Multipath Transmission for the Internet: A Survey

Ming Li, Andrey Lukyanenko, Zhonghong Ou, Antti Ylä-Jääski, Sasu Tarkoma, Matthieu Coudron, Stefano Secci

***Abstract*—Smart devices equipped with multiple network interfaces are becoming commonplace. Nevertheless, even though multiple interfaces can be used to connect to the Internet, their capabilities have not been fully utilized yet because the default TCP/IP stack supports only a single interface for communication. This situation is now changing due to the emergence of multipath protocols on different network stack layers. For example, many IP level approaches have been proposed utilizing tunneling mechanisms for hiding multipath transmission from the transport protocols. Several working groups under IEEE and IETF are actively standardizing multipath transmission on the link layer and transport layer. Application level approaches enable multipath transmission capability by establishing multiple transport connections and distributing data over them. Given all these efforts, it is beneficial and timely to summarize the state-of-the-art, compare their pros and cons, and discuss about the future directions. To that end, we present a survey on multipath transmission and make several major contributions: (1) we present a complete taxonomy pertaining to multipath transmission, including link, network, transport, application and cross layers; (2) we survey the state-of-the-art for each layer, investigate the problems that each layer aims to address, and make comprehensive assessment of the solutions; (3) based on the comparison, we identify open issues and pinpoint future directions for multipath transmission research.**

***Index Terms*—Multipath transmission, TCP-friendly, resource pooling, packet reordering**

I. Introduction

The Internet was originally designed as a "two-connected net" to guarantee that no single failure would cause any non-failed portion of the network to lose connectivity [113]. In essence, any source-destination pair needs to maintain more than one path to assure the reliability and resiliency of the network. Although the rich resources have been existing in the Internet, they have not been fully utilized since the birth of the Internet. The reason lies in the fact that, by default, the conventional TCP/IP only uses a single "best" path according to certain routing metrics; the other available paths remain standby only for backup and recovery purposes.

Nonetheless, this situation has been changing in the past few years, which is indicated by several trends from the standardization organization, academia, and industry. From the standardization perspective, both IEEE and IETF are active on concurrent multipath transmission. There have been several working groups dedicating on the standardization, for example, [2, 3, 11, 39, 51, 56, 61, 87, 112, 127] [1]. From the academia, hundreds of scientific articles revolving around multipath transmission have been published, covering different network stack layers on various aspects ranging from packet reordering, scheduling to buffer management, fairness, Resource Pooling (RP). From the industry, several companies have implemented their own link layer aggregation schemes, such as [15, 32, 94]. Deutsche Telekom offers hybrid access by bundling DSL-line with LTE in its portfolio [43]. Tessares company [1] tried to develop new innovative network services on top of Multipath TCP (MPTCP). For example, its first product aims to aggregate the bandwidths of different infrastructure (LTE/DSL). In 2015, OVH company [138] announced a new product called Overthebox. This product combines MPTCP and SOCKS proxies to enable users to bond different DSL lines together. Apple has implemented a variant of MPTCP on part of their Siri servers and allows iOS 7 users to use it in their iPhones [4]. At IETF'93 in 2015, KT Corporation presented Gigapath, a commercial service which can achieve high bandwidth (800 Mbps and more) by combining LTE and WiFi networks on Multipath TCP enabled smart phones [20].

There already exist a few survey articles focusing on different aspects of multipath transmission. For example, [86, 108, 154, 174, 193] mainly focused on the control plane problem (i.e., multipath routing of how to compute and select paths). [183] covered the control plane problem as well as the data plane problem (i.e., how to split the flow on the chosen paths) in wired networks. [153] assumed multiple paths had been established by routing protocols and focused on load distribution in terms of traffic splitting and path selection. [163] and [45] considered multipath transmission in wireless and wired networks respectively. [7] investigated the common features of various approaches and classified the features into layer-dependent and layer-independent features. In addition, [7] also abstracted common design patterns and proposed a unified networking architecture to enable mobile nodes to make context-aware decisions about how and when to use each or a combination of networks.

Nevertheless, to the best of our knowledge, this survey is the first one to provide a holistic view on the data plane issues of multipath transmission. We have surveyed papers from the year 1975 to 2015, and investigated various research problems from different layers, covering link layer, networking layer, transmission layer, application layer as well as cross layer. The primary research problems include packet reordering, fairness control, RP, Pareto-optimality and path diversity.

M. Li, A. Lukyanenko, and A. Ylä-Jääski are with the Department of Computer Science, Aalto Unviersity, Finland (email: {ming.li, andrey.lukyanenko, antti.yla-jaaski}@aalto.fi).

Z. Ou is with the School of Computer Science, Beijing University of Posts and Telecommunications, China (email: zhonghong.ou@bupt.edu.cn).

S. Tarkoma is with the Department of Computer Science, University of Helsinki, Finland (email: sasu.tarkoma@helsinki.fi).

M. Coudron and S. Secci are with Sorbonne Universités, UPMC Univ Paris 06, UMR 7606, LIP6 (email: {matthieu.coudron, stefano.secci}@upmc.fr).

[1][39] gives an overview of bandwidth aggregation mechanisms discussed in the context of Banana mailing list https://www.ietf.org/mailman/listinfo/banana

All frequently used acronyms in this survey are reported in the appendix.

A. Why Multipath?

As stated previously, the original Internet was designed as a "two-connected net" with path diversity in mind. Nevertheless, computers with multiple network interfaces were not an immediate design priority at the early stage. Only the routers were equipped with several physical network interfaces. However, the Internet has since then evolved significantly. For example, most servers have been equipped with more than one network interface nowadays. The abundance of network resources from the server domain has spurred the adoption of multipath transmission in data center networks. In the consumer electronics domain, the proliferation of mobile devices equipped with cellular (e.g., 3G and LTE) and WiFi interfaces, represented by smart phones, brings with it a growing number of multi-homed hosts onto the Internet. Thus, there exists a mismatch between single-path transport and the multitude of available network paths. Those multi-interface devices require multipath capability to improve end-to-end communication performance and resilience.

Meanwhile, technological advancement has made multipath transmission possible in theory. We investigate some of the major benefits as well as requirements, and give brief descriptions as follows.

- *Reliability*: multipath transmission can enhance the reliability of data transfer because additional paths can keep the connection alive in case of a failing or less performing path. In wireless environment, reliability can be further improved because signal interference is minimized due to the use of heterogeneous wireless access techniques.
- *Bandwidth aggregation*: it is expected that the bandwidth aggregation can potentially multiply the experienced throughput by the number of available paths. If efficient bandwidth aggregation can be achieved in this manner, a multi-homed device can obtain a much better performance.
- *Fairness and RP*: TCP fairness requires that a multipath transmission protocol receives no larger bandwidth of the shared bottleneck link than a competing TCP flow. This is important because TCP is the dominant transport protocol on the Internet. If new protocols acquire unfair capacity, they tend to cause problems such as congestion collapse. RP is a concept that changed the notions of fairness in a way that made multipath communications widely acceptable in practice. Instead of handling per path resource independently, the RP principle advocates making improved use of multiple path resources by allowing separate paths to act as if they were a single large resource. This principle is a significant step towards a practical multipath-aware end system.
- *Pareto-optimality*: it is a state of resource allocation in which there is no alternative state that would make some people better off without making anyone worse off. In the case of multipath transmission, it means that upgrading some regular single-path users to multipath ones can not reduce the throughput of other users with any benefit to the upgraded users.
- *Security*: as data can be distributed over independent paths, it will be more complex for a malicious entity to capture the entire content.

B. Potential Blocking Points

Multipath transmission also has its own challenges we must face. Some of the requirements mentioned in the last section can be seen as disadvantages as well. In this section, we investigate some of the potential blocking points from the perspective of deployment in practice.

- *Packet reordering*: it is difficult to schedule data packets over heterogeneous paths without causing reordering and performance penalties. A robust multipath transmission solution should be able to cope with any kind of path heterogeneity, throughput fluctuations, or jitter. It should also be able to deal with persistent reordering of data packets. Otherwise, users would have less incentive to upgrade if the solution fails to work properly in certain network environments.
- *Fairness*: traditionally, fairness has been one of the obstacles to the concurrent multipath transmission. It used to be on a "per interface" basis which is unfair if there is a common bottleneck later on. For example, simply utilizing multiple flows would result in an unfair share of the bandwidth at the bottleneck; for example, n TCP flows get approximately n times throughput as a competing single TCP flow does.
- *Compatibility*: it is hard to implement a generic multipath solution without modifying standardized protocols or changing third-party network equipments. For example, on link layer dedicated setup (even equipment) is required on both sides. On other layers above the link layer, either hosts or networks (sometimes both) need to upgrade in order to support multipath transmission.
- *Pareto-optimality*: MPTCP is the first multipath transmission proposal which requires Pareto-optimality [101, 102] but there is little experience how well/often it respects this requirement. There are indeed cases where MPTCP may perform worse than normal TCP due to path heterogeneity.
- *Path diversity*: it describes the ability of having multiple disjoint paths to reach a destination. Users expect to obtain high throughput from the use of multiple paths. But if the paths (or partial of them) travel through a shared bottleneck link, the multiple flows can only get as much throughput as a comparable TCP flow does due to the fairness guarantee. Currently, reliable bottleneck detection is really hard in practice. No individual path selection scheme can fit with all network environments.
- *Security*: multipath transmission has broken trust models organizations placed in single network providers. For example, although traffic splitting makes sniffing harder, firewalls or gateways may miss part of the data delivered over more than one network providers and thus become unable to analyze the flow. This would result in broken security solutions including intrusion detection and data leak prevention.

Note that this survey is neither limited to these problems nor cover all of them. We present the scope of this survey in section II.

Table I
CLASSIFICATION OF THE RESEARCH WORK BASED ON THE INTERNET PROTOCOL LAYERS.

Stack position	**Research work**
Link	Multi-Link PPP (MP) [185, 186], strIPe [9, 73], FatVAP [95], IEEE 802.1AX-2008 [2], EtherChannel [32], Aggregated Ethernet [94], Multi-Link Trunking [15], IEEE 802.1AX-2008 [2], OpenFlow [135, 188], IEEE 802.1aq [3], TRILL [51], SPB [57]
Network	Phatak et al. [148, 149], BAG [26, 27], PRISM [104, 105], ETOM [109], MAR [166], INTELiCON [122], MLP [52], SIMA [150], mHIP[65, 151], Sun et al. [189], LISP [56], OSCAR [68], LISP-HA [127]
Transport	BA-SCTP [13], W-SCTP [24], CMT-SCTP [8, 47, 48, 49, 50, 89, 90, 91, 120, 175], LS-SCTP [5, 6], cmpTCP [172], WiMP-SCTP [85], cmpSCTP [118], mSCTP-CMT [21], FPS-SCTP [129], R-MTP [121], Lee et al. [111], pTCP [82, 83, 84], R^{2}CP [81, 106], Cetinkaya et al. [25], mTCP [204], M-TCP [28], M/TCP [167], R-M/TCP [168], cTCP [46], MPLOT [178, 179], JOSCH [198], Super-aggregate [191], BMC [79], MPTCP [17, 60, 76, 101, 102, 110, 116, 140, 158, 160, 161, 162, 182], Han et al. [103], NC-MPTCP [114], FMTCP[37, 38], QoS-MPTCP [44], CWA-MPTCP [206], Openflow-MPTCP [194], Balia [142, 143], Coudron et al. [34], A-MPTCP [36], Yang et al. [201], SC-MPTCP [115], MPTCP-MA [119], EW-MPTCP [117], Yang and Amer [202], DRePaS [58], Coudron et al. [35]
Application	XFTP [10], PSockets [184], GridFTP [92], PA [165], ATLB [74, 75], Tavarua [155], SBAM [171], DMP [195, 196], MultiTCP [192], PATTHEL [16], Kaspar et al. [96, 97], Evensen et al. [53, 54, 55], Miyazaki et al. [130], DBAS [70, 71], G-DBAS [69], OPERETTA [66], MPTS-AR [112, 205]
Cross-layer	PRISM [104, 105], MPTCP-MA [119], ATLB [74, 75], Tavarua [155], SBAM [171], MultiTCP [192], PATTHEL [16], DBAS [70, 71], G-DBAS [69], OPERETTA [66], A-MPTCP [36], Openflow-MPTCP [194], Coudron et al. [34]

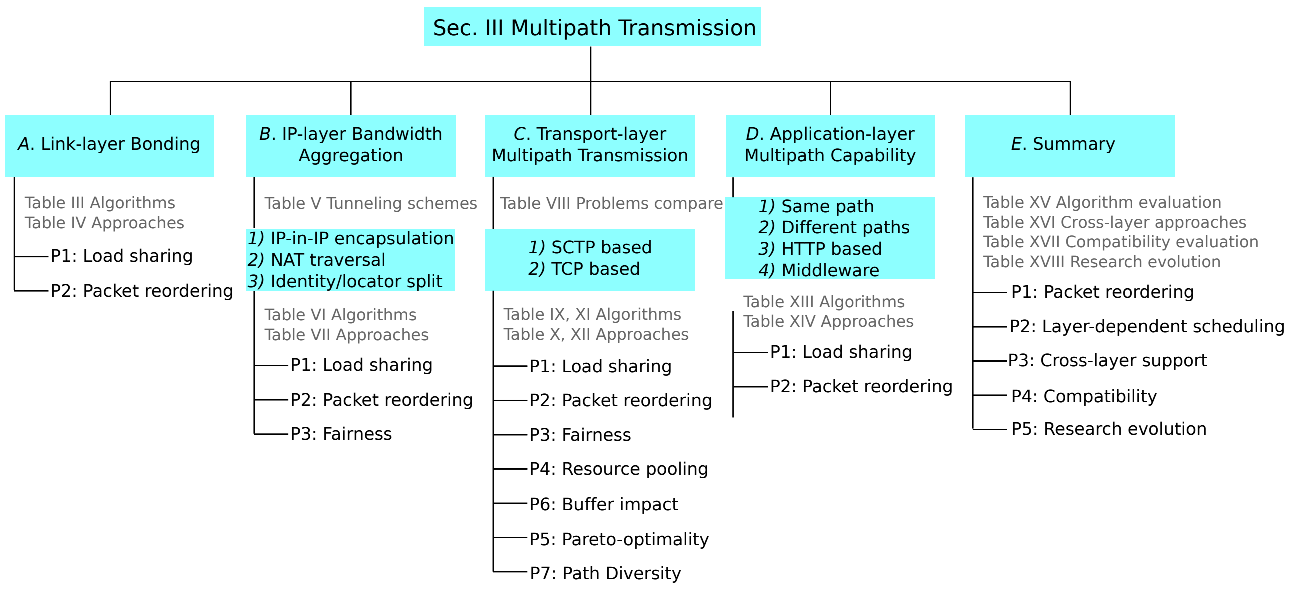

Figure 1. Structure of Section III and research problems (P) to address.

C. Contributions

We provide a comprehensive survey of multipath transmission, covering various aspects on different layers. Towards that direction, we make several key contributions and summarize them as follows: (1) a complete taxonomy regarding multipath transmission is presented, covering various protocol layers including link layer, network layer, transport layer, application layer and cross layer; (2) the state-of-the-art for each layer is surveyed, the problems addressed by layer specific approaches are investigated, and comprehensive comparisons among them are made; (3) the standardization efforts from various parties are summarized, including working groups from IETF and IEEE; (4) by the means of comparison, open issues are identified for future development of multipath transmission. We believe this work will bring insights to the researchers and practitioners working in this field, and foster a set of new research towards different directions of multipath transmission.

D. Organization, Structure, and Research Problems

Grouping and discussing multipath transmission approaches according to their stack position are beneficial for researchers and practitioners to understand the benefits and trade-offs from each layer, and make an all-around decision. Therefore, we survey the state-of-the-art multipath transmission from layer-specific perspectives. Table I shows the classification of the research work according to the stack position.

The structure of the survey is organized as follows. Section II reviews a number of related surveys and layouts the position of ours. In Section III, various approaches are classified based on their network stack position and cross-layer approaches are discussed separately (see Table I). Figure 1 illustrates the structure and coverage of Section III. In each discussion of the layer specific approaches, we investigate the problems the approaches on that layer aim to address. Some problems are common to all layers, such as the load sharing and packet reordering problems. Some are addressed only on certain layers. For example, the fairness problem is addressed only on IP and transport layers. Compared with other layers, transport layer approaches have more problems to address including RP, buffer impact, Pareto-optimality and path diversity. The discussion of approaches follows a chronological order except that we group some research work which has similarity or progression. In addition, two tables are used to summarize the key algorithms and approaches respectively. The approach table is connected to the key algorithm table by the means of listing the key algorithms used in each approach as well as the intended network environments of the algorithms. Note that the same algorithms, which are used on different layers, are not repeatedly described in different key algorithm tables. Instead, we only provide explanation in the table when the algorithm is first discussed. Following the discussion of the approaches on specific layers, we make a summary to present a comprehensive comparison from five perspectives. In Section IV, we point out the lessons that can be learned from this survey. In Section V, we pinpoint future research directions. Finally, we conclude this survey in Section VI.

In this article, there are many abbreviations. To help readers track them easily, we provide a list of frequently used acronyms in the appendix.

II. Scope and Related Surveys

First of all, we investigate multipath transmission in wired and wireless networks but leave its discussion in sensor networks out of the scope. For surveys on multipath transmission in sensor networks, we refer the readers to [156]. In addition, we focus solely on the data plane problem of how to split data on multiple paths and intentionally leave out all work that focuses on multipath routing, i.e. the control plane problem of how to compute and select the routes. We refer the readers to articles and recent surveys that cover such work [86, 108, 154, 174, 193]. Furthermore, the security issue of multipath transmission and P2P applications are also out of the scope of this survey.

Compared with surveys on multipath routing, surveys on the data plane problems are less popular. There are only a few surveys [7, 45, 67, 153, 154, 163, 183] that touch on the same topic as ours. In a somewhat old but still relevant survey [163], Ramaboli et al. reviewed some bandwidth aggregation approaches in heterogeneous wireless networks which consist of a variety of integrated and jointly managed radio access technologies. They found that packet reordering is the most dominant challenge because it can introduce undesirable delays for real-time applications and unnecessary retransmissions for TCP applications. In this regard, their survey focused mainly on the issues caused by packet reordering and the approaches to address them accordingly. Those approaches were classified into two groups according to their adaptiveness to dynamic conditions: non-adaptive and adaptive approaches. The former ones do not have the ability to adjust the resource allocation and traffic schedule in dynamic network conditions. In contrast, the latter ones take varying traffic and link conditions into consideration in order to derive optimal resource allocation and scheduling decisions. In each group, the approaches are further classified according to their layer position in TCP/IP protocol stack.

Prabhavat et al. [153] presented a literature review of various existing load distribution models for multipath networks, and classified the models in terms of their key functionalities: traffic splitting and path selection. A generalized multipath forwarding mechanism was used to discuss the two key functionalities without considering which protocol stack position the mechanism is implemented on. The paper did not address routing to establish multiple paths. Instead, it assumed that multiple paths had been established by routing techniques.

Domżał et al. [45] considered multipath transmission in wired networks. They classified the approaches based on three different layers in which they operate, i.e., link layer, network layer and transport layer. Specifically, on the link layer, they discussed a couple of multipath transmission approaches based on Ethernet. On the network layer, they investigated various routing techniques that can be used for the construction and selection of multiple paths. On the transport layer, they gave a brief introduction of MPTCP. However, [45] lacks many key approaches added in this survey. For example, on the network layer, all the tunneling based solutions are missing. On the transport layer, they only discussed MPTCP but have left out all the other approaches.

Sateesh et al's survey in [7] covers some aspects related to multipath transmission. The authors first reviewed some protocols and architectures that enable heterogeneous networking support, and then abstracted common design patterns and proposed a unified networking architecture to enable mobile nodes to make context-aware decisions about how and when to use each or a combination of networks. However, the scope of survey [7] is only partially overlapping with our work, in particular the focus is mostly shifted to seamless mobility, multihoming, security and other aspects that are out of the scope of our work.

Habak et al. [67] investigated the common features of various approaches and classified the features into two categories: layer-dependent and layer-independent features. The layer-dependent features include, for example, the common research problems shared by the approaches on the same layer. In the discussion of these features, they performed a layer by layer analysis of the available approaches. The layer-independent features include scheduling algorithms, estimation of interface/application characteristics, and networking models. In contrast, in this survey, we argue that some seemingly layer-independent features are not completely layer independent. For example, certain scheduling algorithms are closely connected to a specific layer so that they may perform differently on different layers.

Table II
COVERAGE OF RELATED SURVEYS ($\surd$ means having been covered).

Survey	Year	Application	Transport	IP	Link	Physical	Cross-layer	Network Environment
[163]	2012	√	√	Scheduling, tunneling	√			Wireless
[67]	2013	√	√	Scheduling, tunneling	√			General
[7]	2013		√	Tunneling				General
[45]	2015		√	Routing	√			Wired
[183]	2015	√	√	Routing	√	√	√	Wired
[154]	2015		√	Routing			√	General
This survey	2015	√	√	Scheduling, tunneling	√		√	General

Singh et al. [183] surveyed both multipath routing and provisioning. In the discussion of provisioning multiple paths between end hosts, they also followed a layer-based structure, from application layer to physical layer, to review the existing approaches. Although [183] has a similar structure as our survey, they have several key differences: 1) [183] excluded multipath transmission in wireless networks which are the primary environment for path diversity. Due to its focus only on the wired networks, it missed many key references added in this survey; 2) [183] lacked a detailed discussion, for example, which scheduling algorithms are used to avoid reordering in the receiver and how well these algorithms perform. Instead, this analysis is one of the main focuses in our survey; 3) compared to our survey, [183] has missed some important contents including whether the approaches considered fairness, implemented the Pareto-efficiency and resource pooling features, and what compatibility issues the approaches had on each layer.

Qadir et al. [154] investigated multipath transmission on the transport layer, despite their main focus on network-layer multipath routing. They organized their investigation by discussing five questions relating to how the multiple paths are used. The five questions have covered: 1) the number of paths to be used concurrently, 2) the configuration of how multiple paths work together (backup or concurrent), 3) the load balancing methods (static or dynamic), 4) the congestion control methods (coordinated or uncoordinated) and 5) the controller entity which performs load balancing and traffic engineering (host or network). In this survey, we organize the discussion of multipath transmission in a different way, for example, investigating the various problems to address on different stack positions (see Figure 1). We believe that these two different perspectives on the same target are complementary and would give readers more insights.

As we have discussed previously, most of the existing surveys have classified various solutions based on protocol layers (excluding [153]). In this regard, we use Table II to compare the coverage of the previous surveys as well as this survey on different protocol layers. In this survey, we summarize the key algorithms used on all the other layers except the physical layer (we refer the readers to [29, 199] for multipath transmission at the physical layer) and associate the algorithms to the problems they were designed to address. We extract the scheduling algorithms and compare their efficiency in terms of packet reordering and load sharing capabilities without considering their layer dependency. The approaches based on a cross-layer design are summarized and discussed in a separate section. The approaches on different layers are also evaluated from the viewpoint of compatibility capability. We also discuss the evolution of the research questions on multipath transmission and found that only the transport layer approaches following a nice evolution.

