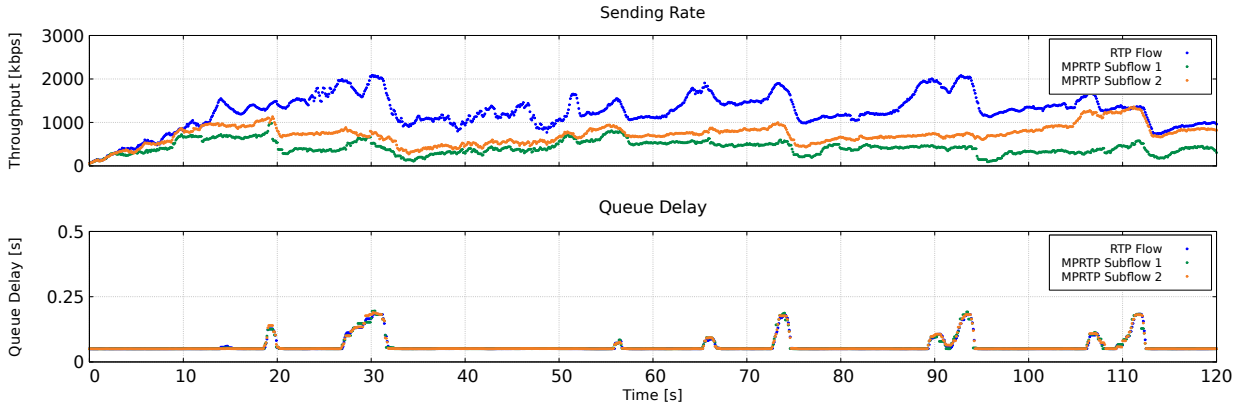

Fig. 5. The sending bitrate and the queue delay for a single- and a multipath capable media sources sharing the same bottleneck link.

TABLE II
SUMMARY OF THE MEASUREMENTS MEASURING THROUGHPUT AGGREGATIONS.

	1 Subflow	2 Subflows	5 Subflows
GP [kbps]	286 ± 17	515 ± 21	1205 ± 37
ABU [%]	76% ± 2%	67% ± 2%	60% ± 1%
LR [%]	0% ± 0%	0% ± 0%	0% ± 0%
QMD [ms]	107.81 ± 1.25	103.98 ± 0.54	102.66 ± 0.38

Avoiding Fading Effect: In this scenario, we evaluate the packet scheduler algorithm's ability to mitigate traffic fading. Two subscenarios are considered. In the first, one subflow uses a path with a bottleneck capacity varying between 500 kbps and 1000 kbps, with changes occurring every 25 s. In the second, two subflows are used, each with a bottleneck capacity varying between 500 kbps and 1000 kbps, but the aggregated capacity is 1500 kbps. The *Rate Error* (RE) is calculated for each subflow, and the results are summarized in Table III.
When the bottleneck capacity reduces on a path, FRACTaL detects congestion and recalculates the $SR_{estimated}$ for the subflow. The fractional distribution algorithm adjusts the $SR_{allocated}$ for each subflow. The MRCM requests a new $MR_{demanded}$ from the media source and forwards the updated $SR_{allocated}$ values to the MPPM. The MPPM then uses the packet scheduler algorithm to calculate the $SR_{transient}$, preventing traffic fading during the bitrate change. Consequently, the subflows' SR_{actual} fluctuates around the $SR_{transient}$, avoiding traffic fading.

TABLE III
SUMMARY OF MEASUREMENTS TEST FADING EFFECT BETWEEN SUBFLOWS

	1 Subflow	2 Subflows
GP [kbps]	369 ± 29	621 ± 28
RE [%]	16% ± 1%	16% ± 1%
LR [%]	1% ± 0%	0% ± 0%
QMD [ms]	110.2 ± 3.54	103.5 ± 0.74

Quality of Transferred Media Content: In this scenario, we compare the quality of the received media content with the original. We measure SSIM [18] and MS-SSIM [19] scores for the encoded media before sending and the decoded media after receiving. We use one, two, and five subflows, with disjoint paths and a total bottleneck capacity of 5000 kbps, divided equally among the subflows. SSIM and MS-SSIM are averaged over 2000 frames, with the average and standard deviation calculated over 30 measurements. The results, summarized in Table IV, show similar SSIM scores but better MS-SSIM scores when using multiple paths.