III. MULTIPATH TRANSMISSION

Before we dig into the technical details of multipath transmission, we present the timeline of its milestones in Figure 2 in order to give readers a general picture of its development. The first paper on TCP was published in 1974. In the following year, Dr. Maxemchuck proposed Dispersity Routing [126] in his Ph.D. dissertation to concurrently transmit data over multiple paths. From that point onward, various forms of multipath transmission have been proposed. For example, the idea of building multipath capability into TCP was, to the best of our knowledge, first suggested by Huitema [87] as an Internet draft in IETF in 1995. In 2002, the first 3G network to go commercially live was launched in South Korea, which promoted the proliferation of mobile devices equipped with multiple wireless interfaces. In 2006, Key et al. [100] used fluid-flow modeling to demonstrate that multipath transport can provide not only robustness but also balanced congestion in a stable manner. In the same year, Shakkottai et al. [176] used a non-cooperative pricing game to show that multihoming outperforms unihoming in terms of throughput and profit to the Internet service providers (ISPs). In 2008, Wischik et al. [200] investigated the RP principle, which makes a collection of resources behave like a single pooled resource. This principle is a significant step towards a practical multipath-aware end system. From 2009, IETF started to define and standardize MPTCP, which employs a coupled congestion control algorithm to achieve RP principle.

In the remainder of this section, the state-of-the-art multipath transmission schemes are classified according to which layer of the protocol stack the proposed approach performs at: link layer, network layer, transport layer and application layer.

A. Link Layer Bonding

High end workstations and data centers can easily saturate existing Local Area Networks (LANs). On the link layer, multipath transmission is typically called bonding or link aggregation because multiple physical channels are bundled (or aggregated) into a single logical channel. The primary

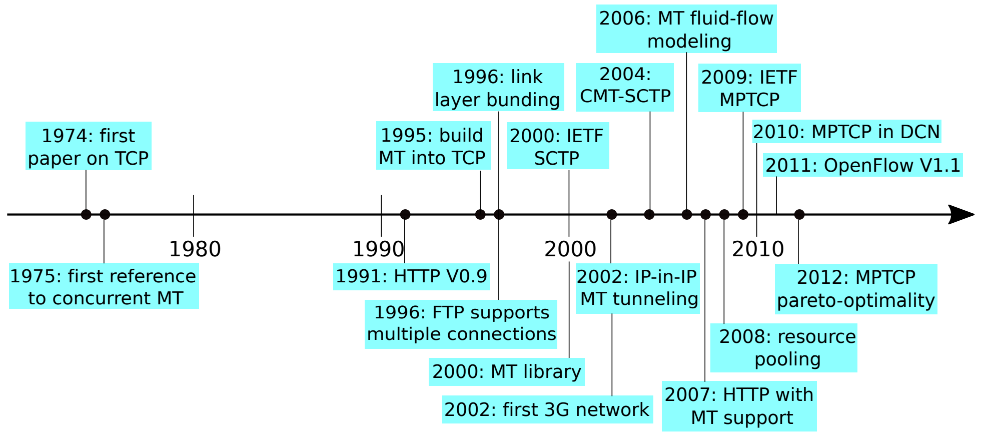

Figure 2. Milestones in the evolution of multipath transmission. MT: Multipath Transmission, DCN: Data Center Network, MPTCP: Multipath TCP.

Table III
KEY ALGORITHMS FOR LINK LAYER BONDING (sorted according to their order mentioned in Table IV).

Algorithm	Problems to address	Description
WRR (Weighted Round Robin)	Load sharing	It is designed to better distribute data onto paths with different capabilities. Each path is assigned a weight which indicates the path's transmission capability in terms of bandwidth, delay and packet loss (or partial of them). Data is distributed over different paths proportionally to their transmission capability.
FLSA (Fair Load-Sharing Algorithm), FQA (Fair-Queuing Algorithm)	Load sharing	A FLSA is obtained by transforming the operations of a Fair-Queuing Algorithm (FQA) in a time reversed manner. FLSA and the corresponding FQA need to run at the sender and the receiver respectively to provide a fair load sharing in the presence of variable sized packets and variable capacity channels.
PCA (Per-Conversation Allocation)	Load sharing, packet reordering	It allocates frames on a per conversation basis. For example, frames belonging to the same conversation is distributed only onto the same path. Multiple different conversations could share the same path.
PFA (Per-Flow Allocation)	Load sharing, packet reordering	It allocates traffic on a flow-by-flow basis. For example, traffic belonging to the same TCP flow is distributed only onto the same path. Multiple different flows could share the same path.
ECT (Equal Cost Tree)	Load sharing	It allows shortest path forwarding in an Ethernet mesh network context utilizing multiple equal cost trees. ECT supports much larger layer-two topologies than per-hop based ECMP.
ECMP (Equal Cost Multipath)	Load sharing	It is a routing strategy where next-hop packet forwarding to a single destination can occur over multiple "best paths". ECMP is a per-hop decision that is limited to a single router.
RR (Round Robin)	Load sharing	This simple scheduling algorithm orders paths and sends each piece of data on the next available/possible path in circular order.

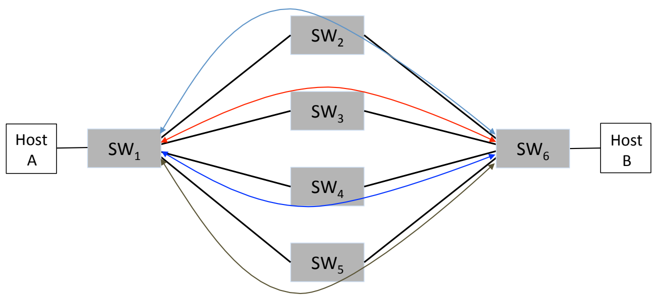

Figure 3. Link aggregation between Ethernet switches. SW: Switch.

goal of link layer bundling is to coordinate multiple independent links between a fixed pair of systems, providing a virtual link with a larger bandwidth than what a single link can sustain. Figure 3 shows a simplified example of link aggregation between two Ethernet switches (SW_1 and SW_6). These switches can obtain increased throughput by striping data across multiple interfaces.

In the following discussion, we use Table III and Table IV to summarize the key algorithms and approaches respectively. The approaches in Table IV are sorted in chronological order. The algorithms in Table III are sorted according to their order mentioned in Table IV. Note that the algorithms in Table III may be not only adopted by approaches on the link layer, but may also be used by those on other layers. In this survey, we will not elaborate the algorithms that have been discussed previously. This same rule is applicable for all other algorithm tables.

Multi-Link PPP (MP) [185], designed for Integrated Services for Digital Network (ISDN), aggregates multiple links using the PPP protocol [181]. In order to detect fragment loss and disorder, MP uses a 4-byte re-sequencing header

Table IV
SUMMARY OF LINK LAYER BONDING APPROACHES.

Scheme	Year	Algorithm and Protocol	Network Environment	Re-sequence Header
MP [185]	1996	WRR	ISDN	Yes
strIPe [9, 73]	1996, 1999	FLSA, FQA	General	Support
LQB [186]	1999	WRR	WWANs	Yes
LACP [2, 88]	2000, 2008	PCA	Ethernet	No
FatVAP [95]	2008	PFA	WAPs	No
SPB IEEE802.1aq [57]	2012	PFA, ECT	Ethernet	No
TRILL [51]	2014	PFA, ECMP	General	No
OpenFlow [135, 188]	2014, 2015	PCA	Ethernet, data center	No

(RSH) for synchronization and detecting lost fragments at the receiver. Therefore, a reorder buffer is required at the receiver to accommodate the out-of-order fragments caused by link aggregation. MP suggests a Weighted Round Robin (WRR) scheduling scheme so that data can be distributed proportionally to the transmission rates of the links. To achieve this goal, two methods for fragmentation have been proposed. The first one divides packets into segments with sizes proportional to the transmission rates of different paths. The other method divides packets into many small equal sized fragments and distributes the number of the fragments proportionally to the transmission rates of different paths.

Adiseshu et al. [9, 73] added a "strIPe" layer, a virtual IP interface below the IP layer and above the data link layer, to aggregate multiple data links. The stripe layer implements the striping algorithm at the sender and the fair queuing algorithm at the receiver. The authors first showed how a fair-queuing algorithm (FQA) can be transformed to a fair load-sharing algorithm (FLSA), and then proposed that the FQA should run in a reversed manner of the load-sharing algorithm in order to solve the load sharing issue with variable packet size. It implies that identical equipment (or of the same vendor) is required on both sides of the aggregation. They also dealt with the FIFO delivery issue for two separate cases. For example, if a RSH can be added to each packet, the issue can be solved by using the additional reordering number; if no header can be added, they proposed a way of synchronization in the event of frame loss to provide quasi-FIFO delivery.

An implementation of MP in Wide Area Networks (WANs) was discussed by Snoeren et al. in [186]. They proposed a Link Quality Balancing (LQB) scheme to bundle multiple channels of the same Wide-area Wireless Access Network (WWAN) technology. In order to adjust traffic striping across bundled links according to their transmission capabilities, LQB adapts the maximum transmission unit (MTU) of each link in proportion to its available bandwidth (short-term averages of the observed throughput). A link layer receive buffer is also required to reorder fragments.

FatVAP (an 802.11 driver design) [95] is another work in a wireless environment for link bonding. FatVAP aggregates the bandwidth available at multiple wireless access points (WAPs) that are worth connecting to and balances their loads by scheduling traffic to different APs according to their available bandwidth. In order to continue delivering the sum of the bandwidths available across all APs, FatVAP uses a constant estimation of both end-to-end and wireless bandwidth to react to changes within a few seconds. Note that FatVAP uses a Per-Flow Allocation (PFA) strategy to distribute traffic over APs. For example, when a new flow arrives, FatVAP determines which AP to assign this flow to and records the mapping in a hash table. Subsequent packets in the flow are simply sent through the AP recorded in the hash table.

Within IEEE specifications, the Link Aggregation Control Protocol (LACP) allows multiple links of Ethernet to be aggregated together to form a Link Aggregation Group (LAG). As such, the Media Access Control (MAC) client can treat the LAG as a single link. LACP allows a network device to negotiate an automatic bundling of links by sending LACP packets to the peer. LACP was initially released as 802.3ad [88] in 2000. Nearly every network vendor quickly adopted this standard over their proprietary standards. In 2008, the protocol was transferred to the 802.1 group with the publication of IEEE 802.1AX-2008 [2]. In LACP, a Frame Collector (FC) at the receiver is responsible for maintaining any frame ordering constraint. In order to avoid frame reordering, the Frame Distributor (FD) at the sender transmits all frames that compose a given conversation[2] only to a single link, which is a Per-Conversation Allocation (PCA) strategy (very similar to PFA strategy). Therefore, no frame reordering scheme or reordering buffer is required at the FC. In addition to the IEEE link aggregation standards, there are a number of proprietary aggregation schemes, including EtherChannel [32] from Cisco, Aggregated Ethernet [94] from Juniper, Multi-Link Trunking [15] from AVAYA. These proprietary aggregation schemes and IEEE 802.3ad standards are very similar and accomplish the same goal.

Since the version 1.1 [33], OpenFlow has supported multi-link aggregation on layer-2. The specification of OpenFlow switch has introduced Link Aggregation (LA) to obtain the ability for one port to point to a group of other ports. Using LACP for exchanging dynamic information between LA-supported devices, the OpenFlow controller has full control over the switches on how frames are distributed and collected on multiple links. Nguyen-Duc et al. [135] investigated the operation of LA in OpenFlow switches and found that LACP on OpenFlow switch provides a slightly lower throughput than the one on a conventional switch. Thus, OpenFlow switches need to be further optimized to achieve equivalent performance.

[2]A set of frames transmitted from one end station to another, where all of the frames form an ordered sequence, and where the communicating end stations require the ordering to be maintained among the set of frames exchanged.

Subedi et al. [188] presented an adaptive multipath forwarding architecture in a layer-2 OpenFlow data center network. In the architecture, all-to-all forwarding paths are set up proactively among the edge nodes. Aggregated bandwidth is achieved by using all the available paths simultaneously. To avoid the out-of-order delivery issue due to using all available paths, a PCA style scheduling algorithm is used in [135, 188]. Specifically, the algorithm excludes the paths whose path length exceeds the shortest path length significantly.

In the last few years, there are notable new protocols designed to support multipath forwarding at link-layer in IEEE and IETF standards, e.g., Shortest Path Bridging (SPB) [57] (specified in the IEEE 802.1aq standard [3]) and IETF Transparent Interconnection of a Lot of Links (TRILL) [51]. SPB and TRILL are potential successors of the Spanning Tree Protocol (STP). SPB now supports multipath forwarding by using Equal Cost Multipath (ECMP) and Equal Cost Tree (ECT) routing strategies. TRILL currently only supports multipath forwarding by using ECMP routing strategy. In both ECMP and ECT strategies, if multiple equal cost paths are present towards a destination, network traffic is distributed over those multiple paths. In order to avoid reordering and path MTU discovery problems, similar to FatVAP, both TRILL and SPB use per-flow multipath forwarding. Specifically, frames belonging to the same data flow take the same path and frames belonging to other flows can take the other paths.

From Table III, we find that the main problems the algorithms trying to address are load sharing and packet reordering. Among all the algorithms, the simplest one is Round Robin (RR) where the sender allocates fragments among all the available links in equal portions and in an ordered fashion. In the long run, RR scheduling provides a fair share of fragments as long as these fragments are of the same size. Nevertheless, the basic RR scheduling strategy is rarely used in practice because it provides no load sharing with either variable sized fragments or different link capacities. In order to solve these issues, the Weighted RR variations are the more widely used scheduling strategies.

The main advantage of link-layer bonding is that the signaling rate of the channel is relatively stable and can be utilized to mitigate reordering. However, link-layer approaches only work on a point-to-point link and even require dedicated Ethernet cards installed on both sides. Thus, they are not applicable in general scenarios of end-to-end communications where the different domains involved are controlled by different providers.

B. IP Layer Bandwidth Aggregation

The IP layer, originally proposed to handle global addressing and routing, is a natural candidate to host the multipath capability to enhance end-to-end communication. A network approach has the advantage of being transparent to transport protocols and applications, making wide spread deployment much easier. In theory, each packet of a TCP flow can be sent over a different path, and the IP protocol ensures that all packets reach their destination. For example, Sun et al. [189] explored the use of multipath routing to reduce the file transmission delay in a wireless network. Specifically, they proposed taking advantage of packet level erasure code (e.g., digital fountain code) to transmit data file with redundancy over a set of paths. They obtained the intuitive understanding of the trade-off between the code rate and delay reduction. Their research was made for a special network environment where a source and destination pair has a rich set of identical and disjoint paths, hence no packet reordering issue introduced. Nevertheless, in most practical network environments, when packets inside one connection taking more than one path, they can experience different propagation delay and arrive out of order. The TCP receiver sends duplicate acknowledgments (ACKs) to the sender, which causes the TCP sender mistakenly interprets packet reordering as packet loss. The results found in [19, 104, 203] show that TCP suffers significant performance degradation due to frequent packet reordering. Thus, the use of multiple paths with varying characteristics deteriorates the problem.

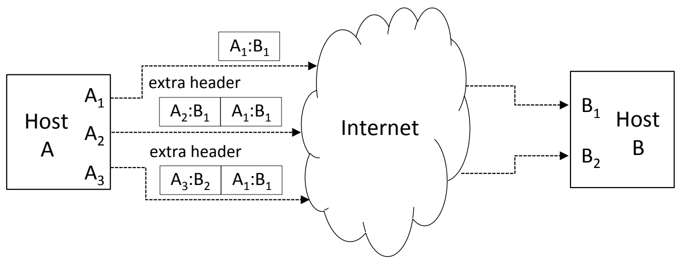

Figure 4. IP-in-IP tunneling between two multi-homed hosts.

In the following discussion, the state-of-the-art is divided into three categories: IP-in-IP encapsulation, Network Address Translation (NAT) traversal, and Identity/locator split. We summarize their features in Table V and discuss each category according to the order they show in the table. Table VI and Table VII are used to summarize the key algorithms and approaches respectively. The “Proxies or Updated Routers” in Table VII indicates the required number of proxies or updated routers.

1) IP-in-IP Encapsulation: A widely used IP layer approach for aggregating bandwidth of multiple IP paths is to use tunneling mechanisms which transparently redirect packets between two hosts on routing level. For example, Phatak et al. [148, 149] proposed using IP-in-IP encapsulation [144] to split a data flow across multiple network interfaces. As shown in Figure 4, at the source (A), the transport layer assembles all packets as if they were going through A_1 and addressed to B_1. The packets going out on interface A_2 get encapsulated in new IP packets each with an extra header having destination B_1 and source A_2. Likewise, each packet going out on interface A_3 can be encapsulated in a new IP packet having destination B_2 and source A_3. The destination (B) can then recognize IP-in-IP packets and strip the outer header. This leaves the original packets with source A_1 and destination B_1 to be delivered up the network stack to TCP in a transparent manner. The same encapsulation scheme is used for tunneling in the mobile IP standard [145]. In order to avoid fast retransmission, Phatak et al. used a WRR style scheduler, which distributes packets

Table V
SCHEMES USED ON IP LEVEL FOR BANDWIDTH AGGREGATION.

Scheme	Description	Update
IP-in-IP encapsulation	*No Proxy*: The client and server open a TCP connection with an agreed IP for each other. When a packet is sent through interfaces with IPs other than the agreed one, the packet is encapsulated in another packet with the agreed IP.	Endpoints
	One proxy at the client side: A proxy is required. IP-in-IP encapsulation is running between the proxy and the client to hide the usage of multiple IPs from TCP. The server which is unaware of the client's multiple IPs communicates with the proxy using normal TCP.	Client, network
	Two proxies at the both sides: Two proxies on the client and server sides are required. IP-in-IP encapsulation is running between the proxy client and server. Each endpoint communicates with the proxy (client or server) with normal TCP. The usage of their multiple connections is hidden from both endpoints.	Network
NAT (Network Address Translation)	*No Proxy*: The client and server agree with one IP for each other. The source and destination IPs at the client are replaced with the agreed ones. Upon receiving a packet, the server reverses its source and destination IPs using the agreed ones before forwarding it to TCP.	Endpoints
	One proxy at the client side: One proxy is required. NATing is running between the proxy and the client to hide the usage of multiple IPs from TCP. The server which is unaware of the client's multiple IPs communicates with the NAT box using normal TCP.	Client, network
	Two proxies at the both sides: A proxy client and sever are required. NATing is running between the proxy client and server. Each endpoint communicates with the proxy (client or server) with normal TCP. The usage of multiple connections between proxies is hidden from both endpoints.	Network
Identity/Locator Split	*Host-level:* The identity of a host is separated from its location (i.e., IP address). Each host uses its globally valid identity to shield the presence of its multiple IPs from transport and application layers.	Endpoints
	Network-level: The IP space is separated into two spaces, one for identity of a host and the other for locator of a border router. A mapping system is required to provide mapping between the identity and locator. Multipath transmission could be provided between source and destination border routers for the purpose of traffic engineering.	Network

proportionally to the effective rates of the paths.

Chebrolu et al. [27] presented a network layer architecture to aggregate bandwidth on multiple paths for real-time applications. They made the assumption that an infrastructure proxy (like the Home-Agent in Mobile IP [146]) is aware of the multiple interfaces of the client, and tunnels the captured packets to the client using IP-in-IP encapsulation. The advantage of a proxy solution is that it is fully controllable and allows servers to remain unchanged and hide using multiple IPs from TCP. Chebrolu et al. proposed a scheduling algorithm, Earliest Delivery Path First (EDPF), to ensure that packets meet their playback deadlines by scheduling packets based on the estimated delivery time of the packets. To improve the overall performance of IP-in-IP tunneling based bandwidth aggregation by the means of minimizing packet reordering, Chebrolu et al. in [26] proposed a two-pronged approach. Firstly, a scheduling policy Packet-Pair based EDPF for TCP applications (PET) was used to partition traffic onto different paths. The design of PET has the same concept of EDPF but with idealized delay and bandwidth values replacing the estimates. Secondly, working together with the scheduling policy, a receiver-side Buffer Management Policy (BMP) was used to delay forwarding the out-of-order packets to TCP and to detect losses, so that a variety of adverse effects can be hidden.

Kim et al. [104, 105] introduced PRISM, another proxy based approach that enables TCP to efficiently utilize the WWAN connections from community members. The proxy can be a trusted party or a community member. PRISM uses a cross-layer approach that involves support from both transport and network layer. We classify PRISM as network layer approach because the PRISM proxy, which is the main entity for multipath support, is located on the network layer. PRISM uses a packet-scheduling algorithm, i.e., Adaptive Scheduler (ADAS), to maintain up-to-date path state. Using the up-to-state information, ADAS sends packets according to their expected arrival time (a variant of EDPF algorithm) to reduce packet reordering. ADAS also uses the path state to adjust path weight by using the Additive Increase and Multiplicative Decrease (AIMD) strategy from TCP, so that ADAS can dynamically react to congestion from partial paths and control the amount of traffic to be allocated on those paths. Moreover, PRISM masks the effects of out-of-order delivery by identifying spurious duplicate ACKs and re-sequencing them so that a TCP sender receives correctly sequenced ACKs.

Lan et al. [109] designed a different proxy based multipath network protocol called Enhancements for TCP On a Multihomed mobile router (ETOM) that runs transparently to both clients and servers. ETOM involves two proxy components instead of one kind: MR (Mobile Router) and HA (Home Agent). The client and the MR (as well as the server and the HA) use normal single-path connection, whereas all packets traveling between MR and HA are IP-in-IP encapsulated. ETOM uses a reordering buffer to eliminate packet reordering. For example, out-of-order packets are buffered at the HA until the missing packets are received, and packets are then sent out in order to the destination. ETOM also uses a variant of EDPF algorithm to further reduce packet reordering. Note that unlike other IP layer approaches, ETOM employs a subflow sequence

Table VI
KEY ALGORITHMS FOR IP LAYER BANDWIDTH AGGREGATION (sorted according to their order mentioned in Table VII).