TABLE IV
SUMMARY OF MEASUREMENTS FOR SSIM AND MSSIM.

	1 subflow	2 subflows	5 subflows
SSIM	0.78 ± RUN	0.82 ± RUN	0.84 ± RUN
MSSIM	0.53 ± RUN	0.61 ± RUN	0.73 ± RUN

B. Fairness tests

We evaluate the fractional distribution algorithm's fairness by calculating Jain's fairness index for media sources using single and multiple paths. For a single path, fairness is the congestion control algorithm's ability to equally share the bottleneck capacity between flows. For multiple paths, fairness considers all flows across all paths.

Our test includes three scenarios: 1) A multipath source competes with a single-path source on a shared bottleneck; 2) A multipath source competes with a single-path source on a shared bottleneck and uses a non-shared one; 3) Two multipath sources compete over shared and non-shared bottleneck links.
One Bottleneck with Multiple Media Sources: We use two media sources on a shared bottleneck link, with one using two subflows and the other a single RTP flow. The shared bottleneck has a capacity of 3000 kbit/s (Figure 5). A summary of the measurements is shown in Table V.

Due to the fractional distribution algorithm, the multipath source ramps up more slowly, reaching saturation later than the single-path source. As a result, each subflow of the multipath source uses less bandwidth than the single flow. We calculated Jains index by normalizing throughputs with the available bandwidth of the used path. The Jains fairness index for this scenario is 0.97 ± 0.03. Due to FRACTaL's conservative nature around the bottleneck, its early congestion detection, and FEC protection during ramp-up, the flows do not experience packet loss.

TABLE V
SUMMARY OF MEASUREMENTS EVALUATING FAIRNESS BETWEEN A SINGLE- AND A MULTIPATH CAPABLE MEDIA SOURCE USING A SHARED BOTTLENECK.

	RTP flow	MPRTP flow 1	MPRTP flow 2
GP [kbps]	954 ± 180	464 ± 139	616 ± 113
ABU [%]	37% ± 6%	19% ± 5%	24% ± 4%
LR [%]	0% ± 0%	0% ± 0%	0% ± 0%
QMD [ms]	113.6 ± 4.36	113.62 ± 4.34	113.82 ± 4.15

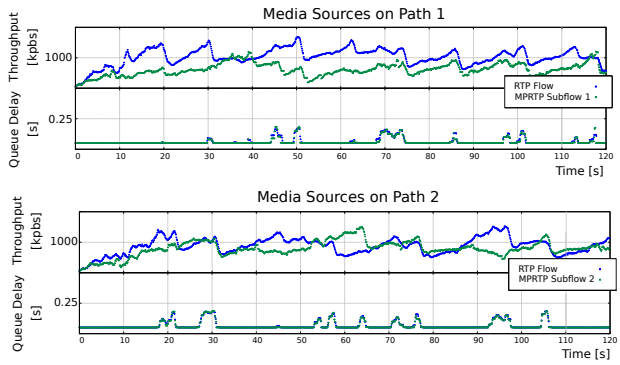

Fig. 6. The sending rate and the queue delay for a single- and a multipath capable media sources on different paths.

Two Bottlenecks with Two Media Sources: We use two media sources on a shared bottleneck link, with one using two subflows and the other a single RTP flow. Both paths have a capacity of 2000 kbit/s, and the test duration is 120 s. A summary of the measurements is shown in Table VI.

The multipath source ramps up slower than the single-flow source, leading to the single-flow source occupying a larger portion of the shared path. The Jains index is normalized by the aggregated bandwidth for multipath flows and by the single bandwidth capacity for single-path flows. The fairness index for this scenario is 0.98 ± 0.03.

TABLE VI
SUMMARY OF MEASUREMENTS EVALUATING FAIRNESS BETWEEN A SINGLE- AND A MULTIPATH CAPABLE MEDIA SOURCESS USING TWO DISJOINT PATHS.

	RTP flow	MPRTP flow 1	MPRTP flow 2
GP [kbps]	783 ± 161	428 ± 85	975 ± 401
ABU [%]	47% ± 8%	26% ± 4%	57% ± 20%
LR [%]	0% ± 0%	0% ± 0%	0% ± 0%
QMD [ms]	106.9 ± 2.75	106.2 ± 2.53	102.48 ± 1.95

Two Bottlenecks with Three Media Sources: We use three media sources on two shared bottleneck links, with one multipath capable source using two subflows across both paths, and one single-flow capable source on each path. Both paths have a capacity of 2000 kbit/s, and the test duration is 120 s (Figure 6). A summary of the measurements is shown in Table VII.

Due to the fractional distribution algorithm, the multipath source ramps up slower than the single-flow sources, resulting in a 2:1 bandwidth ratio in favor of the single-flow sources on each path. However, using aggregated throughput, the Jains index for this scenario is 0.95 ± 0.03.

TABLE VII
SUMMARY OF MEASUREMENTS EVALUATING FAIRNESS BETWEEN A SINGLE- AND A MULTIPATH CAPABLE MEDIA SOURCE USE DIFFERENT BOTTLENECK LINKS

	RTP flow 1	RTP flow 2	MPRTP flows
GP [kbps]	537 ± 151	487 ± 120	994 ± 100
LR [%]	0% ± 0%	0% ± 0%	0% ± 0%
QMD [ms]	106.92 ± 1.73	104.9 ± 1.5	105.91 ± 0.61

Two Bottlenecks with Two Multipath Capable Media Sources: We use two multipath capable media sources on two shared bottleneck links, each with a capacity of 2000 kbit/s, and the test duration is 120 s. The summary of the measurements is shown in Table VIII. The fractional distribution algorithm regulates both sources, ensuring they share bandwidth equally. The fairness index, calculated using aggregated throughput, is 0.99 ± 0.01.

TABLE VIII
SUMMARY OF MEASUREMENTS EVALUATING FAIRNESS BETWEEN TWO MULTIPATH CAPABLE SOURCES.

	MPRTP Flows 1	MPRTP Flows 2
GP [kbps]	565 ± 69	559 ± 98
LR [%]	0% ± 0%	0% ± 0%
QMD [ms]	106.31 ± 2.65	107.28 ± 1.89

C. Summary

This section evaluated the performance of our proposed algorithms: the packet scheduler and fractional distribution algorithms. We measured the packet scheduler's ability to avoid traffic fading and aggregate throughput. Fairness tests show that the fractional distribution algorithm ensures fairness among subflows across all paths. Our evaluation demonstrated that the packet scheduler algorithm successfully keeps the SR_{actual} around the $SR_{transient}$, avoiding traffic fading while aggregating path capacities. The fractional distribution algorithm regulates subflow ramp-up, leading to slower increases, but maintaining fairness for multiple paths and cross-traffic. No quality degradation was observed in video measurements using single- and multiple-paths.

VI. SYSTEMS CONSIDERATIONS

In this section, we detail a multimedia system implementing Multipath Rate Control using MPRTP [20]. The system sets up and tears down sessions (using RTSP or SIP), encodes/decodes media (e.g., VP8, VP9, H.265), applies SRTP encryption/decryption, and sends packets over multiple paths. It also supports DTLS, STUN, and ICE protocols for NAT traversal [21]. Figure 7 illustrates a component-based design for such a system based on our proposal.

The system has several components: Media Source/Sink (producing/consuming content), Media Encoder/Decoder (encoding/decoding RTP packets), packet sender/receiver, a dejitter buffer (for correct packet order), and a packet scheduler/rate controller (which maps RTP packets to subflows and controls bitrates).

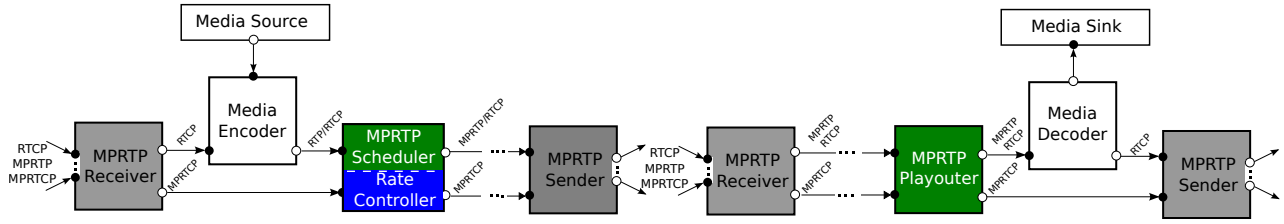

Fig. 7. High-level media pipeline design for a system implementing multipath rate control.