Algorithm	Problems to address	Description
PET (Packet-Pair based EDPF for TCP applications)	Load sharing, packet reordering	It sends TCP packet-pairs on each path periodically to compute inter-arrival time between the hosts, and schedules packets on the path that delivers it the earliest. PET is a variant of EDPF.
BMP (Buffer Management Policy)	Spurious retransmission, packet reordering	It is designed to hide any residual reordering from TCP at the data receiver side so that unnecessary retransmissions are avoided. For instance, the receiver buffers out-of-order data packets at the network layer before passing them to TCP in order.
EDPF (Earliest Delivery Path First)	Load sharing, packet reordering	It estimates the delivery time of the packets on each path, and schedules each packet on the path that delivers it the earliest. This approach is used to minimize reordering and thereby the delay and jitter experienced by the application.
RPC (Reverse Path Controller)	Spurious retransmission, packet reordering	It is designed to handle spurious duplicated ACKs in the data sender side so that unnecessary retransmissions are avoided. For example, RPC exploits TCP's control information carried by ACKs, determine the meaning of duplicated ACKs, corrects them if necessary.
SACK (Selective Acknowledgment)	TCP performance	It is sent from the receiver to the sender informing the sender of the out of order data that has been received. The sender can then retransmit only the missing data segments.
DATA/ACK SEP	TCP performance	It separates the forward (DATA) and the backward (ACK) traffic on different paths.
PBCS (Piggy-Backing for Control Signaling)	Path status	It adds piggy-backing extra information on packets before injecting them into the networking stack for transmission. The information is stripped out at the recipient.
TFCC (TCP-Friendly Congestion Control)	Fairness	It restricts the subflows of one TCP connection to use more bandwidth than normal TCP does at a shared bottleneck.
PRM (Packet Reordering Module)	Spurious retransmission, packet reordering	It runs at both sides of a communication to handle packet reordering issue. Specifically, it delays the data packets at the receiver and their ACKs at the sender before forwarding them to the upper layer. To avoid over-protection, it only delays forwarding them before the timeout.

Table VII
SUMMARY OF IP LEVEL BANDWIDTH AGGREGATION APPROACHES.

Scheme	Year	Tunneling	Proxies or Updated Routers	Algorithm and Protocol	Fairness	Network Environment	Sequence Space
Phatak et al. [148, 149]	2002, 2003	IP-in-IP encapsulation	0	WRR	No	Mobile	Single
MAR [166]	2004	NAT	1 or 2	WRR, PFA	No	Mobile	Single
BAG [26]	2005	IP-in-IP encapsulation	1	PET, BMP	No	Wireless access	Single
PRISM [104, 105]	2005, 2007	IP-in-IP encapsulation	1	EDPF, RPC, SACK	No	Mobile collaborative	Single
BAG [27]	2006	IP-in-IP encapsulation	1	EDPF	No	Wireless access	Single
SIMA [150]	2006	Identity/locator split	0	PFA	No	HIP-enabled	Double
INTELiCON [122]	2008	NAT	0	DATA/ACK SEP, PBCS	No	Wireless access	Single
mHIP [65]	2009	Identity/locator split	0	EDPF	No	HIP-enabled	Double
MLP [52]	2009	NAT	1	WRR	No	Wireless access	Single
mHIP [151]	2011	Identity/locator split	0	TFCC	Yes	HIP-enabled	Double
ETOM [109]	2012	IP-in-IP encapsulation	2	EDPF, BMP	No	Wireless	Double
LISP [56]	2013	Identity/locator split	2	WRR	No	General	Single
OSCAR [68]	2014	NAT	0	PRM, PFA, WRR	No	Mobile collaborative	Single
LISP-HA [127]	2015	Identity/locator split	2	PFA	No	Hybrid access	Single

number in the inner IP header to detect packet loss between MR and HA.

2) NAT Traversal: Unlike the previous approach that relies on IP encapsulation, there exist several approaches taking advantage of NAT instead of tunneling.

Rodriguez et al. [166] introduced MAR system (see Figure 5), a commuter mobile access NAT router that provides a set of local interfaces and a number of wide-area wireless interfaces. The former provides access to local mobile devices and the latter accommodates a variety of wide-area wireless technologies. The MAR router acts as a NAT box that is located in the middle and translates IP addresses and ports of packets for two directions. A MAR router can work alone or cooperate with a MAR proxy-server. With such a proxy-server, the Packet-Oriented Scheduling Mode (POSM) is used where the packets of the same TCP flow can be delivered over multiple paths and a MAR router can implement intelligent optimization including avoiding TCP 3-way handshake, slow-start, spurious timeouts and so on. When the proxy-server is absent, the Flow-Oriented Scheduling Mode (FOSM) is used where a per-flow allocation strategy schedules all packets belonging to the same TCP flow onto the same path. MAR

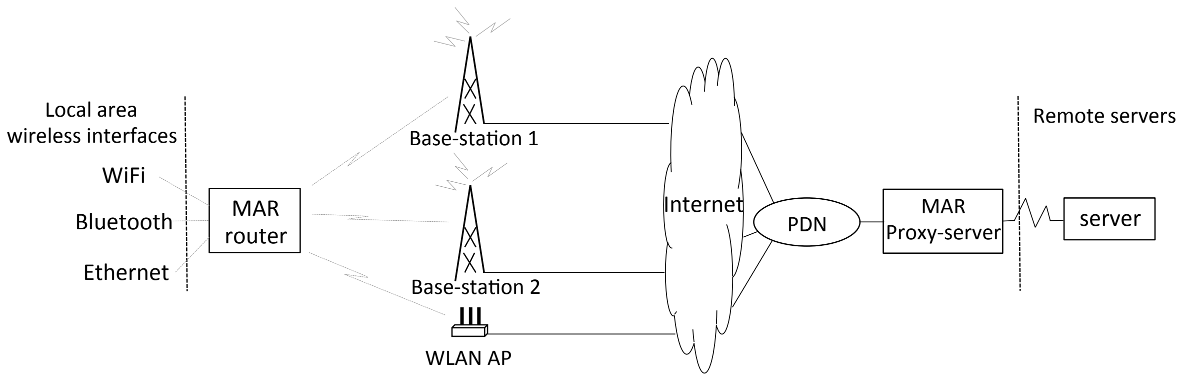

Figure 5. MAR [166] system architecture where the MAR router is placed in public mobile vehicles and data traffic is sent from remote servers to local devices. PDN: Public Data Network, AP: Access Point.

provides an API that can accommodate any custom purpose-built scheduling protocol. But the scheduling protocol itself is not part of the MAR architecture. MAR is also designed to determine the weight that should be assigned to each interface to properly perform load balancing (e.g. dynamically shifts load from poor quality to better quality channels). Note that the MAR router was supposed to be placed in moving vehicles, where users can use their devices for web-browsing and audio/video streaming. Therefore, the traffic load-balanced is only in one direction (i.e., from remote servers to local devices).

Manousakis et al. [122] proposed INTELiCON to allow devices to exploit wireless access diversity. At the sending side, a Packet Processing module manipulates the content of packets, e.g., modifying IP headers and Piggy-Backing for Control Signaling (PBCS) (e.g., timestamp and customized sequence numbers). At the receiving side, the piggy-backed information can be utilized to smooth out the arrival sequence of incoming packets. The extra information is stripped out at the Packet Processing module before the packets are forwarded to the upper layer. Moreover, INTELiCON uses a DATA/ACK separation (SEP) scheme to reduce contention on shared media. For example, it transmits the DATA and ACK packets on different paths.

Evensen et al. [52] proposed a Multilink Proxy (MLP) that makes use of a NAT proxy to rewrite the default destination IP address and port to the address of the other additional interfaces. The client does the inverse address translation of the packets arriving at non-default interfaces and forwards the packets internally. In order to mitigate packet reordering, Evensen et al. uses a WRR based scheduler in the NAT to distribute packets according to estimated throughput ratio of available paths.

Habak et al. [68] proposed OSCAR architecture that works in a distributed environment. An OSCAR-enabled node can share and use the bandwidth available from its OSCAR-enabled neighbors to connect to both legacy and OSCAR-enabled servers. OSCAR has a NAT module at both sides of a connection. At the sender, the NAT module replaces the source and destination IP addresses with the used IPs for transmission. Upon receiving a packet, the NAT module at the receiver reverses the source and destination IPs by replacing them with the negotiated ones before delivering the packet to TCP. When a connection goes through a shared neighbor to a legacy server, the neighbor also needs to have a NAT module that conducts the address translation operation. OSCAR uses a Packet Reordering Module (PRM) to handle packet reordering issues. Specifically, it delays the packets and their ACKs on both sides respectively before forwarding them to the upper layer. OSCAR has two scheduling modes: POSM and FOSM. FOSM is used if the server is a legacy server, where PFA is used. POSM is used if the server is OSCAR-enabled such that a WRR style scheduler is used.

Some of the IP-in-IP encapsulation and NAT based approaches, e.g., [26, 27, 52, 104, 105, 109, 166], assume the presence of a proxy infrastructure in the network. Nevertheless, such approaches work only for plain-text TCP communication and fail in the presence of IPsec encryption or authentication mechanisms. When TCP packets are protected with IPsec, the proxy is not able to observe or modify the packet headers. Next we discuss certain bandwidth aggregation approaches that use IPSec encapsulation for tunneling.

3) Identity/locator Split: Host Identity Protocol (HIP) [131][3] and Site Multihoming by IPv6 Intermediation (shim6) [137] have been proposed and implemented to provide multihoming support for failover with the possibility of flow-based load balancing. shim6 is architecturally related to HIP in that they both introduce an additional addressing layer to allow changing IP addresses on network interfaces, while keeping constant transport-layer identifiers. These two protocols enable IP packet flows to dynamically change paths in the presence of link failure. Thus, they naturally shield the presence of multiple paths from transport and application layers, presenting only the global identity of the peer host. Nevertheless, HIP and shim6 do not support simultaneous multipath transmission without additional extensions.

SIMA [150] is an extension of HIP to use multihoming for assigning separate TCP connections independently to different paths. Like FatVAP [95], TRILL [51], SPB [57], MAR [166] and OSCAR [68], SIMA also uses PFA multipath forwarding

[3]HIP may not be considered as a strict IP layer approach; however, its functions related to multipath transmission are best suited to this layer.

strategy where flow bonding rules are created to define the usage of the local interfaces. SIMA does not define any additional sending or receiving policies to mitigate reordering issue, instead it uses the IPsec Encapsulating Security Payload (ESP) packet processing unit built in HIP to handle each data packet, as specified in [93]. Gurtov et al. [65] designed and implemented Multipath HIP (mHIP), a multipath scheduler based on HIP, to distribute traffic over multiple available paths. Utilizing a EDPF scheduling algorithm, they striped packets within a TCP connection to multiple paths to mitigate packet reordering. Nevertheless, they found that EDPF algorithm is only effective against packet reordering with stable paths in terms of bandwidth and delay. In order to react to dynamic path characteristics, a Marking Technique is used as a part of the multipath congestion avoidance scheme, so that changes of path characteristics can be detected in one Round-Trip Time (RTT).

Polishchuk and Gurtov [151] proposed a TCP-friendly congestion control algorithm for mHIP to prevent stealing bandwidth from legacy TCP flows at the shared bottleneck. Specifically, they proposed a two-level congestion control scheme (removing the Marking Technique): per-path congestion control, and global congestion control on top of it. The global congestion controller coordinates the individual per-path controllers and balances traffic load among the paths based on their available capacity. The per-path controllers are connected so that the aggregated congestion window is the sum of per-flow congestion windows. The goal of this twofold congestion control scheme is to automatically redirect traffic from congested paths to the ones that have available capacity. The concept of joint congestion control algorithm adopted in [151] is also used by certain transport layer approaches (which will be discussed in the next section). Thus, the concern is that the reordering and congestion avoidance algorithms used on the IP layer (or between IP and TCP, like HIP) may need to repeatedly design additional mechanisms that are already existing on the transport layer.

When ESP is used with HIP, a 64-bit sequence number must be used. Therefore, HIP based bandwidth aggregation approaches such as SIMA [150] and mHIP [65, 151] all have a double sequence space design. However, instead of being used for packet reordering, the additional sequence number in HIP is used for the purpose of anti-replay.

Unlike HIP and shim6 which focus on host-level identity and locator separation, Locator/Identifier Separation Protocol (LISP) [56] is an identity and locator separation protocol working on the network-level to improve the scalability of the routing system. LISP creates two numbering spaces and uses two IP addresses: Endpoint Identifiers (EIDs) (assigned to end-hosts) and Routing Locators (RLOCs) (typically assigned to border routers). To achieve the separation of identification and localization, LISP follows a map-and-encapsulate scheme. Specifically, upon reception of a packet from the local network to an outer EID, the border router is responsible for looking up and retrieving the mapping (from a mapping system) between EID and RLOC and this process is invisible to the endpoints. Then the router encapsulates the packet with a LISP header and an outer IP header with the destination RLOC as the destination IP address. When the packet reaches the border router assigned with the destination RLOC, the router decapsulates the outer headers and forwards the inner packet to the destination EID. LISP has two metrics to support multipath transmission: RLOC priority and RLOC weight. If equal priority is sent on the RLOCs, the RLOC weight could be used for the load-balancing ratio. Under such a setting, an IP-level aggregate flow (e.g., the same destination prefix) would use different paths.

Locator/Identifier Separation Protocol - Hybrid Access (LISP-HA) [127] is a mechanism to provide simultaneous hybrid access (e.g., DSL-line and LTE) based on LISP technology in both upstream and downstream direction. LISP by itself has basic capabilities to support hybrid access with static load balancing. However, static load balancing may lead to statistical variations [128] so that some paths are already overloaded while others are underutilized. Instead, LISP-HA can perform dynamic per-flow load-balancing, which increases the efficiency of hybrid access. The basic idea is to obtain feedback about path-specific packet loss and delay, and leverage this information for improved load balancing. In addition, LISP-HA also supports dynamic per-packet load-balancing. Currently, the challenge is the packet reordering problem in the case that paths have different delay.

As summarized in Table VI, packet reordering is one of the main challenges for all IP layer approaches. These approaches use various scheduling algorithms to minimize the reordering effect. In Table VII, we have several observations. First, most of the approaches were proposed either for mobile networks or wireless networks. Second, there are two primary scheduling schemes: EDPF and WRR. Third, some buffer management strategies are used to compensate for the inefficiency of the scheduling algorithm in the scenario of dynamically changing networks.

C. Transport Layer Multipath Transmission

Compared with IP layer based approaches, transport layer approaches have certain inherent benefits because congestion control can be used as a mechanism for resource allocation in a network. At this layer, end-systems can easily obtain information about each path: capability, latency, loss rate and congestion state. This information can then be used to react to congestion in the network by moving traffic away from the congested paths. Current connection-oriented transport protocols, e.g., TCP, Stream Control Transmission Protocol (SCTP) [187], and Datagram Congestion Control Protocol (DCCP), transmit data only over a single path between a source and a destination at any given time. Numerous attempts have been made to tune these existing transport protocols for multipath capability. Currently, Concurrent Multipath Transfer (CMT) for TCP and SCTP are in the process of IETF standardization.

Like bandwidth aggregation on network layer, concurrent multipath transmission at the transport layer introduces an increase in the occurrence of packet reordering due to different path characteristics, including run-time throughput, RTT, loss, and error. Specifically, if a connection is striped over multiple network paths, the overall throughput may potentially be even worse than the throughput available on any one of the paths [58, 140]. There are two main causes of it. The first comes from the impact of heterogeneous RTT. For example, TCP expects a first-in-first-out delivery of packets through the network. Packet reordering at the receiver results in the reception of duplicate ACKs at the sender. The sender will fast retransmit the "missing" packet that may still be on its way over a high RTT path. Due to the misunderstanding of packet reordering, the overall throughput may degrade significantly. The second cause is the receive buffer blocking due to path heterogeneity or path failing. We provide more detail about the receive buffer blocking problem in later discussion on CMT-SCTP.

In this section, we classify the state-of-the-art according to their base protocols, in the order of SCTP and TCP. In each discussion of SCTP and TCP based approaches, we further divide them into two categories: with and without considering fairness. In Table VIII, we make a comparison of the general problems addressed by approaches in each category. In addition to the fairness issue, we also analyze the buffer impact, Pareto-efficiency, and path diversity of the TCP based approach MPTCP. In the end, we give a comparison between two representative approaches based on SCTP and TCP (i.e., CMT-SCTP and MPTCP).

Table IX and Table X are used to summarize the key algorithms and approaches of SCTP based multipath transmission respectively. Likewise, Table XI and Table XII are used to summarize the key algorithms and approaches of TCP based multipath transmission respectively.

1) SCTP based on Multipath Transmission: A standard SCTP is a general-purpose, connection-oriented unicast transport protocol. An SCTP connection denotes an association. The user data is segmented into units of so called DATA chunks[4], which are identified by unique Transmission Sequence Numbers (TSNs)[5]. The Selective Acknowledgment (SACK) [125] mechanism is used as default to acknowledge received data chunks and report gaps (i.e., missing data chunks indicated by their TSNs) to the sender. In this section, we divide the approaches into two groups differentiated by whether they have considered fairness.

Multipath Transmission without Considering Fairness:

Unlike TCP, SCTP was designed with multihoming in mind that an SCTP association allows multi-homed source and destination endpoints. Nevertheless, SCTP uses only one primary path and switches to another path for retransmission of lost packets, or as a backup in the case of failure from the primary path. Note that SCTP uses a single buffer structure on both endpoints, but maintains several states per destination: separate congestion window (cwnd), slow start threshold (ssthresh), retransmission timer, and RTT estimate. Several SCTP extensions, such as BA-SCTP [13], W-SCTP [24], CMT-SCTP [11, 89, 90, 91, 120], LS-SCTP [5, 6], WiMP-SCTP [85], cmpSCTP [118], mSCTP-CMT [21] and FPS-SCTP[129], enable SCTP to transmit data over multiple paths simultaneously.

Argyriou et al. [13] proposed BA-SCTP, a bandwidth aggregation protocol based on SCTP. BA-SCTP implements a mechanism for identifying bottlenecks that are shared by flows from the same aggregate connection. Based on this mechanism, BA-SCTP performs a unified congestion control algorithm for the flows that share the same bottleneck (we name this algorithm UCCSB) instead of applying congestion control for each flow separately. This design guarantees that BA-SCTP flows are fair with the other TCP flows sharing the same bottleneck. BA-SCTP employs a WRR style scheduling strategy, a congestion window based data allocation strategy where each subflow pulls data from the shared sending buffer whenever the subflow has congestion window space to send data. This strategy assumes the congestion window to be a true representative of the bandwidth-delay product of the subflow compared to an estimated product. In the rest of the paper, we use WRR-PULL to denote congestion window based data allocation strategy.

Casetti et al. [24] proposed W-SCTP, a Westwood [124] flavored SCTP to exploit bandwidth aggregation. The authors believed that Westwood style congestion control could fully exploit the advantages of bandwidth estimation which could be utilized for traffic allocation among multiple flows. W-SCTP uses a EDPF style scheduler that chooses the path for next packet by predicting whether it can deliver the packet the fastest to the destination.

Iyengar et al. [89, 91] proposed integrating CMT capability into SCTP, namely CMT-SCTP. CMT-SCTP utilizes the multihoming feature from SCTP to correctly transfer data between multi-homed end hosts. They identified three negative side-effects of CMT, and proposed algorithms to solve them accordingly. First, they proposed a Split Fast Retransmit Changeover Aware Congestion Control (SFR-CACC) algorithm to eliminate the unnecessary fast retransmissions by using a different interpretation of SACK information. Second, they used a congestion window (cwnd) growth algorithm to track the earliest outstanding TSN per destination and update the cwnd, even in the absence of new cum ACKs. Third, they proposed a new Delayed ACK algorithm for CMT-SCTP, namely Delayed ACK for CMT (DAC). The algorithm allows the receiver to delay sending ACK of an out-of-order segment. In [90, 91], they proposed five retransmission policies for CMT. They demonstrated the occurrence of spurious retransmissions with all of those policies, and proposed amendment algorithms to avoid them.

There has been a considerable amount of work on the core CMT-SCTP [91] to overcome its defect and improve performance. In [120], Liu et al. found that all of the five retransmission policies may cause throughput degradation due to receive buffer blocking. This blocking problem is also named Head-of-Line Blocking (HLB). Figure 6 illustrates a simplified example of it. As shown in the figure, a CMT-SCTP receiver maintains a single receive buffer which is shared across two sub-association flows in an association. The C_1 (chunk 1) is transmitted through the path 2 and is lost due to traffic congestion or path failure. During the time period of

[4]Corresponding to segments in TCP.

[5]TSN serves the same function in SCTP as the sequence number does in TCP.

Table VIII
COMPARISON OF PROBLEMS ADDRESSED BY SCTP AND TCP BASED MULTIPATH TRANSMISSION APPROACHES.

	SCTP based approaches	**TCP based approaches**
No fairness	• Maximize throughput • Spurious retransmission • Head-of-line Blocking (HLB)	• Maximize throughput • Spurious retransmission • Head-of-line Blocking (HLB)
Fairness	• SCTP-friendly at the same bottleneck • Resource pooling (RP)	• Avoid establishing multiple subflows at the same bottleneck • Maximize throughput at different bottlenecks • TCP-friendly at a shared bottleneck • Resource pooling (RP) • Incast collapse • Quality-of-service for multimedia applications • Trade-off between responsiveness and friendliness

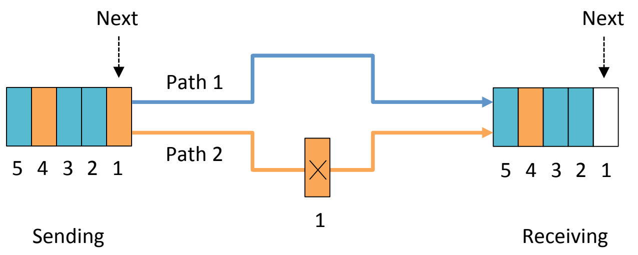

Figure 6. HLB: the receive buffer cannot accommodate other chunks any more before the arrival of the head-of-line chunk (chunk 1). HLB: Head-of-Line Blocking.

C_1's retransmission, the receive buffer cannot accommodate any other packets due to flow control so that the overall throughput degrades. To solve the problem, Liu et al. proposed a compound parameter retransmission policy named RTX-LCS. It limits the retransmission path selection by considering three common conditions: cwnd, ssthresh, and loss rate. Dreibholz and Adhari et al. [8, 49, 50] examined the challenges of CMT-SCTP over dissimilar paths. They identified the issues of sender and receiver queue blocking, which may lead to poor overall performance. In order to improve performance, Dreibholz [49] proposed multiple mechanisms accordingly, including Buffer Splitting, Chunk Rescheduling, and Smart Fast Retransmission. Buffer Splitting is used to avoid one path occupying too much buffer space, which prevents other paths from sending out new chunks. Chunk Rescheduling copes with the problem of certain delayed or lost chunks stalling the whole transmission. Smart Fast Retransmission deals with spurious fast retransmission bursts. For example, it does not consider chunks being moved from another path in the decision about fast retransmissions on the new path. In [8], they presented an optimized buffer handling technique to further improve performance. Specifically, they proposed to use the shared buffer space dynamically so that a faster path can have the possibility to send more data by granting it more buffer space. In this paper, we use Buffer Splittingv2 to denote this updated version of Buffer Splitting mechanism. Moreover, in [50], they proposed using the Multi-streaming feature of SCTP to mitigate the HLB problem. Specifically, each message is assigned an identifier to indicate a stream. With this identifier, the protocol only needs to restore the sequence of messages belonging together. Hence, after a packet loss, only the messages of the affected streams have to be delayed to restore the sequence. The other messages can be processed immediately without delay.

The authors in LS-SCTP [5, 6] and cmpSCTP [118] proposed separating the association flow control from per path congestion control (denoted as FCCS). The congestion control is performed per path, whereas the flow control is performed per association. In order to achieve this goal, LS-SCTP uses two different sequence numbers. The first one is the Association Sequence Number (ASN) that is used to reorder the received data at the receiver association buffer. The second one is the Path Sequence Number (PSN) that is used for reliability and congestion control on each path. The scheduling module of LS-SCTP uses the current congestion window (cwnd) of each path as an estimate of its current bandwidth-delay product. For example, it assigns data to the paths according to the cwnd/RTT of each path. cmpSCTP distributes data over available paths based on real-time bandwidth estimation of each path. Similar to LS-SCTP, cmpSCTP also distributes the data on the available paths based on the estimation of the available bandwidth of each path. Thus, they all use a WRR style data scheduler.

Sarkar [172] proposed Concurrent Multipath TCP (cmpTCP), an extension of SCTP. cmpTCP splits packets concurrently over all available paths from a shared sending buffer. cmpTCP maintains a virtual retransmission queue (RTxQ) on each path to control the number of outstanding bytes on the path. The receiver sends back ACKs on the same path on which the packets are received. These two designs may help to ignore spurious gap reports and eliminate unnecessary packet retransmissions. Sarkar also developed a Markov model in cmpTCP to estimate the data transport rate on each path when the transmission has reached a steady state. cmpTCP uses a WRR style scheduler by considering the number of outstanding bytes and congestion window size.

Huang et al. [85] proposed Wireless Multi-Path SCTP (WiMP-SCTP), which devised two data transmission modes, i.e., Data-striping Mode and Data-duplicating Mode, for multi-path transmission in multiple wireless access networks. When the network status is good, the Data-striping Mode is selected

Table IX
KEY ALGORITHMS FOR SCTP BASED CONCURRENT MULTIPATH TRANSMISSION (sorted according to their order mentioned in Table X).

Algorithm	**Problems to address**	**Description**
WRR-PULL	Scheduling	It is a congestion window based data allocation without estimation of each path's available bandwidth. Data packets are stored in a shared sending buffer and are pulled by sub-flows when the sub-flows have congestion windows space to transmit data.
UCCSB (Unified Congestion Control for flows sharing the Same Bottleneck)	Fairness	It first identifies the bottlenecks that are shared by flows from the same connection, and then performs an unified congestion control for those flows so that they compete bandwidth of the bottleneck fairly with other TCP flows.
SFR-CACC (Split Fast Retransmit Changeover Aware Congestion Control)	Spurious retransmission	It introduces a virtual queue per destination within the sender's retransmission queue. The sender uses SACK along with history information in the retransmission queue to deduce missing reports for a segment by inferring cumulative ACK and gap reports per destination.
Cwnd Updates	Cwnd slow growth	It first tracks the earliest outstanding TSN per destination and then uses SACKs and history information to deduce missing reports for a segment by inferring cumulative ACK and gap reports per destination. Therefore, the algorithm can update the cwnd even in the absence of new cumulative ACKs.
DAC (Delayed ACK for CMT)	Spurious retransmission	It delays sending an ACK if an out-of-order segment arrives at the receiver.
RTX-LCS (Lossrate, Congestion window and Slow start threshold)	Spurious retransmission	It prioritizes the retransmission through the subflow with the largest cwnd. If subflows have the same cwnd, the retransmission is made through the subflow having the largest ssthresh. If subflows have the same ssthresh, the retransmission is sent through the subflow with the lowest loss rate. Otherwise, a subflow is selected randomly.
FCCS (Flow and Congestion Control Separation)	Load sharing	It separates the association (or connection) flow control from congestion control. The flow control is on association basis. Both endpoints use their association buffer to hold the data chunks from all paths. Congestion control is performed on per path basis. Thus, the sender has a separate congestion control for each path.
CMT-PF (CMT Potentially Failed scheme)	Packet reordering	It marks a path that experiences a single timeout as a "potentially failed" path so that no further data transmission is allowed on that path. To detect its status, the sender sends heartbeat packets to the receiver. If the sender successfully gets the heartbeat ACKs, it will reactive the path for data transmission again.
Buffer Splitting	Packet reordering	It splits the shared sender buffer of size into *n* (i.e. number of paths) fixed per-path sections. A new chunk on a path could only be sent if its own buffer share has available space.
Chunk Rescheduling	Packet reordering	For each retransmission, it searches the first chunk which blocks the removal of chunks on the path from the sender buffer. That chunk is rescheduled on the path immediately when the congestion window has available space. Chunk Rescheduling is triggered when the path blocks more than half of the path's buffer share.
Smart Fast Retransmission	Spurious retransmission	It does not consider chunks that are moved from a path in the decision about fast retransmissions on a new path.
Multi-streaming	Packet reordering	It assigns each message with an identifier to indicate a stream. Each stream is sent over a certain path. It only needs to restore the sequence of streams belonging together. Hence, after a packet loss only messages of the affected streams have to be delayed to restore the sequence.
CMT/RP (CMT/Resource Pooling)	RP	It takes the interaction of the congestion controls on different subflows into account instead of handling them independently. One of its key operations, for example, is that it incorporates the possibility of shared bottlenecks by trying to halve the overall congestion window on the lossy path. CMT/RP assumes that paths have similar characteristics.
FPS (Forward Prediction Scheduling)	Packet reordering	It takes account of the transmission delay of each path and schedules the specific data unit accordingly on each path so that the data could arrive at the receiver in order.
Buffer Splittingv2	Packet reordering	It splits the shared sender buffer space dynamically. For instance, it grants more buffer space to faster paths so that they could have opportunity to send more data.
CMT/RPv2 (CMT/Resource Pooling Version 2)	RP	It is an updated version of CMT/RP-SCTP to overcome its limitations by considering different path characteristics.
BERP (Bandwidth Estimation Based Resource Pooling)	RP	It applies the RP principle based upon the bandwidth estimates obtained by observing the data flow on the paths.

Table X
CONCURRENT MULTIPATH TRANSFER PROTOCOLS BASED ON SCTP.

Scheme	**Year**	**Algorithm and Protocol**	**Paths**	**Receive Buffer**	**RP & Fairness**	**Sequence Space**	**Network Environment**
BA-SCTP [13]	2003	WRR-PULL, UCCSB, SACK	General	Constrained	Fairness	Single	General
W-SCTP [24]	2004	EDPF, SACK, Westwood	Disjoint	Not specified	No	Single	General
CMT-SCTP [89]	2004	SFR-CACC, Cwnd Updates, DAC, SACK	Independent	Infinite	No	Single	General
CMT-SCTP [90]	2004	Retransmission policies, SACK	Independent	Infinite	No	Single	General
LS-SCTP [5, 6]	2004	FCCS, WRR, SACK	General	Constrained	No	Double	
CMT-SCTP [91]	2006	SFR-CACC, Cwnd Updates, DAC, Retransmission policies, SACK	Independent	Constrained	No	Single	General
cmpTCP [172]	2006	WRR, SACK	Independent	Infinite	No	Single	General
WiMP-SCTP [85]	2007	WRR, SACK	Independent	Not specified	No	Single	Wireless
CMT-SCTP [120]	2008	RTX-LCS, SACK	General	Constrained	No	Single	General
cmpSCTP [118]	2008	FCCS, SACK, WRR	General	Constrained	No	Double	General
mSCTP-CMT [21]	2009	SACK, CMT-PF	Disjoint	Constrained	No	Single	Wireless
CMT-SCTP [49]	2010	Buffer Splitting, Chunk Rescheduling, Smart Fast Retransmission, SACK	General	Constrained	No	Single	General
CMT-SCTP [50]	2010	Multi-streaming, SACK	General	Constrained	No	Single	General
CMT-SCTP [48]	2010	CMT/RP, SACK	Similar paths	Not specified	Yes	Single	General
FPS-SCTP [129]	2010	FPS, SACK	Disjoint	Constrained	No	Single	Mobile
CMT-SCTP [8]	2011	Buffer Splittingv2, SACK	General	Constrained	No	Single	General
CMT-SCTP [47]	2011	CMT/RPv2, SACK	General	Constrained	Yes	Single	General
CMT-SCTP [175]	2011	BERP, SACK	General	Not specified	Yes	Single	Wireless

to aggregate bandwidth. On the other hand, when the network status becomes bad, the Data-duplicating Mode is selected to increase destination reachability. To switch between the two modes, a mode selection scheme that determines the status of these multiple paths was proposed. Specifically, it designed a HEARTBEAT scheme where heartbeat chunks are sent periodically on the paths. The transmission error counter increases when a heartbeat is not acknowledged within one retransmission timeout interval. If the number of transmission error counter plus the number of consecutive retransmissions exceeds a certain threshold, the Data-duplicating Mode is switched on. Otherwise, the Data-striping mode is used. Nevertheless, Huang et al. did not present a complete explanation of the scheduling algorithm used by WiMP-SCTP. They only mentioned that the sender transmits data as soon as the corresponding receive window at the receiver side allows data to be sent. We speculate that this is a variant of the WRR style scheduling algorithm.

Budzisz et al. [21] proposed an mSCTP-CMT protocol to investigate the applicability of using CMT-SCTP to distribute data between two paths during the handover transition process. They emphasized the consequence of a sender-introduced reordering and its effect on congestion control. The authors found that in CMT-SCTP the receive buffer may be filled with out-of-order data caused by complete or short-term failures during handover. Although handover is out of the scope of this paper, the handover scenario is very similar to the worst case in CMT where a path experiences a long delay suddenly. To solve this problem, they proposed using Potentially Failed (CMT-PF) scheme that a path experiencing a single timeout is marked as "potentially failed" and no further data transmission is allowed on that path. They utilized a heartbeat scheme to probe whether the potentially-failed path has got back to a positive state in the case of successful heartbeat acknowledgement.

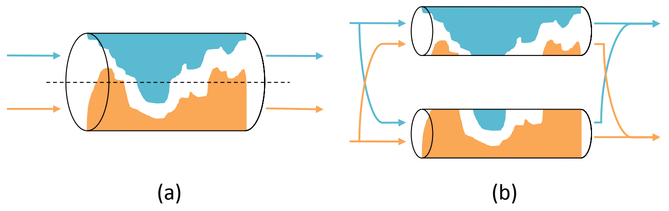

Figure 7. RP (a) a single path shares its resource fairly to competing flows (b) multiple paths are treated as a single pooled resource. RP: Resource Pooling.

Mirani et al. [129] proposed a multipath Forward Prediction Scheduling (FPS) for SCTP, namely FPS-SCTP. In order to reduce the number of out-of-order packets, it estimates the arrival time of each packet in advance and decides which packet is to be sent through a certain path so that the packets can arrive at the destination in order. They used roughly a half of the RTT on a subflow to estimate the one trip time from data leaving the send buffer to being received at the receiver. This prediction is an approximation considered as too coarse because previous studies have shown that a majority of Internet connections experience latency asymmetry [141, 197].

Multipath Transmission Considering Fairness:

The work discussed previously on CMT-SCTP performs independent congestion control on each path, and considers little about the fairness against other single-path flows. For example, in the case of n CMT-SCTP paths, the association will get n times the bandwidth share of a competing non-CMT SCTP or TCP flow over the same bottleneck.

The concept of RP [200] is a milestone for multipath transmission aggressiveness control. In the context of multipath transmission, it makes a collection of resources behave like a single resource by balancing traffic across multiple paths.

As shown in Figure 7, when several TCP flows compete for a single path, TCP can share the path's capacity fairly among them. When the flows go through more than one path, the paths are treated as a single pooled resource. With appropriate coordination, traffic can move away from more congested paths to less congested ones, and larger bursts can be accommodated.

Dreibholz et al. [48] proposed a RP congestion control for CMT-SCTP denoted as CMT/RP-SCTP by combining CMT-SCTP with the concept of RP. The goal of CMT/RP-SCTP is to improve the data throughput while still remaining fair to concurrent single-path flows on the shared bottleneck. For example, when two paths are used concurrently for data transmission and they share a bottleneck link, the overall throughput obtained by a CMT/RP-SCTP association should be similar as that of a standard SCTP. However, CMT/RP-SCTP assumes similar paths, i.e., paths having very similar characteristics in terms of bandwidth, delay and loss rate. Dreibholz et al. [47] proposed an updated version of CMT/RP-SCTP (denoted as CMT/RPv2-SCTP) to overcome the limitations of CMT/RP-SCTP by considering path bandwidths. They studied the behavior of CMT/RPv2-SCTP on dissimilar paths and found that CMT/RPv2-SCTP achieves the goals of RP. Furthermore, they observed that compared with MPTCP (which will be discussed later), CMT/RPv2-SCTP distributes bandwidth to flows equally when possible regardless of the number of paths used for transport.

Note that both CMT/RP-SCTP and CMT/RPv2-SCTP apply RP principle only during the congestion control phase. Shailendra et al. [175] argued that this strategy is not optimum on heterogeneous networks with wireless links because losses in wireless links may happen because of reasons other than congestion. Hence they considered the resources to be as a single pool of resources during the congestion detection phase as well. To achieve this goal, they proposed a Bandwidth Estimation Based Resource Pooling (BERP) algorithm which applies the RP principle based upon the bandwidth estimates obtained by observing the data flow on the paths.

2) TCP based Multipath Transmission: In contrast to SCTP, which was designed with multihoming support in nature, TCP is unaware of multiple interfaces and allows only a single IP address per endpoint. Nevertheless, TCP has dominated the Internet traffic and has sparked a lot of interests in enabling TCP to support simultaneous multipath transmission. In this section, we divide the approaches in four groups. The grouping principle is influenced by the research issues the approaches aim to address, such as fairness, buffer impact on performance, Pareto-optimality and path diversity. Although some approaches may cover more than one research issue, we only discuss them in the group which we believe the approaches are best fit into. In our study, we found that SCTP and MPTCP share many similar issues and certain algorithms. At the end of this section, we summarize their common features as well as their differences.

Multipath Transmission without Considering Fairness:

Magalhaes et al. [121] proposed Reliable Multiplexing Transport Protocol (R-MTP), which is a rate-based reliable transport protocol multiplexing data across multiple network interfaces (i.e., a WRR style scheduler). It relies on explicit bandwidth probing via the packet pair method [99] to estimate bandwidth in order to adjust the rate on the available paths accordingly. For example, it measures packet inter-arrival times and jitter to sense bandwidth scarcity. The probing period should occur on a fine time-scale to reflect the fluctuation of the available bandwidth.

Lee et al. supported two transmission modes in their work [111]: FOSM and POSM. In POSM, they investigated multiple schemes to address the spurious retransmissions by modifying two TCP operations: 1) Increasing the Fast Retransmit Threshold (IFRT) and 2) enabling Delayed ACKs for out-of-order packets as well as sending immediately ACKs for retransmitted packets. The second modified operation is like an advanced version of DAC used in [89, 91]. Thus, we name it DACv2 in this paper. IFRT makes the TCP sender wait for more than triple duplicate ACKs, which reduces the number of the fast retransmission and the fast recovery events. DACv2 enhances performance because when ACKs are being delayed, new packets may fill the gap and change the out of order packets in order.

Parallel TCP (pTCP) [82, 83, 84] functions as a wrapper around a modified version of TCP. It opens multiple TCP flows, one for each interface in use. pTCP performs data-striping across multiple micro-flows (TCP flows) by considering their bandwidth difference. Specifically, pTCP uses cwnd/RTT ratio, a WRR-PULL scheduler, to allocate traffic proportionally to path capacity. In addition, pTCP has several other strategies addressing specific problems. For example, the congestion window could be an over-estimate especially just before congestion occurs. This can result in an undesirable hold up of data in subflows. Instead of reassigning data to other subflows later on, pTCP uses a Delayed Binding strategy to adapt to instantaneous changes in path capacity. Specifically, it pulls data from the shared sending buffer only when the data is scheduled to send out immediately through a subflow. In order to avoid an overflow of the receive buffer, pTCP uses a Packet Re-striping strategy to retransmit a packet through a different subflow instead of the subflow which transmitted that packet earlier, and uses a Redundant Striping strategy to send a duplicated packet on one subflow to another. Moreover, pTCP uses SACK feedback mechanism to recover a lost packet in a much shorter time period.

Reception Control Protocol (RCP) [81, 106] is a receiver-centric transport protocol with a minimized sender design. The receiver controls all the key functions in RCP. To support CMT, a multi-state extension of RCP, i.e., Radial RCP (R^2CP), was proposed. R^2CP maintains one RCP pipe (the same as a TCP flow) per end-to-end path with congestion control being handled by individual RCP pipes. Traffic is scheduled to each RCP pipe based on the (estimated) time the requested segment will arrive through the concerned pipe (a variant of EDPF). Note that each RCP pipe maintains a local sequence number space internally to facilitate loss detection and recovery. The local sequence number can be converted to the global sequence number, and vice versa.

Chen et al. [28] proposed M-TCP that uses a Duplicate

Table XI
KEY ALGORITHMS FOR TCP BASED CONCURRENT MULTIPATH TRANSMISSION (sorted according to their order mentioned in Table XII).

Algorithm	Problems to address	Description
IFRT (Increase Fast Retransmission Threshold)	Spurious retransmission	It makes TCP senders wait for more than triple duplicate ACKs so as to reduce the number of the fast retransmission and the fast recovery events.
DACv2 (DAC version 2)	Spurious retransmission	It is an updated version of DAC. In addition to delaying sending ACKs for out-of-order segments, it allows the receiver to send immediately ACKs for retransmissions.
Delayed Binding	Path Capacity Fluctuation	When a packet is transmitted by a subflow, it binds a virtual packet to the real packet in the shared sending buffer. This binding is performed only just before that packet is transmitted.
Packet Re-striping	Reordering	In the case of path capacity fluctuations, it re-sends a packet that was earlier transmitted through a path through another one that has space in its congestion window.
Redundant Striping	Reordering	Every time a timeout is experienced, it redundantly re-sends the packet inside the congestion window through another path that has space in its congestion window.
OMS (Opportunistic Multipath Scheduler)	Scheduling	It opportunistically favors low-delay high-throughput paths while simultaneously ensuring that the traffic splitting ratios defined by the routing policy are satisfied.
Shared Congestion Detection	Fairness	It dynamically detects and suppresses paths with shared congestion. For example, the sender detects shared congestion by examining the correlations between the fast retransmit times of the subflows.
Duplicate Transmission	High packet loss	It duplicates each packet and transmits a copy over each of the multiple paths provided by routing.
Duplicated ACK	Data acknowledgement	It sends an ACK at the receiver immediately through more than one path upon the receipt of a data segment.
Duplicated & Delayed ACK	Data acknowledgement	It sends an ACK at the receiver for every other data segment through more than one path.
RCC (Rate-based Congestion Control)	Congestion avoidance	It estimates the queue length at the bottleneck link. If the queue length grows beyond a predefined threshold, the sender will recalculate a new congestion window to avoid packet loss due to traffic congestion.
Duplicated ACK classifier	Spurious retransmission	It handles packet reordering by differentiating whether a duplicated ACK is likely a real duplicated ACK, or is caused by CMT and should be ignored.
Selective Offloading	Wireless signal contention	It diverts ACKs to the 3G channel to prevent them from contending with the data transmission in the WiFi channel, even if the 3G interface may have a much smaller amount of bandwidth.
WCC (Weighted Congestion Control)	Fairness, RP	It makes a bundle of subflows in a multipath connection fairly compete with normal TCP flows at the shared bottleneck, and also fairly allocates bandwidth among different connections across the distinct bottlenecks.
LIA (Linked-Increases Algorithm)	Fairness, RP	It is a joint congestion control algorithm by allowing a bundle of subflows in a multipath connection fairly compete with normal TCP flows at the same bottleneck. Following the RP principles, LIA is able to move traffic from more congested paths to less congested ones.
DWC (Dynamic Window Coupling)	Fairness, performance	It dynamically detects distinct bottlenecks. DWC only couples the subflows sharing a common bottleneck, while using separate congestion control for other subflows.
OLIA (Opportunistic Linked-Increases Algorithm)	Pareto-optimality	It ensures both MPTCP and competing TCP users could obtain enhanced performance, which satisfying the design goals of MPTCP.
Opportunistic retransmission	Packet reordering	If a subflow holds up the trailing edge of the receive window, it re-sends the packet which is previously sent on another subflow.
Penalizing slow subflows	Packet reordering	If a subflow holds up the advancement of the receive window, it halve its congestion window and sets the slow-start threshold to the reduced window size. To avoid repeatedly penalizing the same flow, only one reduction is applied per RTT.
Partial Reliability	Multimedia QoS	It is defined as the possibility for the communication system to not recover "acceptable" losses in order to improve QoS such as delay or bandwidth.
Balia (Balanced Linked Adaptation)	Balance among friendliness, responsiveness, and window oscillation	It explicitly balances the trade-off among friendliness, responsiveness, and window oscillation.
New Delayed ACK	Packet reordering	It is a new Delayed ACK mechanism designed for MPTCP. It removes the Minimum RTO constraint at the sender while reserving the Delayed ACK function at the receiver.
Penalizing slow subflows (Improved)	Packet reordering	It's an updated version of the Penalizing slow subflows mechanism. If a subflow is in its slow-start phase, it should not adjust the slow-start threshold.
CWA (Congestion Window Adaptation)	Path delay difference	If a large delay ratio (the ratio of the maximum path delay over the minimum path delay) is detected, it decreases the congestion window of the path with the maximum delay even if there is no packet loss indicated by duplicate ACKs.
AOLIA (Adapted OLIA)	Fairness	It is an adaptation of the original OLIA. AOLIA ensures controlled aggressiveness of the MPTCP subflows to improve the overall performance.
EDWC (Extension of DWC)	Fairness, performance	It utilizes a delay congestion event to trigger an alert and either delay or loss congestion event is used to add subflows to the coupling set.
PSPLH (Push-Pull-Hybrid)	Packet reordering	It is a hybrid scheduler used to efficiently operate packet scheduling by combining both push and pull style strategies. For example, the hybrid scheduler presented in [182] allocates data segments to active flows (Pull) with dynamic size (Push).
NR-SACKs (Non-Renegable SACKs)	Send buffer blocking	Once a data packet has been sacked, it does not remove it from the receive buffer. Thus, the sender can free the sacked data in order to alleviate the send buffer blocking.
DRePaS (Dynamic Relative Path Scoring)	Packet reordering	It dynamically scores the paths relative to the best path. If a path's score is lower than the predefined threshold, it suspends scheduling data over that path until its performance improves measured by probing traffic.

Table XII
CONCURRENT MULTIPATH TRANSFER PROTOCOLS BASED ON TCP.