Both sender and receiver sides include a Multipath Sender and Receiver. The **Multipath Sender** transmits MPRTP/MPRTCP packets across mapped subflows, while non-MPRTP packets go through one path. The **Multipath Receiver** collects packets from all interfaces and outputs multiplexed MPRTP packets and non-MPRTP packets (e.g., MPRTCP, SDP, STUN).

On the sender side, the **Multipath Scheduler and Rate Controller** selects subflows for RTP packets and maps them to MPRTP packets. The Rate Controller monitors congestion and applies the fractional distribution algorithm to calculate the sending bitrate.

On the receiver side, the **Multipath Playouter** uses a dejitter buffer [8] to reorder packets for playback and sends MPRTCP packets for congestion control.

Compatibility

WebRTC applications use Secure RTP (SRTP) for encryption. The proposed subsystem enables RTP packets to operate over multiple paths without modifying other components. Since the Multipath Scheduler and Rate Controller alter RTP headers, security (e.g., encryption/decryption) must occur after header changes on the sender side and before playout on the receiver side.

VII. Discussion and Related Work

Much work has focused on congestion control for real-time media [4]–[6], [10], [22]–[25] and multipath capabilities for network protocols [26]–[29], but congestion control using multipath for real-time media is underexplored. Congestion control algorithms for real-time media impose strict delivery times. Many use packet loss as a congestion and bandwidth indicator [30], but retransmission is impractical as packets may miss the playout point. Monitoring queue delay improves reliability, but these algorithms are outperformed by loss-based methods. The IETF's RTP Media Congestion Avoidance Technique (RMCAT) working group developed congestion control algorithms considering queue delay fluctuations while competing with loss-based methods. NADA [6], [22] uses delay- and loss-based modes, estimating queuing delay via per-packet inter-arrival times. SCReAM [4] and Google Congestion Control (GCC) [5] model delay variation and apply filters for estimating queuing delay. FEC-based Rate Adaptation [23], [31] (FBRA) combines queue-delay congestion detection with error protection, while FRACTaL [10] improves FBRA for RMCAT. Multipath protocols like MPTCP [27], SCTP [32], and QUIC [29] use multiple paths for robustness. MPTCP design principles focus on cross-traffic fairness during bulk data transfer [9]. Recent MPTCP evaluations [33] highlight limitations in real-time applications, with cross-layer solutions for multipath communication in ad hoc networks [34] not optimized for real-time needs. For multimedia delivery, we used Multipath Real Time Protocol (MPRTP) [8], as MPTCP and SCTP dont consider real-time boundaries. MPRTP shifts traffic from congested to non-congested paths, though its path capacity exploration is limited. This research prompted further investigation into MPRTP with congestion control algorithms for all paths without media bitrate limitations. Optimized BBR [35] has shown promise in enhancing media delivery by optimizing bandwidth and reducing latency. Real-time media imposes strict time constraints, prohibiting bitrate throttling and retransmission. For interactive communication, media content may need rebalancing due to bandwidth limitations, with bitrate adjustments lagging behind production. Congestion control must ensure cross-traffic fairness [36]. Emerging research refines multipath congestion control, focusing on delay-based methods and machine learning to address real-time media delivery in heterogeneous networks [33], [36], [37], shaping future protocols and standards for real-time multipath communication.

VIII. Conclusions

This paper presented a rate control system for real-time media over multiple paths. The goal was to ensure fairness, avoid traffic fading, and efficiently utilize bandwidth. A congestion control algorithm was applied to all paths, with the ramp-up phase regulated for fairness. To avoid traffic fading, the transient sending bitrate and media source time lag were considered.

We proposed a layered architecture with the MRCM for congestion control and the MPPM for packet distribution. We developed algorithms for sending rate allocation and packet scheduling, implemented as a Gstreamer plugin. The system was tested in an emulated network environment, showing that our approach avoids traffic fading and maintains fairness.

We also discussed implementation details for multimedia systems and plan to apply the system in WebRTC, evaluating performance with RMCAT congestion control algorithms in future work.