Scheme	Year	Algorithm and Protocol	Paths	Receive Buffer	RP & Fairness	Sequence Space	Network Environment
R-MTP [121]	2001	SACK, WRR	Disjoint	Not specified	No	Single	Mobile
Lee et al. [111]	2002	IFRT, DACv2, PFA	General	Not specified	No	Single	General
pTCP [82, 83, 84]	2002, 2005	WRR-PULL, Delayed Binding, Packet Re-striping, Redundant Striping, SACK	Independent	Constrained	No	Double	Mobile
R^2CP [81, 106]	2003, 2005	EDPF, Packet Re-striping	General	Constrained	No	Double	Wireless
Cetinkaya et al. [25]	2004	OMS	Independent	Constrained	No	Single	General
mTCP [204]	2004	WRR, Shared Congestion Detection	Disjoint	Constrained	Yes	Single	General
M-TCP [28]	2004	Duplicate Transmission	Disjoint	Not specified	No	Single	Ad hoc
M/TCP [167]	2004	OWTT, WRR, Duplicate Transmission, Duplicated ACK, Duplicated & Delayed ACK	Independent	Not specified	No	Single	General
R-M/TCP [168]	2005	OWTT, WRR, Duplicate Transmission, RCC	Independent	Constrained	No	Single	General
cTCP [46]	2007	WRR, Duplicated ACK classifier	Independent	Constrained	No	Single	General
MPLOT [178, 179]	2008, 2012	Packet coding, ECN, EDPF	General	Not specified	No	Single	Lossy
JOSCH [198]	2009	WRR	Independent	Constrained	No	Single	Wireless
Super-aggregate [191]	2009	Selective Offloading, IP-in-IP encapsulation, Duplicate Transmission	Independent	Constrained	No	Single	Mobile
BMC [79]	2009	WCC	General	Not specified	Yes	Single	General
MPTCP [60, 158, 162]	2009	LIA, WRR	General	Constrained	Yes	Double	General
MPTCP [161]	2010	LIA, WRR	General	Constrained	Yes	Double	Data center
MPTCP [17]	2011	LIA, WRR-PULL	General	Constrained	Yes	Double	General
Hassayoun et al. [76]	2011	DWC, WRR-PULL	General	Constrained	Yes	Double	General
MPTCP [101, 102]	2012, 2013	OLIA, WRR-PULL	General	Constrained	Yes	Double	General
Han et al. [103]	2012	EDPF	Independent	Constrained	Yes	Double	General
NC-MPTCP [114]	2012	Packet coding, FPS	Independent	Constrained	Yes	Double	General
FMTCP[37, 38]	2012, 2014	Packet coding, FPS	Independent	Constrained	Yes	Double	General
MPTCP [160]	2012	Opportunistic retransmission, Penalizing slow subflows	General	Constrained	Yes	Double	General
QoS-MPTCP [44]	2012	Partial Reliability	Independent	Infinite	Yes	Double	General
Peng et al. [143]	2013	Balia	General	General	Yes	Double	General
MPTCP/OpenFlow [194]	2013	OpenFlow	Independent	General	Yes	Double	General
A-MPTCP [36]	2013	LISP	Independent	General	Yes	Double	Cloud
Coudron et al. [34]	2013	LISP, TRILL	Independent	General	Yes	Double	Cloud
MPTCP [116]	2013	New Delayed ACK, packet coding	Independent	Constrained	Yes	Double	General
MPTCP [140]	2013	Penalizing slow subflows (Improved)	General	Constrained	Yes	Double	General
CWA-MPTCP [206]	2013	CWA, FPS	General	Constrained	Yes	Double	General
Singh et al. [182]	2013	AOLIA, EDWC, PSPLH	General	Constrained	Yes	Double	General
Yang et al. [201]	2013	NR-SACKs	General	Constrained (Send buffer)	Yes	Double	General
Lim et al. [119]	2014	Detect MAC-Layer path status	General	General	Yes	Double	Wireless
Ferlin et al. [58]	2014	DRePaS	General	General	Yes	Double	Wireless
SC-MPTCP [115]	2014	Packet coding, FPS	Independent	Constrained	Yes	Double	General
EW-MPTCP [117]	2014	WCC	Independent	Constrained	Yes	Double	General
Yang and Amer [202]	2014	FPS	Independent	Infinite	Yes	Double	General
Le et al. [110]	2015	FPS	Independent	Infinite	Yes	Double	General
Coudron et al. [35]	2015	ΔOWD	General	Constrained	Yes	Double	General

Transmission mode for the lossy wireless environment with high interference. In this transmission mode, multiple copies of the same packet are sent on different paths so that the chance that all copies are lost is much reduced. Unfortunately, they only present sending-side modification without addressing duplicate ACKs due to multiple copies of the same packet.

Rojviboonchai and Hitoshi [167] proposed a multipath Transmission Control Protocol (M/TCP). M/TCP uses One-Way-Trip Time (OWTT) [169], a similar method as FPS, at the sender to estimate the delay time of the forward path and reverse path separately in order to calculate per path RTO timer. In addition, M/TCP employs two mechanisms to deal with packet loss. In the case of fast retransmission, M/TCP uses Duplicate Transmission policy, i.e., duplicating the missing segment and sending each copy through all paths so that a quick and reliable retransmission can be desirable; in the case of timeout retransmission, the missing segment on a flow is sent through the other flows. Moreover, M/TCP uses two algorithms for the receiver to transmit an ACK in the case of CMT. Namely, using Duplicated ACK algorithm, an

M/TCP receiver sends an ACK immediately upon the receipt of a data segment to more than one path; using Duplicated & Delayed ACK, the receiver transmits an ACK for every other data segments through more than one path. Rojviboonchai et al. proposed R-M/TCP [168] as an extension of their previous work M/TCP. R-M/TCP is a rate-based M/TCP that performs congestion control in a rate-based and loss avoidance manner (we use RCC to denote it) to avoid packet loss by adjusting the congestion window before buffer overflows. Specifically, R-M/TCP schedules data packets in a WRR manner while it estimates the queue length at the bottleneck link. If the queue length grows beyond a predefined threshold, the sender recalculates a new congestion window to achieve a fair share at the bottleneck.

Cetinkaya et al. [25] proposed an Opportunistic Multipath Scheduler (OMS) that follows a traffic splitting policy that favors low-delay high-throughput paths opportunistically for a short term variations in path quality. To avoid violating path weights of the routing protocol and potentially leading to oscillation, OMS ensures that over longer time scales traffic is split according to the ratios determined by the routing protocol.

Dong et al. [46] proposed concurrent TCP (cTCP) that uses a single congestion window to control the global throughput and a single sending buffer to be shared among all paths. It uses Credit-Weighted Round-Robin (a variant of WRR) as the scheduling algorithm. Each time an ACK comes back to the sender, the capacity estimation of that path is updated, and a new sending credit (similar to the congestion window size) is added to the sender. The new credit is for all the paths combined, and it is further divided into each path. cTCP uses the path credit (similar to the path capacity) to distribute data among the available paths. Furthermore, cTCP adopts a duplicated ACK classifier that handles packet reordering by differentiating whether a duplicated ACK is likely caused by CMT or a real duplicated ACK.

Sharma et al. [178, 179] proposed Multi-Path LOss-Tolerant protocol (MPLOT) to provide multipath transmission on multiple heterogeneous, highly lossy paths. MPLOT uses erasure based Forward Error Correction (FEC) packet coding. The major benefit of packet coding stems from its ability to compensate for missing packets from redundancy. This makes data transmission over lossy networks robust and efficient. To counter against packet reordering, MPLOT estimates path parameters (i.e., loss rate, capacity, and RTT) continuously to provide adaptive FEC coding. In particular, MPLOT performs latency-aware packet mapping, a variant of EDPF. For example, it maps packets that are not required immediately to paths with long delays, while mapping the more immediately useful packets to paths with short delays. MPLOT uses Explicit Congestion Notification (ECN) [164] to distinguish congestion losses from those due to faulty/lossy links.

Wang et al. [198] proposed a segment-based adaptive Joint Session Scheduling (JOSCH) mechanism. The main goal is to restrain the delay difference among multiple Radio Access Networks (RANs) by means of allocating the traffic to different RANs dynamically with reasonable ratios. Specifically, JOSCH obtains network conditions by a segment-based feedback approach, where a "segment" is defined as a predetermined size of data block. Its size is configurable according to the delay sensitivities of different services. After each segment transmission, the receiver sends feedback to the sender. According to the feedback, the sender adjusts traffic allocation dynamically according to the estimated transmission rates and delays.

Tsao and Sivakumar [191] argued that aggregated bandwidth of two wireless interfaces (3G and WiFi) is a Simple Aggregation due to path heterogeneity. For example, the low bandwidth interface (e.g., 3G with 100-500Kbps) can only achieve negligible bandwidth compared to the high bandwidth interface (e.g., WiFi with 2-54Mbps). They proposed a super-aggregation to achieve performance that is better than the sum of throughput achievable through each of the interfaces individually by the means of three mechanisms: Selective Offloading, Proxying, and Mirroring. In spite of the fact that an interface may have a relatively small amount of bandwidth, these mechanisms can provide considerable performance improvement. Specifically, the Selective Offloading mechanism is to receive TCP data segments over a comparably high-speed WiFi and return ACKs over a low-speed 3G path to address self-contention in WiFi networks. The Proxying mechanism, following IP-in-IP encapsulation, allows a 3G path to notify the TCP sender about blackout events on the WiFi path. The Mirroring mechanism (i.e., Duplicate Transmission) establishes an addition TCP connection through 3G to fetch the missing segments due to random loss on the WiFi path.

Multipath Transmission Considering Fairness:

Similar to the early development of SCTP based CMT, at the early stage, TCP based multipath transmission was only used for utilizing multiple TCP flows with intelligent scheduling algorithms to mitigate packet reordering. Nevertheless, it was found that simply utilizing multiple TCP flows concurrently at a bottleneck would result in a fairness issue, i.e., an unfair share of the bandwidth at a bottleneck link. For example, NewReno [77] is the most common TCP congestion control variant as it yields an equal share of the congested link. This equal share outcome of NewReno results in an unfair share of the bandwidth if more than one TCP flow is active for a single multipath transmission connection at the bottleneck link. From the literature review we have made, we find that multipath transmission approaches based on TCP in recent years have also started to make fairness a necessary feature.

To the best of our knowledge, mTCP [204] proposed by Zhang et al. is among the earliest proposals that have taken fairness issue into consideration for TCP based multipath transmission. To address the fairness issue, they proposed not establishing multiple flows through the same bottleneck. For example, to alleviate the aggressiveness problem, Zhang et al. integrated a shared congestion detection mechanism into mTCP so as to identify and suppress subflows that traverse the same set of congested links. For example, mTCP detects shared congestion by examining the correlations among the fast retransmit times of the subflows. mTCP also uses a scheduler in a WRR manner. For example, it maintains a counter, i.e., $pipe_i$, to represent the number of outstanding packets on the i_{th} path. $pipe_i$ is incremented by 1 when the

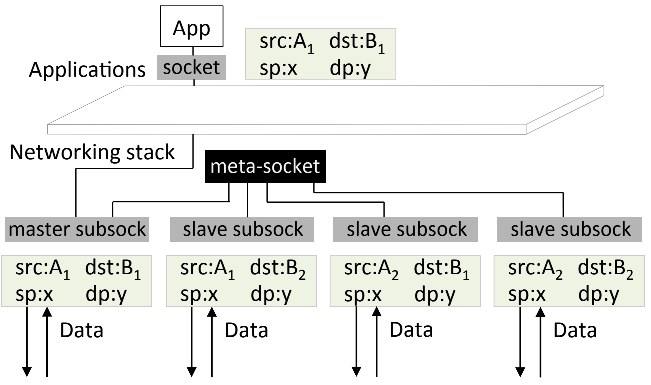
App
Applications
socket
src:A_1 dst:B_1 sp:x dp:y
Networking stack
meta-socket
master subsock
slave subsock
slave subsock
slave subsock
src:A_1 dst:B_1 sp:x dp:y
src:A_1 dst:B_2 sp:x dp:y
src:A_2 dst:B_1 sp:x dp:y
src:A_2 dst:B_2 sp:x dp:y
Data
Data
Data
Data

Figure 8. The architecture of MPTCP, which has the application compatibility by keeping the standard socket API to legacy applications. MPTCP: Multipath TCP, src: Source IP address, dst: Destination IP address, sp: Source Port, dp: Destination Port, APP: Application.

sender either sends or retransmits a packet over the i_{th} path, and is decremented by 1 when an incoming ACK indicates that a packet previously sent has been received. The sender associates a score, i.e., $pipe_i/cwnd_i$, for each path. The path with the minimum score has priority to send the next packet.

Instead of avoiding shared bottlenecks, Honda et al. [79] proposed Bidimensional-Probe Multipath Congestion Control (BMC) to address the fairness issue. Specifically, BMC uses a Weighted Congestion Control (WCC) approach that applies the weight to each subflow so that the throughput of each subflow is in proportion to its weight. In addition, WCC maintains the sum of the weight so that a bundle of subflows in the multipath connection is kept as aggressive as one TCP flow. WCC can achieve not only fair resource allocation at the shared bottleneck, but also RP [200] along the disjoint bottlenecks. For example, multiple different connections can obtain fair resource allocation across distinct bottlenecks.

Approximately in 2009, MPTCP [60, 158, 162] was proposed with the fairness property as well as RP feature in mind. Specifically, MPTCP, under discussion of IETF, has the following set of goals to achieve:

- Improve throughput: MPTCP should perform at least as a single TCP flow running on the best path.
- Do no harm: MPTCP subflows should not take more capacity than a single TCP flow would get at a shared bottleneck.
- Balance congestion: MPTCP should utilize the least congested path the most.

Figure 8 shows the architecture of MPTCP. It is a major extension to TCP and allows a pair of hosts to use several paths to exchange the segments that carry the data from a single connection. MPTCP presents a standard TCP socket API to the upper layer so that legacy applications can run upon MPTCP transparently. A Coupled Congestion Control (CCC) algorithm, Linked Increase Algorithm (LIA) [159], is used to guarantee fair resource allocation on multiple paths and provide RP feature among them. Its RP feature can shift traffic away from more congested paths to less congested ones. In addition to the joint congestion control algorithm, MPTCP also has a few other design features. For example, MPTCP adds connection-level sequence numbers in order to reassemble the data stream in-order from multiple subflows. A Data Sequence Signal (DSS) option [62] specifies a full mapping from the connection-level sequence number to the subflow sequence number. In the early stage of MPTCP, a PUSH style WRR scheduling strategy is used where the scheduler tries to fill all subflows when there is data coming from the application. Later, MPTCP adopts a WRR-PULL manner scheduler [17], a similar design adopted by pTCP [82, 83, 84] where the application data stored in a shared connection-level sending buffer is pulled by subflows whenever they have space in their congestion window. Both PUSH and WRR-PULL scheduling strategies are variants of WRR. Their difference lies in the fact that WRR-PULL strategy uses less time waiting in a subflow queue before its actual transmission on the wire. Although this time period seems minor, the path properties may change during that time. For the latest development of MPTCP in IETF, we refer readers to RFCs such as [61, 62, 159].

Raiciu et al. [161] were the first proposing a natural evolution of data center transport from TCP to MPTCP. They demonstrated that MPTCP could efficiently and seamlessly use available bandwidth, provide improved throughput and better fairness compared to single path TCP. The same authors further investigated what caused these benefits in [157]. They found that using MPTCP allows to rethink data center networks and approach them with a different mindset as to the relationship between transport protocols, routing, and topology. One of the challenges of deploying MPTCP in data centers is the Incast effect, a behavior of MPTCP as well as TCP that results in the gross under-utilization of link capacity in certain many-to-one traffic patterns [40, 132, 147, 190]. Incast collapse is not specific to MPTCP, but is inherited from TCP. Li et al. [117] investigated how to share network resources among different MPTCP flows by performing an additional weighted congestion control based on the coupled congestion control mechanism.

Although LIA ensures MPTCP subflows to be no more aggressive than a competing TCP flow, LIA is not able to differentiate whether the subflows share the same bottleneck or different ones. This may cause sub-optimal performance of MPTCP. Hassayoun et al. [76] proposed a Dynamic Window Coupling (DWC) algorithm to address it by detecting distinct bottlenecks and only coupling those that share a common bottleneck, while allowing other subflows to use separate congestion control. For example, DWC uses a loss congestion event to trigger an alert while using either the Delay or Loss congestion event to group subflows. Singh et al. [182] proposed an extension of the DWC algorithm, denoted as EDWC. This extension uses a delay congestion to trigger an alert while either delay or loss congestion is used to group and couple subflows. The concept behind DWC and EDWC algorithms is similar to that of UCCSB algorithm used in BA-SCTP [13] because they both identify the shared bottleneck and guarantee that the aggregated flows on the bottleneck is TCP-friendly.

Diop et al. [44] proposed QoS-oriented MPTCP that takes advantage of the two sub-layers architecture of MPTCP to use Quality of Service (QoS) techniques for multimedia applications over multiple paths. MPTCP originally offers a fully reliable and fully ordered service. Nevertheless, full reliability may not be required by certain multimedia applications. Diop et al. investigated the QoS benefits induced by the implementation of the Partial Reliability [12] feature in MPTCP for interactive video applications. Partial Reliability is an important concept for multimedia transmission over IP networks and is defined as the possibility to not recover losses under a threshold in order to improve QoS.

Peng et al. [142, 143] presented a fluid model to investigate a few existing congestion control algorithms designed for MPTCP, and identified design criteria that guarantees the existence, uniqueness, and stability of system equilibrium. They characterized algorithm parameters for TCP friendliness and proved that there is an inevitable trade-off between responsiveness and friendliness. Based on the study, in [143] they proposed a new congestion control algorithm, Balia (balanced linked adaptation). This algorithm generalizes existing algorithms and strikes a good balance among TCP-friendliness, responsiveness, and window oscillation.

Lim et al. proposed MPTCP-MA [119] to improve MPTCP performance during intermittent path connectivity in wireless environment. MPTCP-MA exploits MAC-Layer information to estimate path status, and suspends/releases a path based on the estimation. By quickly detecting path failure and recovery, MPTCP-MA can avoid unnecessary losses and utilize recovered paths more quickly.

Buffer Impact on MPTCP:

Although MPTCP was designed with several merits such as fairness, RP, and Pareto-optimality in mind, buffer size has significant impact on MPTCP performance. This problem stems from the packet reordering issue due to heterogeneous path characteristics. We now discuss the related work which has explicitly examined the impact of buffer size on MPTCP. Note that the impact of buffer size is not limited to MPTCP but on all other approaches using POSM.

Barré et al. [17] evaluated the impact of heterogeneous paths on the receive buffer and aggregated throughput. The experiment result shows that losses on one subflow have a limited impact on the performance of the other subflows. Nevertheless, this observation is based on the assumption that the reordering buffer is big enough to accommodate all the out-of-order data. Nguyen et al. [133, 134] evaluated the performance of MPTCP in terms of load sharing and throughput optimization with and without LIA respectively. The results show that the context of mismatched path characteristics has a great impact on the performance of MPTCP with constrained receive buffers. Han et al. [103] proposed a reordering scheme that considers packet scheduling algorithm at the sender to reduce the receive buffer. The main idea is to estimate packet arrival time and schedule packets accordingly. The sender maintains a per path time table including calculated values of receiving time at the receiver from now for each packet. When the sender has opportunity to send a new packet, it chooses the path that can deliver the packet faster than others. Thus, it is a EDPF style scheduling algorithm.

The impact of buffer size on MPTCP performance has also been observed in [37, 38, 114, 115, 116], which proposed packet coding based approaches to address it. For example, in [114] Li et al. proposed NC-MPTCP that introduces packet coding to some but not all subflows. The regular subflows deliver original packets while the coding subflows deliver linear coded packets. The coded packets are used to compensate for the lost and much delayed packets in order to avoid receive buffer blocking. They used an out-of-order scheduler that calculates the expected packet arriving time by taking RTT, throughput, and loss ratio into account. Thus, the packets that are sent out of order are expected to arrive at the receiver in order. This scheduling algorithm is the same as the FPS algorithm used in [129]. In [115], Li et al. proposed SC-MPTCP to mitigate the packet reordering issue with constrained receive buffer. In SC-MPTCP, they proposed to make use of coded packets only as redundancy to compensate for expensive retransmissions while minimizing the encoding/decoding operations. The redundancy is provisioned in both proactive and reactive manners. Specifically, SC-MPTCP transmits proactive redundancy first and then delivers the original packets. The proactive redundancy is continuously updated according to the estimated aggregate retransmission ratio. In order to avoid the proactive redundancy being underestimated, a pre-blocking warning mechanism is utilized to retrieve the reactive redundancy from the sender. Cui et al. [37, 38] proposed applying the fountain code for multipath scheduling to mitigate the impact of path heterogeneity. They also designed a data allocation algorithm based on the expected packet arriving time and decoding demand to coordinate the transmissions of different subflows. Li et al. in [116] dealt with the packet reordering issue from a different perspective. They demonstrated that the traditional Delayed ACK mechanism can lead to significant performance degradation in the presence of timeouts. Thus, they proposed a New Delayed ACK (NDA) for MPTCP aiming to remove the Minimum RTO constraint at the sender while to reserve the Delayed ACK function at the receiver.

Raiciu et al. [160] proposed schemes of opportunistic retransmission and penalizing slow subflows to avoid the reordering problem. If a subflow holds up a packet at the trailing edge of the receive window, the opportunistic retransmission allows the sender to resend the packet that is previously sent on another subflow. This scheme is similar to the Packet Re-striping used in [81, 82, 83, 84, 106] and Pre-blocking warning used in [115]. These three similar schemes are used in different scenarios but for the same purpose. Opportunistic retransmission is used only if the connection is receive-window limited. Packet Re-striping is employed in the case of path capacity fluctuations. Pre-blocking warning is triggered if the proactive redundancy is underestimated. The penalizing scheme of [160] is used to slow down the slow subflows. For example, if a subflow has caused too many out-of-order packets in the reordering buffer, the congestion window of that subflow is reduced by half and its slow-start threshold is set to the current congestion window. But if that subflow has been in the slow-start phase already, the reordering problem may become worse because the penalization mechanism will set its slow-start threshold to a smaller value. Paasch et al. [140] proposed improving the penalization mechanism by adjusting the slow-start threshold only when a subflow is not in its slow-start phase. However, they also identified that the penalization mechanism is far from perfect because a subflow at full sending speed may still overflow the receive buffer while another subflows is in slow-start.

Ferlin et al. [58] argued that the opportunistic retransmission scheme does not reduce the effect of extreme RTT heterogeneity. Instead, they proposed a Dynamic Relative Path Scoring (DRePaS) algorithm to dynamically score the paths relative to the best path and adapt the scheduling accordingly. Specifically, when the score of a path is less than a threshold, no payload is scheduled over that path until its score is larger than the threshold measured by probing traffic. Note that the standard MPTCP uses the congestion window as an estimation of path capacity. In contrast, Ferlin et al. believed that the smoothed in-flight data on each path reflects the behavior of the connection more dynamically than the congestion window.