REFERENCES

[1] N. Blum, S. Lachapelle, and H. Alvestrand, "Webrtc: Real-time communication for the open web platform," *Communications of the ACM*, vol. 64, no. 8, pp. 50–54, 2021.

[2] ITU-T, "G.114 One-way transmission time," *SERIES G: TRANSMISSION SYSTEMS AND MEDIA, DIGITAL SYSTEMS AND NETWORKS International telephone connections and circuits General Recommendations on the transmission quality for an entire international telephone connection*, pp. 1–20, 2003.

[3] IETF. (2017) RMCAT working group of the IETF. [Online]. Available: http://datatracker.ietf.org/wg/rmcat

[4] I. Johansson and Z. Sarker, "Self-clocked rate adaptation for multimedia," 2015.

[5] H. Alvestrand, S. Holmer, and H. Lundin, "A Google Congestion Control Algorithm for RTCWEB," 2011, iETF Draft, draft-alvestrand-rtcweb-congestion.

[6] X. Zhu, R. Pan, S. Mena, P. Jones, J. Fu, S. D'Aronco, and C. Ganzhorn, "NADA: A unified congestion control scheme for real-time media," 2013.

[7] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications," RFC 3550 (Internet Standard), RFC Editor, Fremont, CA, USA, pp. 1–104, Jul. 2003, updated by RFCs 5506, 5761, 6051, 6222, 7022, 7160, 7164, 8083, 8108. [Online]. Available: https://www.rfc-editor.org/rfc/rfc3550.txt

[8] V. Singh, S. Ahsan, and J. Ott, "Mprtp: multipath considerations for real-time media," in *Proceedings of the 4th ACM Multimedia Systems Conference*. ACM, 2013, pp. 190–201.

[9] D. Wischik, C. Raiciu, A. Greenhalgh, and M. Handley, "Design, Implementation and Evaluation of Congestion Control for Multipath TCP," in *Proc. USENIX NSDI*, 2011.

[10] B. Kreith, V. Singh, and J. Ott, "Fractal: Fec-based rate control for rtp," in *Proceedings of the 2017 ACM on Multimedia Conference. ACM*, 2017, pp. 1363–1371.

[11] gstreamer.org, "Gstreamer." [Online]. Available: http://gstreamer.org/

[12] openwebrtc.org, "OpenWebRTC." [Online]. Available: http://openwebrtc.org/

[13] S. Hemminger, "Network Emulation with NetEm," *Proceedings of the 6th Australian National Linux Conference (LCA 2005)*, no. April, pp. 1–9, 2005.

[14] Netem, "Traffic Controller for Netem." [Online]. Available: http://man7.org/linux/man-pages/man8/tc-netem.8.html

[15] Xiph.org, "YUV Sequences." [Online]. Available: http://media.xiph.org/video/derf/

[16] J. Bankoski, P. Wilkins, and Y. Xu, "Technical overview of VP8, an open source video codec for the web," in *Multimedia and Expo (ICME), 2011 IEEE International Conference on*, IEEE. IEEE, 2011, pp. 1–6.

[17] R. Jain, A. Durresi, and G. Babic, "Throughput fairness index: An explanation," in *ATM Forum contribution*, vol. 99, no. 45, 1999.

[18] Z. Wang, "The ssim index for image quality assessment," *https://ece.uwaterloo.ca/~z70wang/research/ssim*, 2003.

[19] Z. Wang, E. P. Simoncelli, and A. C. Bovik, "Multiscale structural similarity for image quality assessment," in *The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers*, 2003, vol. 2. IEEE, 2003, pp. 1398–1402.

[20] V. Singh, T. Karkkainen, J. Ott, and S. Ahsan, "Multipath RTP (MPRTP)," 2012, iETF Draft, *draft-singh-avtcore-mprtp*.

[21] T. Lévai, B. E. Kreith, and G. Rétvári, "Supercharge webrtc: Accelerate turn services with ebpf/xdp," in *Proceedings of the 1st Workshop on eBPF and Kernel Extensions*, 2023, pp. 70–76.

[22] S. D'Aronco, L. Toni, S. Mena, X. Zhu, and P. Frossard, "Improved Utility-Based Congestion Control for Delay-Constrained Communication," *IEEE/ACM Transactions on Networking (TON)*, vol. 25, no. 1, pp. 349–362, 2017.