Chen et al. [30] explored the performance of MPTCP over wireless networks. In order to avoid performance degradation, they set the receive buffer up to 8 MB, which is not feasible in practice for many devices. Shamszaman et al. [177] analyzed the feasibility of MPTCP for big data applications. They found that constrained receive buffer leads to poor performance of MPTCP. Zhou et al. [206] proposed CWA-MPTCP that examines the goodput of MPTCP with bounded receive buffers. They found that if the paths have similar end-to-end delays, the MPTCP goodput is near optimal, otherwise the goodput will be degraded significantly. For a wireless environment, they proposed a Congestion Window Adaptation (CWA) algorithm that can adjust the congestion window dynamically for each TCP subflow so as to mitigate the variation of end-to-end path delay, maintaining similar end-to-end delays over multiple paths. The primary idea behind CWA is that a large delay ratio indicates that the high-delay path is overloaded. Its congestion window needs to be decreased to relieve traffic and reduce path delay. For wired environment with stable end-to-end delay they proposed using a delay-aware scheduling algorithm to predict the receiving sequence, i.e., a FPS manner scheduler, so that packets can arrive at the receiver in order.

Paasch et al. [139] designed and implemented a generic modular scheduler framework that enables testing of different schedulers for Multipath TCP. Using this framework, they evaluated different schedulers for MPTCP and provided an in-depth performance analysis. They identified the impact of scheduling decisions on the performance of MPTCP and illustrated the underlying root cause for the observed behavior. For example, they discovered that a bad scheduling decision triggers two packet reordering effects. First, EDPF based scheduler is more efficient than simple RR in terms of avoiding the HLB problem. Second, receive-window limitation may prevent the subflows from being fully utilized.

In the last few years, several articles visited how to use delay-aware scheduling algorithms in MPTCP to improve the receive buffer utilization. Yang and Amer [202] used one-way communication delay of a TCP connection to design an MPTCP scheduler that transmits data out-of-order over multiple paths such that their arrival is in-order. Le et al. [110] dealt with the packet reordering problem of MPTCP using a Forward-Delay-based packet scheduling algorithm. Its main idea is that the sender distributes packets via multiple paths according to their estimated forward delay and throughput difference. This scheduling algorithm is an advanced version of FPS because it took throughput difference into consideration. Due to the latency asymmetry on the Internet, obtaining a good one-way delay is difficult. Therefore, Coudron et al. [35] proposed relying to the one-way delay (OWD) difference (ΔOWD) of multiple subflows to improve the scheduling. Their estimator functions more like OWTT than FPS because it estimates not only forward delay difference but also backward delay difference.

The MPTCP performance is not only impacted by the receive buffer but also by the send buffer. [201] Yang et al. found that in an MPTCP connection with several high-BDP subflows, send buffer blocking can occur and seriously decrease the overall throughput. They introduced Non-Renegable Selective Acknowledgments (NR-SACKs) to MPTCP. The idea is that once a data packet has been sacked, it can't be removed from the receive buffer. Thus, the sender can free the sacked data sooner than the advance of the MPTCP level cumulative acknowledgement. Arzani et al. [14] found that the send buffer size has significant impact on the performance of MPTCP. For example, MPTCP provides higher performance gains with a larger send buffer. However, they did not propose any method to address the problem.

Pareto-efficient MPTCP:

Khalili et al. [101, 102] demonstrated that MPTCP is not Pareto-optimal because they found that MPTCP users can be excessively aggressive toward TCP users over congested paths even without any benefit to the MPTCP users. They attributed the problem to the LIA of MPTCP. To deal with the problem, they proposed an Opportunistic Linked Increases Algorithm (OLIA) as an alternative for LIA and proved that OLIA is Pareto-optimal and satisfies the three design goals of MPTCP. Like LIA, OLIA is a window-based congestion-control algorithm that couples the increase of congestion windows and uses unmodified TCP behavior in the case of loss. The increase part of OLIA has two terms. The first term provides the Pareto optimality. The second term guarantees non-flappiness[6] as MPTCP with LIA and responsiveness (i.e., the rate of algorithm convergence). OLIA also compensates for different RTTs by adapting the window increases as a function of RTTs. However, Singh et al. [182] found that OLIA of MPTCP still has performance issues. They presented Adapted Opportunistic Linked Increases Algorithm (AOLIA) to ensure controlled aggressiveness of the MPTCP subflows. In order to minimize the packet reordering delay, they also proposed a Push-Pull-Hybrid (PSPLH) scheduler where Pull strategy is used to allocate data segments to multiple flows, and Push strategy is used to tune the size of the segments dynamically.

Path Diversity for MPTCP:

[6]Flappiness means that MPTCP would use one path almost exclusively for a while, then flip to another path, and then repeat.

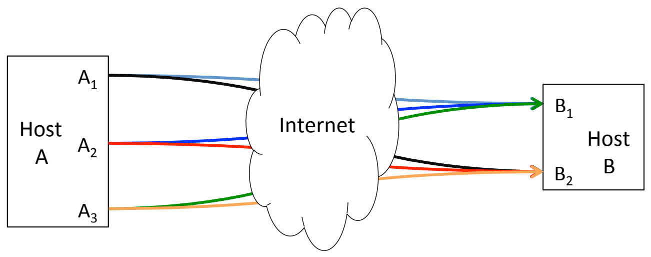

Figure 9. Static full-mesh of possible network paths between two MPTCP enabled hosts: A_1:B_1, A_1:B_2, A_2:B_1, A_2:B_2, A_3:B_1, A_3:B_2.

By default, MPTCP uses a "fullmesh" manager to create static full-mesh of possible network paths among the available IP addresses (see Figure 9). This path management may not only lead to a large number of subflows being established but also ignore the benefits the path diversity could offer. van der Pol et al. [194] combined MPTCP and OpenFlow to dynamically exploit path diversity (choose disjoint paths) between two endpoints to improve stability (in the case of partial path failure) and obtain higher throughput. Specifically, in [194] OpenFlow is used to discover the topology of the network, calculate multiple disjoint paths and configure these paths. MPTCP is used to distribute the traffic across the selected paths.

In contrast to limiting the number of subflows [194], Coudron et al. [34, 36] exploited the path diversity in cloud networks from a different perspective, for example, creating more subflows over disjoint WAN paths (under the assumption that the WAN capacity is the throughput bottleneck). To achieve this goal, in [36] Coudron et al. proposed a cross-layer approach which combines MPTCP and LISP [56]. LISP is used to gather information and give hints to MPTCP about the WAN paths diversity between two MPTCP endpoints. MPTCP could use the path manager "ndiffports" to create additional LISP-enabled subflows which share the same source IP with different TCP ports. In their another work [34], apart from MPTCP and LISP, Coudron et al. introduced one more protocol TRILL [51] to support Ethernet level multipath transmission. For example, MPTCP employs coordinated congestion control to achieve RP, LISP handles path diversity at the external data center packets between border nodes, and TRILL deals with multipath Ethernet-layer communications.

Comparison between CMT-SCTP and MPTCP

SCTP and TCP are the main base protocols widely used to support multipath transportation. Currently, CMT-SCTP and MPTCP are in the focus of the IETF and academia. In the following discussion, we make a comparison of them.

Although CMT enabled SCTP shares the same issues with MPTCP in terms of fairness, reordering, and retransmission policies, moving legacy applications from TCP to SCTP involves a number of challenges such as making SCTP work through NATs, the need to modify applications, and the lack of an easy way to negotiate SCTP versus TCP between a client and a server. None of the issues are insurmountable, but together they make adoption of SCTP as a TCP alternative a challenge. From the previous discussion, we found that MPTCP instead of CMT-SCTP has become the main stream of multipath transportation solution. In this section, we discuss the reason behind it.

By comparing the key algorithms used by CMT-SCTP and MPTCP (see Table X and Table XII), we found that the two protocols have shared the same or similar key algorithms. Therefore, algorithm is not the primary factor that determines which protocol could become the mainstream because the algorithms are not specific to any base protocols but could be used on any of them with minor adaptation. We believe that the reason mainly comes from their difference in terms of backward compatibility and sequence number design. For example, unlike SCTP which modifies the interfaces of legacy TCP to applications, MPTCP presents a single TCP interface to the application layer (see Figure 8). This seemingly minor difference makes MPTCP compatible with all legacy applications. As such, the implementation of multipath in TCP, which dominates Internet traffic, is a much more attractive deployment strategy.

The sequence space design is another primary difference. To make a fair comparison, we assume that SCTP has been widely deployed as TCP has, otherwise CMT-SCTP packets would be dropped by legacy middleboxes. As summarized in Table X, CMT-SCTP keeps using one single sequence space as SCTP does. Therefore, a new SACK mechanism and its interpretation accordingly are required to avoid spurious retransmissions. Moreover, single sequence space makes bytestream discontinuous on multiple paths so that certain middleboxes may break such associations [160]. In contrast, in MPTCP the sequence numbers carried in the TCP headers are separate on each path so that the interpretation of out-of-order packets and ACKs remain the same as before. Hesmans et al. [78] and Honda et al. [80] found that MPTCP could traverse most of the middleboxes because of its double sequence space design. Furthermore, with checksums MPTCP can detect middleboxes interference and fallback to legacy TCP. To allow clients to benefit from MPTCP in its early deployment (e.g., servers have not upgraded to support MPTCP), Detal et al. [41] proposed a protocol converter, MIMBox, to translate MPTCP to TCP and vice versa. MPTCP is not only little influenced by middleboxes but is even extended to explicitly add specified middleboxes in the middle of an ongoing communication. For example, the proposed solution in [136] used MPTCP to implement connection acrobatics (i.e., the ability to explicitly redirect connections via a middle point and the ability to migrate the endpoint of a connection). Therefore, MPTCP connections can be redirected to middleboxes located anywhere in the Internet to improve services like load balancing, DDoS filtering and anycast.

One more difference between CMT-SCTP and MPTCP lies in the path management strategy. By default, MPTCP creates a full-mesh of possible network paths among the available IP addresses, whereas CMT-SCTP only uses pairs of addresses to set up communication paths (creating only one additional path per additional source address). Becke et al. [18] found that MPTCP's path management strategy performs significantly better than that of CMT-SCTP in the case of asymmetric paths.

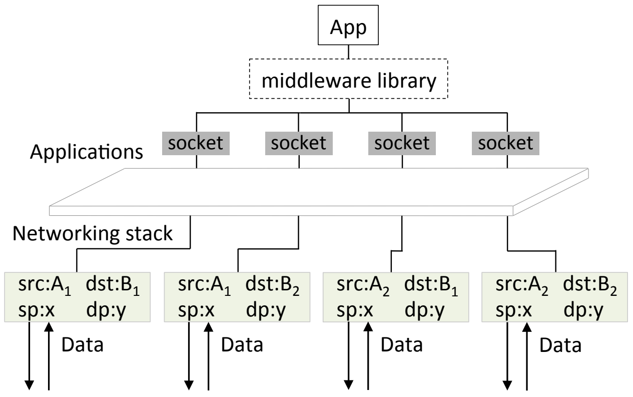
App
middleware library
Applications
socket
socket
socket
socket
Networking stack
src:A$_1$ dst:B$_1$ sp:x dp:y
src:A$_1$ dst:B$_2$ sp:x dp:y
src:A$_2$ dst:B$_1$ sp:x dp:y
src:A$_2$ dst:B$_2$ sp:x dp:y
Data
Data
Data
Data

Figure 10. Multipath transmission on application layer. It is assumed that each host has two interfaces. src: Source IP address, dst: Destination IP address, sp: Source Port, dp: Destination Port, APP: Application.

D. Application Layer Multipath Capability

Provisioning multipath capability at the application layer has received a lot of attention because the approaches are almost independent of the underlying access technologies and network-layer routing. It is a common practice that an application establishes multiple transport connections, binds them to different IP addresses, and distributes the data in proportion to the available path capacity over these connections (see Figure 10). To reassemble the data delivered over different connections, each packet or a group of packets are usually assigned additional sequence numbers.

In the rest of this section, we also divide the approaches into four groups. The first group discusses approaches using the same path. The second group discusses approaches using different paths. The third group investigates approaches based on HTTP in order to highlight the importance of the combination of HTTP and multipath transmission. The fourth group exploits middleware approaches. Table XIII and Table XIV are used to summarize the key algorithms and approaches respectively.

1) Multiple Connections over the Same Path: In the early stage of research work on application layer multipath transmission, the focus was on bandwidth aggregation using multiple TCP connections over the same physical path. For instance, Allman et al. [10] developed a new application called XFTP, a modified version of FTP [152], to improve the poor performance of TCP over long-fat satellite channels. XFTP creates multiple TCP connections. To send a file, XFTP divides the file into records, reads the file from local storage one record at a time, and sends each record over whichever connection has resource available for transmission. To reassemble the records into the correct order, XFTP uses an additional 4-byte sequence number to each record. GridFTP [92] is another extension of the FTP protocol implemented for bulk data transfer, where parallel TCP connections are created to increase the throughput in a bottleneck link. Specifically, GridFTP divides the data to be transferred into multiple portions and transfers each portion with a separate TCP connection. When competing with non-GridFTP connections over a bottleneck link, the GridFTP connections will be less likely to be selected to drop their packets. Hacker et al. [72] examined the effects of using parallel TCP flows to improve end-to-end network performance. They found that in the absence of congestion, the use of parallel TCP connections is equivalent to using a large Maximum Segment Size (MSS) on a single connection. In addition, they addressed the question of how to select the maximum number of connections to maximize the overall throughput while avoiding congestion. For example, if the selected value is too large, the aggregate flow may cause network congestion and throughput will not be optimized.

2) Multiple Connections over Different Paths: The approaches mentioned above aim to increase application throughput by using multiple TCP connections through the same physical path. Nevertheless, if they are used for striping data over different physical paths, the reordering issue at the receiver would render them inefficient because they do not take into consideration the reordering issue caused by heterogeneous paths. In the 2000s, researchers started to seek solutions to provide bandwidth aggregation over different physical paths. A simple approach to achieve this goal is to directly add support for multiple interfaces to a given application by opening multiple TCP sockets (one for each active interface), and performing striping of data across different sockets. If the interfaces of a client are connected to independent networks, the simultaneous use of multiple paths can achieve a total throughput close to the sum of all the throughput from individual interfaces.

Given that popular files are often replicated on multiple servers, it becomes natural for clients to connect in parallel to several mirror servers to retrieve a file (i.e., many-to-one fashion). Golubchik et al. in [64] investigated the potential benefits of an application layer multipath streaming approach between a set of senders and a receiver. They found that multipath streaming exhibits better loss characteristics than single path streaming. Rodriguez and Biersack in [165] described a parallel-access (PA) scheme that allows users to fetch different portions of a file from multiple servers at the same time and reassemble the file locally. The PA scheme allows dynamic load sharing among all servers so that faster servers will deliver bigger portions of a file while slower ones will deliver smaller portions.

Shiwen et al. proposed Multiflow Real-time Transport Protocol (MRTP) [123] which is a multipath transmission extension to Realtime Transport Protocol (RTP) [173] for real-time applications. MRTP supports multimedia services by exploring multipath transport in mobile ad hoc networks, where link bandwidth may fluctuate and paths are unreliable. The authors studied the impact of traffic partitioning on congestion at bottleneck links and found that the bandwidth utilization of a bottleneck node could be much improved by two strategies (thinning and striping [22]). Furthermore, they showed analytically that most of the performance improvement can be achieved with a few paths (e.g., two or three paths), while only marginal improvement is gained by further increase in the number of paths.

Wang et al. [195, 196] proposed Dynamic MPath-Streaming (DMP), a scheme for live streaming over multiple TCP connections. DMP allocates packets over multiple paths according to their current throughput. DMP does not use an explicit prob-

Table XIII
KEY ALGORITHMS FOR APPLICATION LAYER MULTIPATH CAPABILITY (sorted according to their order mentioned in Table XIV).

Algorithm	Problems to address	Description
HTTP Range Retrieval Request (HTTP-RRR)	Request segments over multiple paths	It commonly makes a halted download to proceed with the outstanding portion at a later time. In multipath transmission, it could be used to download unique segments of a file at the server. Note that each request must be sequentially handled before the next request could be sent out.
HTTP Request Pipelining (HTTP-RP)	Concurrent HTTP-RRR	It generates multiple requests from the client without waiting for each response from the server.

Table XIV
CONCURRENT MULTIPATH TRANSFER APPLICATIONS.

Scheme	Year	Algorithm and Protocol	Path(s)	Network Environment
XFTP [10]	1996	WRR	Same	Satellite channel
PSockets [184]	2000	Middleware, RR	Same	General
GridFTP [92]	2001	Fair share (RR)	Same	Bottleneck link
PA [165]	2002	WRR	Different	Many senders and one receiver
ATLB [74, 75]	2005, 2007	Middleware, WRR, FPS	Different	General
Tavarua [155]	2006	Middleware, WRR	Different	Cellular uplink channel
SBAM [171]	2006	Middleware, WRR	Different	Wireless access
DMP [195, 196]	2007, 2009	WRR-PULL	Different	General
MultiTCP [192]	2008	Receiver-driven rate control	Different	Insufficient bandwidth due to traffic bursts
PATTHEL [16]	2009	Middleware, WRR	General	General
Kaspar et al. [97]	2010	WRR, HTTP-RRR	Different	Wireless
Kaspar et al. [96]	2010	WRR, HTTP-RRR, HTTP-RP	Different	Wireless
Evensen et al. [55]	2010	HTTP-RRR, HTTP-RP, Request scheduler, WRR	Different	Wireless
Evensen et al. [53, 54]	2011, 2012	HTTP-RRR and HTTP-RP, Improved Request scheduler, WRR	Different	Wireless
Miyazaki et al. [130]	2012	Middleware, EDPF	Different	Wireless
DBAS [70, 71]	2012, 2013	Middleware, WRR, PFA	Different	Wireless
G-DBAS [69]	2012	Middleware, Energy-awareness Scheduling, WRR, PFA	Different	Wireless
OPERETTA [66]	2012	Middleware, Energy-awareness Scheduling, WRR	Different	Wireless
MPTS-AR [112, 205]	2014, 2015	OpenPath, MPTP	Different	General

ing scheme for bandwidth estimation on each path. Instead, it uses the WRR-PULL scheduler to allow each connection to pull data from a shared queue whenever it has opportunity to send data. Thus, the paths with higher throughput deliver more packets than others.

Tullimas et al. [192] proposed MultiTCP for multimedia streaming. It aims at providing resilience against short term insufficient bandwidth due to traffic bursts by using multiple TCP connections for the same application. MultiTCP is a "smart" application that allows the application to control the desired sending rate during congestion periods. MultiTCP achieves rate control by the means of adjusting the receiver window.

Zhang et al. [205] proposed a general framework of multipath transport system based on application-level relay (MPTS-AR), currently under the standardization within the IETF [112]. This framework defines three logical entities and two protocols. The entities include user agent, relay server, and relay controller. The protocols are OpenPath and MPTP (Multipath Transport Protocol), which are used in control plane to manage relay paths and in data plane to facilitate multipath data transport respectively. However, they left a few key functions we concern the most out of the scope. For example, how to split the original data stream into several substreams, how to mitigate the reordering issue at the receiving side, how to provide path diversity among all available paths, and how to obtain the performance of overlay paths are all out of the scope.

3) HTTP based Multipath Media Streaming: In addition to multipath approaches for specific applications, HTTP [59] with multipath capability is currently one of the most common protocols for streaming video on the Internet through multiple paths. Kaspar et al. [96, 97] and Evensen et al. [53, 54, 55] presented HTTP-based methods for downloading multimedia content simultaneously over multiple network interfaces.

Kaspar et al. [97] proposed an HTTP-based on-demand streaming service over multiple wireless access networks. They presented a proof-of-concept implementation of a progressive download service, which uses HTTP-RRR capability [59] to download specific segments of a file from a media server. The drawback of the range retrieval request is that each request must be handled sequentially by the server before the client can send the next request. Thus, an average time overhead of one RTT is introduced for each request. In order to avoid waiting for each response, in [96] they presented an improved version of their work by using an additional HTTP capability, i.e., HTTP-RP [59]. The request pipelining function of HTTP allows a client to make multiple requests simultaneously. In [97], they studied the effect of file segmentation on the buffer requirements and found that there exists an optimal segment size for which the aggregation efficiency is maximized. In [96], they found that due to the use of request pipelining, very small segments can provide efficient throughput aggregation.

Evensen et al. [55] introduced an adaptive, WRR-PULL based scheduler that achieves smooth playback by scheduling requests for video segments of different quality levels over multiple interfaces simultaneously. Like their previous work in [97], they still utilized HTTP-RRR and HTTP-RP functions to support multipath transmission. In order to avoid video deadline misses, they proposed an additional request scheduler in [55]. The scheduler is mainly used for estimation of the aggregated throughput for chosen video quality level and request of segments over the available interfaces. However, the weakness of the request scheduler is that the segments are divided into fixed-sized subsegments, which may lead to low performance with constrained receive buffer. Evensen et al. [53, 54] proposed improving the request scheduler by loosing the segment size constraint. For example, the segment sizes are dynamically calculated on the fly based on the capability of each path.

4) Session layer Multipath Capability: Unlike application layer approaches discussed previously, some approaches open multiple TCP flows without any change to existing applications by providing specialized middleware or virtual sockets at the session layer between the application and transport layer.

Sivakumar et al. proposed PSockets (Parallel Sockets) in [184]. PSockets is a library that transparently partitions upper layer data into multiple transport streams through the same physical path. The principal idea is to split data equally across several open sockets without application upgrade. PSockets has the same Application Programming Interfaces (APIs) as those of a regular socket. Note that this work was published in the year 2000. Back then, the TCP window size had to be tuned manually for high-speed networks at both the source and the destination in order to achieve the maximum throughput. Differently, PSockets allowed applications to achieve the best performance without tuning the window size.

Yohei et al. [74, 75] proposed an Arrival-Time matching Load-Balancing (ATLB) layer between the application layer and TCP layer. ATLB consists of a distributed data transfer method and a path-failure detection/recovery method. In order to mitigate the reordering effect at the receiver, ATLB calculates the data arrival time for each path. It considers the time that data segments spend in the TCP queue at a sender and the time needed for data segments to pass through the network.

Qureshi et al. presented a prototype system Tavarua in [155]. Tavarua is a middleware for providing network striping capability to applications with high demands on uplink throughput. Note that Tavarua runs upon UDP. In an effort to mitigate the impact of variations in bandwidth, an application can use feedback information to estimate the maximum available data transmission rate. Then the bit-rate at which the video is encoded is adapted dynamically. The middleware also handles low-level issues related to the network interfaces (e.g., congestion control, disconnections, and reconnections).

Sakakibara et al. [171] proposed a Socket-level Bandwidth Aggregation Mechanism (SBAM) to offer aggregated bandwidth. SBAM is located at the socket layer (close to TCP) so that it can collect system resources efficiently. For example, it has a network monitoring function to collect delay and available bandwidth of each path. Using this information, the traffic scheduler decides the amount of data to fill the bandwidth-delay product of each path.

Like the other middleware approaches, Parallel TCP Transfers Helper (PATTHEL) [16] also provides APIs for applications. The difference of PATTHEL lies in two facts. First, PATTHEL incorporates a separate data connection and control connection, where the control connection is created first to manage the other data connections for the entire communication period. Second, PATTHEL is a cross-layer protocol because it adds an entrance to the routing table in order to deliver data over a certain channel.

Miyazaki et al. [130] examined how much receive buffer is needed in various scenarios and found that the buffer size is proportional to the ratio of the bandwidth of the two interfaces. A larger bandwidth difference leads to a bigger receive buffer and vice versa. The scheduling algorithm used in [130] is EDPF.

Habak et al. [70, 71] proposed a Deplorable Bandwidth Aggregation System (DBAS) middleware architecture for multi-interface enabled devices. Like the work in [68, 111, 166], DBAS also supports both FOSM and POSM. In FOSM where the server is not DBAS enabled, DBAS schedules different connections to the interfaces such that a connection can be assigned to only one of the available interfaces. If both sides are DBAS enabled, POSM is used so that each packet can be scheduled independently on a different interface. To make better scheduling decisions, DBAS estimates the characteristics of the applications dynamically based on their behavior and stores them in a database for history track. DBAS focuses on the actual implementation of the basic core system. The authors presented an extended work based on DBAS, a Green DBAS (G-DBAS) [69] to balance overall throughput with energy consumption. For example, they introduced a new utility based scheduler that takes energy consumption of each interface into account in order to balance the trade-off between maximizing throughput and minimizing power consumption. Note that G-DBAS only works in FOSM. OPERETTA [66] is an extension of G-DBAS to support POSM.

Application layer approaches split a single file or byte stream into segments that are transmitted concurrently over different paths. These kind of approaches seem to be simple in the sense that the applications are in full control of the striping decisions. Thus, it does not require any protocol change at lower layers so that clients and servers can find an optimal way to collaborate. Nevertheless, the complexity and overhead at the application layer are considerable. For example, an application-level sequence number has to be included in each of the application defined headers. Meanwhile, the application has to explicitly ensure that the application layer data units, which carry unique application-level sequence numbers, do not get fragmented during transmission. Moreover, a dedicated resequencing mechanism is required to reassemble the data at the receiver. In practice, different paths may have diverse characteristics, and the striping ratio may not exactly match the ratio of data rate from different paths. A large receive buffer (on the application level) is required to accommodate the out-of-order data. Finally, in order to split intelligently, the application has to implement a bandwidth estimation mechanism redundantly despite the same mechanism has been employed by TCP through its congestion control mechanism.

The middleware approaches are very similar to application-layer approaches. They also face the same challenges, for example, the reordering issue. The advantage of middleware approaches is that although it still requires client and server-side modifications, applications usually are not required to be upgraded.

E. Summary

In this section, we summarize the issues that are common to the approaches from all layers we have covered in this survey. Specifically, we first discuss the packet reordering problem and how effective the widely used scheduling algorithms are to mitigate it. After that, we present the approaches which have adopted the cross-layer design. Next, we compare the approaches' compatibility capability, which is inherently determined by their stack positions. At last, we summarize the research problem evolution on each layer.

Table XV
EVALUATION OF END-TO-END PACKET REORDERING ALGORITHMS.

Algorithm	**Reordering**	**Load-sharing**
PCA, PFA, Multi-streaming	++	-
FPS	+	++
EDPF, PET	o	+
WRR	o	++

1) Packet Reordering: When packets travel through different paths which may have mismatched characteristics, they may arrive at the destination out of order. All the presented approaches deal, to some extent, with the reordering issue on the layer which they are located at. If they ignore or have no control over the reordering mechanism on the transport layer, their approach may suffer from performance degradation because of the misinterpretation of out-of-order packets. Table XV lists the primary algorithms used to mitigate packet reordering without considering which layers they are located at. We group and sort them according to their effectiveness in terms of packet reordering and load-sharing capability. We use four levels (++, +, o, -) to grade the mechanisms only for general and relative evaluation. The level ++ indicates the most efficiency. + comes second, and then o and -. A scheme graded - does not imply that it is useless. Instead, that scheme may work well either in certain network context or with additional buffer management.

The first group includes PCA [2, 88, 135, 188], PFA [51, 57, 68, 69, 70, 71, 95, 111, 127, 150, 166], and Multi-streaming [50]. They are the most effective mechanisms which can completely eliminate packet reordering incurred by multipath transmission because the data units required sequencing at the destination are assigned only to the same path. However, a multipath transmission protocol with them usually performs worse than without them in terms of load sharing. For example, if the number of flows is less than that of available paths, there would exist paths which cannot be fully utilized. Therefore, we give - to their load-sharing capability due to their lower performance than average.

FPS [37, 38, 74, 75, 110, 114, 115, 129, 202, 206] breaks the in-order scheduling rule at the source. Whenever a path has opportunity to send a new packet, it estimates that path's capacity and chooses a new data block accordingly so that out-of-order sent out packets could arrive at the receiver in-order. We grade FPS a + to its packet reordering capacity because it concerns most on whether the packets arrive at the receiver, and a ++ to its load-sharing capacity because fully utilizes each path's capacity.

In EDPF [24, 81, 103, 106, 130, 178, 179] and PET [27], the scheduler first sends data on a path with lowest RTT until it has filled its congestion window. Then, the data is sent on the subflow with the next lowest RTT. We grade them o and + to their reordering and load-sharing capacities respectively because EDPF and PET preferentially schedule data on the available path with lowest RTT. The larger the RTT difference is, the more out-of-order packets arrive at the destination. Moreover, EDPF and PET consider only path bandwidth and latency, ignoring packet loss rate caused by congestion. Therefore, it may achieve sub-optimal load-balancing in the presence of high losses or heavy congestion.

WRR [5, 6, 10, 13, 16, 17, 46, 52, 53, 54, 55, 56, 66, 68, 69, 70, 71, 74, 75, 76, 82, 83, 84, 85, 96, 97, 101, 102, 118, 121, 148, 149, 155, 161, 162, 165, 166, 167, 168, 171, 172, 185, 186, 195, 196, 198, 204] is the most widely used scheduling algorithm on all layers with maximizing the overall throughput as its first priority. It could achieve the goal if the multiple paths have similar characteristics. Otherwise, it would cause significant out-of-order packets at the receiver because it considers little about its impact on packet reordering. Therefore, WRR works better on link layer than other layers because link layer has relatively stable link state. Like FPS, WRR could fully utilize the available path capacity. Based on the discussion, we give o to WRR's packet reordering capacity and ++ to its load-sharing capability.

In practical network environments where path characteristics may change dynamically, the scheduling algorithms except PFA, PCA and Multi-streaming may fail to counter against packet reordering. Several additional mechanisms have been proposed to work coherently with the scheduling algorithms, such as ACK manipulation [46, 68, 89, 104, 111, 116, 122, 167], buffering management [8, 26, 49, 68, 109], packet coding [38, 114, 115, 116, 178], blocking warning and fast retrieving [81, 82, 83, 84, 106, 115, 160], slow-path penalization [58, 140, 160] and so on. The comparison of various combinations of the scheduling algorithms is out of the scope of our knowledge because they may be used in various network environments, triggered by different conditions, and supported by diverse assumptions.

To make a conclude on the discussion of packet reordering, the choice of the best scheduling policy depends on path characteristics. There is no single scheduling mechanism which can handle all scenarios. To adapt to different network environments, several approaches, such as [68, 70, 71, 111, 166], could support both flow level scheduling and packet level scheduling strategies.

2) Layer-dependent scheduling algorithms: In our previous discussion, we could find that many scheduling algorithms are shared by different approaches on various layers. For example, FPS variants are used on transport (including [37, 38, 110, 114, 115, 129, 202, 206]) and application layers (including [74, 75]). EDPF variants are used on network layer (including [27, 65, 104, 105, 109]), transport layer (including [24, 81, 103, 106, 178, 179]) and application layer (including [130]). WRR variants are used from link to application layers. However, we argue that it does not imply that these scheduling algorithms are layer-independent. Instead, they are mostly layer-dependent.

Running a scheduling algorithm usually requires measuring bandwidth, delay, loss, or jitter to provide best effort or even QoS guarantees. No single measurement on a certain layer could always give accurate measurements. The measurements may even vary on different layers. Although how to measure those metrics is an entire problem unto its own, the efficiency of the algorithm is closely connected to the correctness of the measurements. Therefore, we argue that the scheduling algorithms which rely on the estimation of certain path characteristics are layer-dependent.

We discuss WRR as an example to support our argument. The same principle is applicable to other algorithms which rely on the estimation of path characteristics. WRR is the most widely used scheduling algorithm. Although each layer adopts many variants of the WRR algorithm, its effectiveness highly depends on how correctly the path capability is estimated in a dynamic fashion. For example, using the congestion window size on the transport layer to estimate the path capability is a more lightweight and accurate way than those using various probing methods on other layers. Furthermore, WRR works much better on link layer than upper layers because link layer has more stable link status.

3) Cross-layer Support: In this survey, we define that any attempt to violate the TCP/IP reference model is considered a cross-layer design. Among the approaches we have discussed previously, several of them have explicitly involved cross-layer interaction for purposes of estimating path status to avoid packet delays and losses, scheduling traffic over multiple paths according to their capacity, exploring path diversity to obtain high throughput, achieving better QoS for multimedia applications, and so on. Due to these benefits the cross-layer design may offer, there has been increased interest in protocols with interactions between various layers of the network stack. In the rest of this section, we discuss the cross-layer approaches based on our previous discussion and give a few observations on them without our own judgment. Instead, we refer readers to Kawadia and Kumar's work [98] which calls for a cautionary approach to cross-layer design. Although [98] examined the issue of cross-layer design in wireless networks, we believe the same principle is also applicable for multipath transmission.

Table XVI shows approaches based on a cross-layer design. The "Base Layer" indicates the layer where the data splitting is initiated, and the "Additional Layers" indicate which layers are required to provide support to the base layer. From the table, we observe that the transport and application layers are the main base layers. From 2005 to 2013, it was drawn most of the attention to implement the base layer on the application level. Some applications obtain low layer information to optimize their behavior in terms of interface selection, load balancing, and energy efficiency. The information includes throughput history and smoothed RTT from transport level, routing table from IP level, and link status (e.g., energy consumption, available bandwidth, and bit error rate). Some applications can even change the low layer protocol behaviors such as changing TCP window size to control the throughput, modifying the routing table to optimize path selection, and disconnecting/reconnecting certain interfaces for energy efficiency or partial failure. In most recent years, it has become more attractive to use transport layer as the base layer with additional support from network and link layers. Although transport layer approaches have advantages from the congestion control mechanism, they lack the choice of path diversity to free the constraint of fairness control. The path optimization support from network and link layers can compensate for the weakness in network and Ethernet levels respectively. In addition, some transport layers can suspend or release a path based on the estimated MAC information.

We can also find that the interaction of cross-layer design may not be limited to adjacent protocol layers. Instead, it allows vertical communication to take place between nonadjacent layers. The cross-layer design approaches are actually not limited to the ones listed in Table XVI because although many approaches above the network layer did not mention or specify how packets are delivered through multiple interfaces, the cross-layer support from routing function on network layer may be assumed implicitly.

4) Compatibility: In this section, we evaluate the compatibility capacity of the approaches on different layers. The evaluation is made in general because there may be exceptions in certain approaches. Table XVII presents the evaluation where we separate network layer approaches into three categories according to how many proxies required in each category. We use MPTCP and CMT-SCTP to represent the transport layer approaches and evaluate their compatibility separately. Likewise, we also separate session layer approaches from application layer approaches.

Application compatibility means that the lower layer changes do not require the legacy applications to be upgraded. Obviously, the link layer and all the network layer approaches are compatible with the legacy applications because they do not change the socket interface between the legacy applications and transport layer protocols. MPTCP presents a standard TCP socket API to the application so that legacy applications can run upon MPTCP transparently. However, the legacy applications running on TCP have to upgrade in order to take advantage of CMT-SCTP. Session layer approaches usually keep the same socket API to applications (e.g, PSockets [184]) so that they require no changes to the legacy applications. Application layer approaches have a serious application com-

Table XVI
CROSS-LAYER SUPPORT FOR MULTIPATH TRANSMISSION.

Scheme	Year	Base Layer	Additional Layers	Description
PRISM [104, 105]	2005, 2007	Network	Transport	1) Use the transport-layer information carried in ACKs to mask adverse effects of out-of-order packet deliveries. 2) tune the TCP congestion control mechanism to handle, at a sender side, packet losses.
ATLB [74, 75]	2005, 2007	Application	Transport	Use the TCP connection's throughput history to calculate the queuing delay in the sending buffer and use TCP's smoothed RTT to calculate the network path delay.
Tavarua [155]	2006	Application	Transport, Link	Handle low-level issues related to congestion control and interface disconnection/reconnection.
SBAM [171]	2006	Application	Network, link	1) Send ICMP packets to periodically measure path delays. 2) read routing functionality to route packets through different network interfaces. 3) monitor link status information on both sides, for example, available network interfaces as well as their up/down status.
MultiTCP [192]	2008	Application	Transport	Dynamically change the receiver's TCP window size to control the throughput of each TCP connection.
PATTHEL [16]	2009	Application	Network	Add entrances to the routing table in order to deliver data over a certain path.
G-DBAS [69]	2012	Application	Link	Estimate the characteristics of each network interface as well as the energy consumption of each interface.
OPERETTA [66]	2012	Application	Link	Estimate the available bandwidth and the energy consumption of each interface.
DBAS [70, 71]	2012, 2013	Application	Link	Estimate the characteristics of each network interface, for example, the available bandwidth and packet error rate.
OpenFlow-MPTCP [194]	2013	Transport	Link	Use OpenFlow to discover the topology of the network, calculate multiple paths and configure these paths on the OpenFlow network. MPTCP distributes the load across the selected paths.
A-MPTCP [36]	2013	Transport	Network	Use LISP [56] to give better knowledge of the underlying IP topology to MPTCP enabled endpoints in cloud networks.
Coudron et al. [34]	2013	Transport	Network, link	Use LISP [56] and TRILL [51] respectively in different environments to calculate and select multiple paths for MPTCP in data centers. LISP [56] handles path diversity between border nodes and TRILL [51] deals with multipath on Ethernet layer.
MPTCP-MA [119]	2014	Transport	Link	Use MAC information to estimate the path status so as to suspend or release a path based on the estimation.

Table XVII
COMPATIBILITY EVALUATION ($\surd$ means being supported).

Compatibility	Link	Network (no proxy)	Network (1 proxy)	Network (2 proxies)	MPTCP	CMT-SCTP	Session	Application
Application	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$		$\surd$	
Backward		$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$
Middlebox	$\surd$				$\surd$		$\surd$	$\surd$
Host				$\surd$				
Infrastructure	$\surd$	$\surd$			$\surd$	$\surd$	$\surd$	$\surd$

patibility issue because the multipath transmission property needs to be implemented for specific applications.

An updated approach has backward compatibility if it can either work with its communicating peer which uses the standard approach or automatically fallback to the standard approach if the communicating peer does not support the new features. Link layer approaches generally do not maintain the backward compatibility because most of them are proprietary approaches and require dedicated setup on both sides. Therefore, we mark that the link layer approaches are not backwards compatible. Network layer approaches with no proxy usually employ some of the fields in protocol headers (e.g., [65, 68, 122, 148, 149, 150]) to negotiate multiple IP addresses to be used or piggy-back information so as to provide backward compatibility. Network layer approaches with one proxy has backward compatibility because the communicating peer is unaware of the multipath transmission between the client and the proxy. The network layer approaches requiring two proxies are also backwards compatible because both sides are unaware of the multipath transmission between the two proxies. MPTCP is backwards compatible with plain TCP because it can fallback to single-path TCP if the communicating host does not support the extensions. CMT-SCTP related articles did mention at all whether it can fallback to SCTP if the server is not CMT-SCTP enabled. We believe CMT-SCTP could also fallback to SCTP if the server does not support concurrent multipath transfer. But anyway, CMT-SCTP has inherited the backward compatibility issue of SCTP itself. For example, if the server is only aware of TCP operations, a CMP-SCTP client may fail to create connection with the server. Session layer and application layer approaches are generally backwards compatible with the legacy applications. When connecting to a server, a client application usually specifies two different TCP ports, a probe one and a fallback one. First, the client tries to establish a connection using the probe port to check whether the server is upgraded. If the operation fails, another attempt is made by using the fallback port to create a standard connection.

The compatibility with existing middleboxes, such as firewalls and NATs, affects whether the "new" packets are able to traverse the legacy middleboxes. First of all, link layer approaches generally would not have this issue because the communication paths or trees are well designed between dedicated multipath-aware devices. Most network layer approaches (with and without proxies) split bytestream over multiple paths. Therefore, the single sequence space across more than one path leaves gaps in the sequence space seen on any individual path, which may upset certain middleboxes. To solve this issue, the double sequence space design was proved to be an effective solution [80, 160]. However, how to use the double space also influenced the outcome. For example, HIP based multipath transmission adopts double sequence space. But the additional sequence space is used for the purpose of anti-replay instead of resequencing. In contrast, MPTCP packets, which carry double sequence numbers for resequencing on two levels, can traverse most of the middleboxes. CMT-SCTP packets may fail to traverse through certain middleboxes due to its single sequence space design. Session and application layer approaches create multiple standalone TCP flows so that their TCP flows can travel through various middleboxes as normal TCP does.

Host compatibility means whether the approaches require changes in hosts. We found that all approaches need changes on hosts except the network layer approaches with two proxies because the multipath transmission between the two proxies are unknown to both communicating peers.

Infrastructure compatibility means whether an approach needs additional network infrastructure such as NAT box and proxy. We found that only the network layer approaches with the proxy support require additional infrastructure.

According to the previous discussion, it is hard to implement a generic multipath solution which can satisfy all the compatibility requirements. As summarized in Table XVII, we have a few more observations. For example, MPTCP and session layer approaches have more compatibility support than others. CMT-SCTP has inherited SCTP's application compatibility issue, which becomes the major obstacle of its real deployment. Network layer approaches lack discussion on backward compatibility.

5) Evolution of Research Problems: Table XVIII presents the research problems the approaches on each layer have tried to address. In the early stage of multipath transmission research, most approaches emphasized only bandwidth aggregation with various scheduling and packet reordering algorithms. Few of them considered the fairness and RP features. Today, these two problems as well as Pareto-optimality problem have become challenges along with the revolutionary development of multipath transmission.

The fairness requirement on multipath transmission was unclear in the beginning. For example, the early research on multipath transmission focused on bandwidth aggregation by taking advantage of the resources through multiple interfaces. The target in the research community matched the potential expectation of end users because an end user can benefit from the aggregated bandwidth if they have paid for both accesses. Thus, the fairness emphasized the fairness of each individual subflow; for example, each subflow gets as much bandwidth as a standalone TCP flow does. In recent years, the research focus was on the fairness of the multiple subflows as a whole at shared bottlenecks. The principle is that a multipath transmission should behave as a single TCP flow at shared bottlenecks. Coordinated congestion control algorithms are used as a powerful tool to achieve it.

The concept of RP [200] was proposed in 2008. The early approaches before it also proposed using less congested paths more, which is one of the RP principles. From the congestion control algorithm viewpoint, the difference between these approaches and the approaches after 2008 is that the latter ones use coordinated congestion control algorithms instead of independently tuning each path's congestion control behavior.

The Pareto-optimality is a state of resource allocation in which there is no alternative state that would make some people better off without making anyone worse off. MPTCP with LIA [17, 60, 158, 162] fails to satisfy the Pareto-optimality because upgrading some regular TCP users to MPTCP can reduce the throughput of other users without any benefit to the upgraded users. OLIA [101, 102] as an alternative for LIA could make MPTCP Pareto-optimal and satisfy the three design goals of MPTCP.

In Table XVIII, we observe that the multipath transmission follows an evolutionary way mostly on the transport layer. We believe this is determined by the stack position at which the proposed approaches are located. For example, the fairness, RP, and Pareto-optimality features are achieved by the means of congestion control, which as default is managed on the transport layer. The link layer and application layer cannot intervene congestion control without breaking the protocol stack layered structure. In our literature review, mHIP [151] is the only protocol which provides fairness feature on network layer. However, mHIP is actually located on a middle layer between network and transport layers. Therefore, because of its closeness to transport layer, mHIP is able to manipulate the congestion control operations. The link layer and network layer can provide path diversity for cross-layer approaches. For example, OpenFlow and TRILL can provide Ethernet level path diversity [34, 194], and LISP can provide routing level path diversity [34, 36].

IV. Lessons Learned

From the evolution of end-to-end multipath transmission viewpoint, we observe that multipath has become increasingly popular at the transport layer with features such as load balancing, fairness control, congestion control and Pareto-optimality. As a major extension to TCP which has not changed very much in the last decades, MPTCP has attracted more and more attention in recent years. We refer to Figure 2, Table XVII and Table XVIII to summarize the lessons of its development we have learned from this survey. We believe that they are valuable lessons that others should learn in order to make their proposals into practice.

Table XVIII
RESEARCH PROBLEMS ON EACH LAYER (√ means having tried to address).

Research Problems	Link	Network	Transport	Application
Bandwidth aggregation	√	√	√	√
Packet reordering	√	√	√	√
Fairness		√	√	
RP			√	
Path diversity	√	√		
Pareto-optimality			√	

As shown in Figure 2, MPTCP was proposed at an opportune time to draw the experience gathered in previous work and correcting the past mistakes. For example, MPTCP was designed with the backward compatibility to legacy applications and middleboxes (see Table XVII), which makes MPTCP instead of CMT-SCTP a big step towards being widely acceptable. In addition to solving the compatibility issue, MPTCP goes further in addressing issues in fairness and RP by joint increase and decrease rules (e.g., the coupled congestion control algorithm LIA [162]). Furthermore, OLIA [102] and AOLIA [182] even make MPTCP meet the requirements of Pareto-efficiency (see Table XVIII). Apart from the technical aspects, we believe the following factors also contribute to the success of MPTCP: kernel implementation in Linux, support from Internet standards organization (e.g., IETF), active and public academic community[7], and incentives from industry [1, 4, 20]. According to the previous discussion, we use the following key words to summarize the key efforts which has driven the development of MPTCP: *collecting previous experience, correcting past mistakes, continuously finding and solving new challenges, implementation, standardization, research community and industrial incentives*.

Although MPTCP is a promising protocol for multipath transmission, it still has room for improvement. We give a few recommendations as follows. First, MPTCP needs more "marketing" work to get more support from industry. Currently, only a few companies (e.g., [1, 20, 138]) make products or services on MPTCP. Although Apple implemented MPTCP (in a backup mode) in iOS 7, later iOS upgrades did not support it any more. Secondly, although MPTCP has been implemented in Linux kernel, people have to manually install it (e.g., by the means of apt-repository or compiling and installing from source) before trying it. It would become much convenient if some mainstream Linux distributions can add the MPTCP kernel directly in their releases. Thirdly, much work has been done to improve the performance of MPTCP in terms of packet reordering and Pareto-efficiency. But they are still open questions and need more technical improvement.