[23] M. Nagy, V. Singh, J. Ott, and L. Eggert, "Congestion Control using {FEC} for Conversational Multimedia Communication," *arXiv*, no. c, pp. 1–13, 2013. [Online]. Available: http://arxiv.org/abs/1310.1582

[24] M. Engelbart and J. Ott, "Congestion control for real-time media over quic," in *Proceedings of the 2021 Workshop on Evolution, Performance and Interoperability of QUIC*, ser. EPIQ '21. New York, NY, USA: Association for Computing Machinery, 2021, p. 17. [Online]. Available: https://doi.org/10.1145/3488660.3493801

[25] A. S. Jagmagji, H. D. Zubaydi, S. Molnár, and M. Alzubaidi, "Utilizing machine learning as a prediction scheme for network performance metrics of self-clocked congestion control algorithm," *Infocommunications J.*, vol. 16, no. 3, pp. 2–17, 2024.

[26] J. R. Iyengar, P. D. Amer, and R. Stewart, "Concurrent multipath transfer using sctp multihoming over independent end-to-end paths," *IEEE/ACM Transactions on networking*, vol. 14, no. 5, pp. 951–964, 2006.

[27] A. Ford, C. Raiciu, M. Handley, and O. Bonaventure, "TCP Extensions for Multipath Operation with Multiple Addresses," RFC 6824 (Experimental), RFC Editor, Fremont, CA, USA, pp. 1–64, Jan. 2013. [Online]. Available: https://www.rfc-editor.org/rfc/rfc6824.txt

[28] K. Rojviboonchai and H. Aida, "An evaluation of multi-path transmission control protocol (m/tcp) with robust acknowledgement schemes," *IEICE transactions on communications*, vol. 87, no. 9, pp. 2699–2707, 2004.

[29] Q. De Coninck and O. Bonaventure, "Multipath quic: Design and evaluation," in *Proceedings of the 13th International Conference on emerging Networking EXperiments and Technologies*. ACM, 2017, pp. 160–166.

[30] A. Afanasyev, N. Tilley, P. Reiher, and L. Kleinrock, "Host-to-host congestion control for tcp," *IEEE Communications surveys & tutorials*, vol. 12, no. 3, pp. 304–342, 2010.

[31] J. O. Varun Singh Marcin Nagy, "Congestion Control Using FEC for Conversational Media," *draft-singh-rmcat-adaptive-fec*, 2013.

[32] S. Shailendra, R. Bhattacharjee, and S. K. Bose, "Mpsctp: A multipath variant of sctp and its performance comparison with other multipath protocols," in *2012 IEEE International Conference on Communication Systems (ICCS)*. IEEE, 2012, pp. 280–284.

[33] Ł. P. Łuczak, P. Ignaciuk, and M. Morawski, "Evaluating mptcp congestion control algorithms: Implications for streaming in open internet," *Future Internet*, vol. 15, no. 10, p. 328, 2023.

[34] M. Aljubayri, Z. Yang, and M. Shikh-Bahaei, "Cross-layer multipath congestion control, routing and scheduling design in ad hoc wireless networks," *IET Communications*, vol. 15, no. 8, pp. 1096–1108, 2021.

[35] S. Zhang and W. Lei, "An optimized bbr for multipath real time video streaming," *arXiv preprint arXiv:1901.09177*, 2019.

[36] M. Maliha, G. Habibi, and M. Atiquzzaman, "A survey on congestion control and scheduling for multipath tcp: Machine learning vs classical approaches," *arXiv preprint arXiv:2309.09372*, 2023.

[37] M. Pieska, A. Kassler, A. Brunstrom, V. Rakocevic, and M. Amend, "Performance impact of nested congestion control on transport-layer multipath tunneling," *Future Internet*, vol. 16, no. 7, p. 233, 2024.

Balázs Kreith is an experienced Software Engineer, focusing on Real-Time Communication Technologies (RTC). He graduated from the University of Debrecen, where he obtained his degree in Computer Science. Since 2016, he has been working in the field of RTC, specializing in developing solutions that enhance real-time communication performance, scalability, and reliability.

Árpád Drozdy graduated as an electrical engineer from the technical university of Budapest. He received his PhD. degree from Aalto University, Finland. He is specialized in 5G wireless telecommunications, holds three best paper awards, and currently works for Nokia.