V. OPEN QUESTIONS

We have identified a few open questions and present them as follows.

- Multipath transmission in Information-Centric Networking (ICN): in contrast with the host-centric paradigm based on perpetual connectivity and the end-to-end principle, ICN was proposed to make the network content centric allowing nodes to request content that is then delivered by the network to them. In this new networking paradigm, information retrieval is pull-based, driven by user requests, point-to-multipoint and intrinsically coupled with in-network caching. In ICN, a content item can be replicated in more than one node. There is an increased interest in adapting multipath transport control to ICN in the literature [23, 42, 170] so that the delivery of the content can follow more than one path to reach the user. The benefits of multipath in ICN include increased resilience and decreased load to content repositories. The combination of ICN and multipath transport brings new challenges in terms of balancing the performance maximization and network cost minimization. A multipath solution for ICN needs to take into account that the content sources are unknown in advance and may vary over time, and that in-network caching may impact the variability of path length and the associated delivery time.
Currently, there are not many existing multipath transmission approaches suitable for ICN. A strategy layer has been proposed for Content Centric Networks (CCN) that make decisions pertaining to the multipath selection process. Naive multipath strategies have been reported to negatively impact CCN efficiency [170]. An analytical model has been proposed for evaluating different ICN multipath forwarding strategies. According to the model, a good forwarding strategy that maximizes the receive-rate should control the pending interests injected in the different paths so as to fill the capacity of their pipelines [42]. Joint multipath congestion control and request forwarding have been investigated in [23] with the twofold objective of maximizing user throughput and minimizing overall network cost. Relevant research topics for multipath ICN include forwarding strategies for wireless and dynamic networks, joint design of forwarding strategies and cache replacement policies, and routing protocol support for multipath.
- Context-aware scheduling: we refer Table XV to support our discussion on this open question. As shown in the table, no individual algorithm wins from both the packet reordering and load sharing capability. This result implies that there is no single algorithm that is one-size-fits-all. The efficiency of an algorithm depends on how much it fits to the network environment. For example, although an algorithm may drive better results than others in certain network environment, it cannot beat others all the time in a dynamically changing network. Each algorithm has its intended network environ-

[7]MPTCP has an active mail list (mptcp-dev@listes.uclouvain.be) for sharing experience.

ment in which to work the best. Currently, most work has been using one single scheduling algorithm all the time, which would definitely lead to low performance in certain network conditions. Therefore, context-aware scheduling policy is required to dynamically detect the network context and switch to the best performing scheduling algorithm accordingly. The concept of context-aware scheduling is not new. Some initial work has been conducted. For example, WiMP-SCTP [85] has two data transmission scheduling algorithms used for different network conditions. When the network condition is good, the data-striping algorithm is selected to aggregate bandwidth. When the network condition becomes bad, the data-duplicating algorithm is switched on to increase destination reachability. In [68, 166], there are two scheduling algorithms which are used for legacy destination and updated destination respectively. For more similar work, we refer readers to [70, 71, 111].

- Richer API of transport services: if the application were to explicitly control the congestion control algorithms by the means of APIs provided by transport protocols, then it would not only know everything the transport layer knows but also what the application knows (e.g., the workload and application content type), which helps in making optimal decisions. This is one of the goals the TAPS (Transport Services) work group (WG) in IETF plans to achieve. For example, the TAPS WG will identify the services provided by existing IETF transport protocols and congestion control mechanisms as well as network requirements of APIs. The application layer approaches could then use the standard APIs to control mechanism underneath.
- Multicast meets multipath: multicast traffic over the Internet is growing steadily with the increasing number of demanding applications. Many of them require certain QoS guarantees, and demand that the network resource be utilized in an efficient way. To achieve these goals, the multipath transmission could be used to effectively split multicast traffic over multiple paths at the edge of the multicast tree. For example, the future of mobile content delivery may use multicast networks for audio/video streaming applications. The last hop of the multicast networks would be based on various wireless technologies (e.g., WiFi, 3G, and LTE). Due to the fact that an individual wireless path may be unreliable and be unable to provide required bandwidth, multipath transmission for the last hop would improve its resilience and throughput. Generally, the challenges include packet reordering and in-network caching issues. We believe that packet coding schemes [31, 63, 107, 180] can be potential solutions to them.
- Heterogeneity of multipath: there are many different network environments for multipath transmission. For example, in modern data centers, there are usually more than one path available between any pair of endpoints. The same applies to access networks, the core of the Internet, and Internet Service Providers (ISPs). Nevertheless, these domains are usually autonomous and isolated from each other. There are specialized network entities, e.g., border gateway, to guarantee the autonomy of each domain. The downside of this network topology is that even though there might be abundant multipath resources available locally within each domain, the globally available multi-path resources are limited to the border gateways. Thus, how to break the border of each autonomous domain to enrich the multipath resources significantly is not only a technical problem, but also a management problem. It requires efforts from multiple domains, including ISPs, policymakers and end users.
- Specialized use of multipath transmission: most of the existing approaches aim to obtain higher throughput from the use of multipath transmission. However, from a user's perspective, boosting the throughput of a multi-homed mobile device may not be the first-priority goal all the time. Instead, some users may be willing to use cheap subscriptions in order to save a few dollars. Others would rather use the low energy-consumption interface(s) to save the battery life. Therefore, multipath transmission strategies which take price and energy into consideration should be provided to users so that they can choose from different transmission strategies to satisfy their demands.

VI. Conclusion

Conventional TCP/IP always uses a single "best" path according to certain routing metrics, even if there may be more than one path between two endpoints. This behavior results in under-utilization of the available network resources. The proliferation of mobile devices equipped with multiple interfaces, represented by smart phones, brings with it a growing number of multi-homed hosts onto the Internet. Thus, this deteriorates the mismatch between single-path transport and the multitude of available network paths. Multipath transmission comes into the picture as a natural solution with several salient features, such as reliability, fairness, RP and Pareto-optimality. In this article, we make a comprehensive survey about the state-of-the-art multipath transmission approaches, intending to provide researchers and practitioners with insightful observations. We hope this survey will inspire a series of new research work in this field.

Acknowledgment

This work was supported by the Academy of Finland, grant number 278207, and HIIT Distributed and Mobile Cloud Systems research programme.

References

[1] Hybrid Internet Access Bonding. http://www.tessares.net [Available Online].

[2] IEEE Standard for Local and Metropolitan Area Networks - Link Aggregation. *IEEE Std 802.1AX-2008*, pages c1–145, 2008.

[3] Ieee standard for local and metropolitan area networks–media access control (mac) bridges and virtual bridged local area networks–amendment 20: Shortest path bridging. *IEEE Std 802.1aq-2012*, pages 1–340, June 2012.

[4] iOS: Multipath TCP Support in iOS 7. 2015. https://support.apple.com/en-us/HT201373 [Available Online].

[5] A. Abd, T. Saadawi, and M. Lee. Improving throughput and reliability in mobile wireless networks via transport layer bandwidth aggregation. *Computer Networks*, 46(5):635–649, 2004.

[6] A. Abd El Al, T. Saadawi, and M. Lee. LS-SCTP: a bandwidth aggregation technique for stream control transmission protocol. *Computer Communications*, 27(10):1012–1024, 2004.

[7] S. Addepalli, H. G. Schulzrinne, A. Singh, and G. Ormazabal. Heterogeneous Access: Survey and Design Considerations. *Columbia University Academic Commons, Technical Report*, 2013. http://dx.doi.org/10.7916/D8QJ7F8P.

[8] H. Adhari, T. Dreibholz, M. Becke, E. P. Rathgeb, and M. Tüxen. Evaluation of concurrent multipath transfer over dissimilar paths. In *IEEE Workshops of International Conference on Advanced Information Networking and Applications (WAINA '11)*, pages 708–714, 2011.

[9] H. Adiseshu, G. Parulkar, and G. Varghese. A reliable and scalable striping protocol. In *Proceedings of the Applications, technologies, architectures, and protocols for computer communications*, volume 26, pages 131–141. ACM, 1996.

[10] M. Allman, H. Kruse, and S. Ostermann. An application-level solution to TCP's satellite inefficiencies. In *Proceedings of the 1st International Workshop on Satellite-based Information Services (WOSBIS '96)*, 1996.

[11] P. Amer, M. Becke, T. Dreibholz, N. Ekiz, J. Iyengar, P. Natarajan, R. Stewart, and M. Tuexen. Load Sharing for the Stream Control Transmission Protocol (SCTP). *draft-tuexen-tsvwg-sctp-multipath-10 (Experimental), Internet-Draft, IETF*, November 2015.

[12] P. D. Amer, C. Chassot, T. J. Connolly, M. Diaz, and P. Conrad. Partial-order transport service for multimedia and other applications. *IEEE/ACM Transactions on Networking (TON)*, 2(5):440–456, 1994.

[13] A. Argyriou and V. Madisetti. Bandwidth aggregation with SCTP. In *Proceedings of the 2003 IEEE Global Telecommunications Conference (GLOBECOM '03)*, volume 7, pages 3716–3721. IEEE, 2003.

[14] B. Arzani, A. Gurney, S. Cheng, R. Guerin, and B. T. Loo. Impact of Path Characteristics and Scheduling Policies on MPTCP Performance. In *Proceedings of the 28th International Conference on Advanced Information Networking and Applications Workshops (WAINA)*, pages 743–748, May 2014.

[15] AVAYA. Multi-Link Trunking. 2011. https://downloads.avaya.com/css/P8/documents/100134063 [Available Online].

[16] A. Baldini, L. De Carli, and F. Risso. Increasing performances of TCP data transfers through multiple parallel connections. In *Proceedings of IEEE Symposium on Computers and Communications (ISCC)*, pages 630–636, July 2009.

[17] S. Barré, C. Paasch, and O. Bonaventure. MultiPath TCP: From Theory to Practice. In *Proceedings of the 10th International IFIP TC 6 Conference on Networking - Volume Part I*, NETWORKING '11, pages 444–457. Springer-Verlag, 2011.

[18] M. Becke, H. Adhari, E. P. Rathgeb, F. Fa, X. Yang, and X. Zhou. Comparison of Multipath TCP and CMT-SCTP based on Intercontinental Measurements. In *Proceedings of the 2013 IEEE Global Communications Conference (GLOBECOM '13)*, pages 1360–1366. IEEE, 2013.

[19] E. Blanton and M. Allman. On making TCP more robust to packet reordering. *ACM SIGCOMM Computer Communication Review*, 32(1):20–30, 2002.

[20] O. Bonaventure. In Korean, Multipath TCP is pronounced GIGA Path. 2015. http://blog.multipath-tcp.org/blog/html/2015/07/24/korea.html [Available Online].

[21] L. Budzisz, R. Ferrús, F. Casadevall, and P. Amer. On concurrent multipath transfer in SCTP-based handover scenarios. In *Proceedings of the IEEE International Conference on Communications (ICC '09)*, pages 1–6, 2009.

[22] D. Bushmitch, S. Panwar, and A. Pal. Thinning, striping and shuffling: traffic shaping and transport techniques for variable bit rate video. In *Proceedings of the 2002 IEEE Global Telecommunications Conference (GLOBECOM '02)*, volume 2, pages 1485–1491, Nov 2002.

[23] G. Carofiglio, M. Gallo, L. Muscariello, M. Papalini, and S. Wang. Optimal multipath congestion control and request forwarding in information-centric networks. In *Proceedings of the 21st IEEE International Conference on Network Protocols (ICNP '13)*, pages 1–10. IEEE, 2013.

[24] C. Casetti and W. Gaiotto. Westwood SCTP: load balancing over multipaths using bandwidth-aware source scheduling. In *Proceedings of the 2004 Vehicular Technology Conference (VTC '04)*, volume 4, pages 3025–3029. IEEE, 2004.

[25] C. Cetinkaya and E. W. Knightly. Opportunistic traffic scheduling over multiple network paths. In *Proceedings of the 23td Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '04)*, volume 3, pages 1928–1937. IEEE, 2004.

[26] K. Chebrolu, B. Raman, and R. R. Rao. A network layer approach to enable TCP over multiple interfaces. *Wireless Networks*, 11(5):637–650, 2005.

[27] K. Chebrolu and R. R. Rao. Bandwidth aggregation for real-time applications in heterogeneous wireless networks. *IEEE Transactions on Mobile Computing*, 5(4):388–403, 2006.

[28] J. Chen, K. Xu, and M. Gerla. Multipath tcp in lossy wireless environment. In *Proceedings of IFIP 3rd Annual Mediterranean Ad Hoc Networking Workshop (Med-Hoc-Net '04)*, pages 263–270, 2004.

[29] X. Chen, A. Jukan, A. Drummond, and N. da Fonseca. A multipath routing mechanism in optical networks with extremely high bandwidth requests. In *Global Telecommunications Conference, 2009. GLOBECOM 2009. IEEE*, pages 1–6, Nov 2009.

[30] Y.-C. Chen, Y. Lim, R. J. Gibbens, E. M. Nahum, R. Khalili, and D. Towsley. A Measurement-based Study of Multipath TCP Performance over Wireless Networks. In *Proceedings of the 2013 ACM Conference on Internet Measurement Conference (IMC '13)*, pages 455–468, 2013.

[31] P. A. Chou, Y. Wu, and K. Jain. Practical network coding. In *Proceedings of the Allerton Conference on Communication, Control, and Computing*, 2003.

[32] Cisco. EtherChannels. 2003. http://www.cisco.com/c/en/us/td/docs/switches/lan/catalyst3550/software/release/12-1_13_ea1/configuration/guide/3550scg/swethchl.html [Available Online].

[33] T. Consortium et al. Openflow switch specification/version 1.1. 0, 2011.

[34] M. Coudron, S. Secci, G. Maier, G. Pujolle, and A. Pattavina. Boosting cloud communications through a cross-layer multipath protocol architecture. In *Proceedings of the IEEE SDN for Future Networks and Services (SDN4FNS)*, pages 1–8. IEEE, 2013.

[35] M. Coudron, S. Secci, and G. Pujolle. Differentiated pacing on multiple paths to improve one-way delay estimations. In *Proceedings of IFIP/IEEE International Symposium on Integrated Network Management (IM)*, pages 672–678, May 2015.

[36] M. Coudron, S. Secci, G. Pujolle, P. Raad, and P. Gallard. Cross-layer cooperation to boost multipath TCP performance in cloud networks. In *Proceedings of the 2013 IEEE 2nd International Conference on Cloud Networking (CloudNet '13)*, pages 58–66, Nov 2013.

[37] Y. Cui, L. Wang, X. Wang, H. Wang, and Y. Wang. FMTCP: A Fountain Code-Based Multipath Transmission Control Protocol. *IEEE/ACM Transactions on Networking (TON)*, January 2014.

[38] Y. Cui, X. Wang, H. Wang, G. Pan, and Y. Wang. FMTCP: A fountain code-based multipath transmission control protocol. In *Proceedings of the 32nd IEEE International Conference on Distributed Computing Systems (ICDCS '12)*, pages 366–375, 2012.

[39] M. Cullen, N. Leymann, C. Heidemann, M. Boucadair, H. Deng, and B. Sarikaya. Problem Statement: Bandwidth Aggregation for Internet Access. *draft-zhang-banana-problem-statement-01 (Informational), Internet-Draft, IETF*, November 2015.

[40] T. Das and K. M. Sivalingam. TCP improvements for data center networks. In *Proceedings of the 5th International Conference on Communication Systems and Networks (COMSNETS '13)*, pages 1–10. IEEE, 2013.

[41] G. Detal, C. Paasch, and O. Bonaventure. Multipath in the Middle(Box). In *Proceedings of the 2013 Workshop on Hot Topics in Middleboxes and Network Function Virtualization (HotMiddlebox '13)*, pages 1–6, 2013.

[42] A. Detti, C. Pisa, and N. Blefari Melazzi. Modeling multipath forwarding strategies in Information Centric Networks. In *Proceedings of IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pages 324–329, April 2015.

[43] Deutsche Telekom. Hybrid Access - a promising approach to increase bandwidth of DSL-lines. 2014. http://www.laboratories.telekom.com/public/english/newsroom/news/pages/hybrid-access.aspx [Available Online].

[44] C. Diop, G. Dugue, C. Chassot, and E. Exposito. QoS-oriented MPTCP Extensions for Multimedia Multi-homed Systems. In *Proceedings of the 26th International Conference on Advanced Information Networking and Applications Workshops (WAINA '12)*, pages 1119–1124, March 2012.

[45] J. Domżał, Z. Duliński, M. Kantor, J. Rzasa, R. Stankiewicz, K. Wajda, and R. Wójcik. A survey on methods to provide multipath transmission in wired packet networks. *Computer Networks*, 77:18–41, 2015.

[46] Y. Dong, N. Pissinou, and J. Wang. Concurrency Handling in TCP. In *Proceedings of the 5th Annual Conference on Communication Networks and Services Research (CNSR '07)*, pages 255–262, May 2007.

[47] T. Dreibholz, M. Becke, H. Adhari, and E. P. Rathgeb. On the impact of congestion control for Concurrent Multipath Transfer on the transport layer. In *Proceedings of the 11th IEEE International Conference on Telecommunications (ConTEL '11)*, pages 397–404, 2011.

[48] T. Dreibholz, M. Becke, J. Pulinthanath, and E. P. Rathgeb. Applying TCP-friendly congestion control to Concurrent Multipath Transfer. In *Proceedings of the 24th IEEE International Conference on Advanced Information Networking and Applications (AINA '10)*, pages 312–319. IEEE, 2010.

[49] T. Dreibholz, M. Becke, E. P. Rathgeb, and M. Tuxen. On the use of concurrent multipath transfer over asymmetric paths. In *Proceedings of the IEEE Global Telecommunications Conference (GLOBECOM '10)*, pages 1–6, 2010.

[50] T. Dreibholz, R. Seggelmann, M. Tuexen, and E. Rathgeb. Transmission scheduling optimizations for concurrent multipath transfer. In *Proceedings of the 8th International Workshop on Protocols for Future, Large-Scale and Diverse Network Transports (PFLDNeT '10)*, volume 8, 2010.

[51] D. Eastlake, M. Zhang, A. Ghanwani, V. Manral, and A. Banerjee. Transparent Interconnection of Lots of Links (TRILL): Clarifications, Corrections, and Updates. *IETF RFC 7180, May*, 2014.

[52] K. Evensen, D. Kaspar, P. Engelstad, A. F. Hansen, C. Griwodz, and P. Halvorsen. A network-layer proxy for bandwidth aggregation and reduction of IP packet reordering. In *Proceedings of the IEEE 34th Conference on Local Computer Networks (LCN)*, pages 585–592. IEEE, 2009.

[53] K. Evensen, D. Kaspar, C. Griwodz, P. Halvorsen, A. Hansen, and P. Engelstad. Improving the Performance of Quality-Adaptive Video Streaming over Multiple Heterogeneous Access Networks. In *Proceedings of the 2nd annual ACM conference on Multimedia systems*, pages 57–68. ACM, 2011.

[54] K. Evensen, D. Kaspar, C. Griwodz, P. Halvorsen, A. F. Hansen, and P. Engelstad. Using bandwidth aggregation to improve the performance of quality-adaptive streaming. *Signal Processing: Image Communication*, 27(4):312–328, 2012.

[55] K. Evensen, T. Kupka, D. Kaspar, P. Halvorsen, and C. Griwodz. Quality-adaptive scheduling for live streaming over multiple access networks. In *Proceedings of the 20th international workshop on Network and operating systems support for digital audio and video*, pages 21–26. ACM, 2010.

[56] D. Farinacci, D. Lewis, D. Meyer, and V. Fuller. The locator/ID separation protocol (LISP). *IETF RFC 6830*, February 2013.

[57] D. Fedyk, P. Ashwood-Smith, D. Allan, A. Bragg, and P. Unbehagen. IS-IS Extensions Supporting IEEE 802.1 aq Shortest Path Bridging. *IETF RFC 6329*, 2012.

[58] S. Ferlin, T. Dreibholz, and O. Alay. Multi-path transport over heterogeneous wireless networks: Does it really pay off? In *Proceedings of IEEE Global Communications Conference (GLOBECOM '14)*, pages 4807–4813, Dec 2014.

[59] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee. Hypertext transfer protocol–HTTP/1.1, June 1999.

[60] A. Ford, C. Raiciu, M. Handley, and S. Barre. TCP Extensions for Multipath Operation with Multiple Addresses. *IETF Internet-Draft*, May 2009. https://tools.ietf.org/html/draft-ford-mptcp-multiaddressed-00.

[61] A. Ford, C. Raiciu, M. Handley, S. Barré, and J. Iyengar. Architectural guidelines for multipath TCP development. *IETF RFC 6182*, March 2011.

[62] A. Ford, C. Raiciu, M. Handley, and O. Bonaventure. TCP Extensions for Multipath Operation with Multiple Addresses. *IETF RFC 6824*, January 2013.

[63] C. Fragouli, D. Lun, M. Médard, and P. Pakzad. On feedback for network coding. In *Proceedings of the 41st Annual Conference on Information Sciences and Systems (CISS '07)*, pages 248–252. IEEE, 2007.

[64] L. Golubchik, J. C. S. Lui, T. F. Tung, A. L. Chow, W.-J. Lee, G. Franceschinis, and C. Anglano. Multi-path continuous media streaming: What are the benefits? *Performance Evaluation*, 49(1):429–449, 2002.

[65] A. Gurtov and T. Polishchuk. Secure multipath transport for legacy Internet applications. In *Proceedings of the 6th International Conference on Broadband Communications, Networks, and Systems*, pages 1–8, 2009.

[66] K. Habak, K. A. Harras, and M. Youssef. OPERETTA: An optimal energy efficient bandwidth aggregation system. In *Proceedings of the 9th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON '12)*, pages 121–129. IEEE, 2012.

[67] K. Habak, K. A. Harras, and M. Youssef. Bandwidth aggregation techniques in heterogeneous multi-homed devices: A survey. *arXiv preprint arXiv:1309.0542*, 2013.

[68] K. Habak, K. A. Harras, and M. Youssef. OSCAR: A Collaborative Bandwidth Aggregation System. *arXiv preprint arXiv:1401.1258*, 2014.

[69] K. Habak, M. Youssef, and K. Harras. G-DBAS: A Green and Deployable Bandwidth Aggregation System. In *Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC '12)*, pages 3290–3295, 2012.

[70] K. Habak, M. Youssef, and K. A. Harras. DBAS: A Deployable Bandwidth Aggregation System. In *Proceedings of the 5th International Conference on New Technologies, Mobility and Security (NTMS '12)*, pages 1–6. IEEE, 2012.

[71] K. Habak, M. Youssef, and K. A. Harras. An optimal deployable bandwidth aggregation system. *Computer Networks*, 57(15):3067–3080, 2013.

[72] T. J. Hacker, B. D. Athey, and B. Noble. The end-to-end performance effects of parallel TCP sockets on a lossy wide-area network. In *Proceedings of the International Parallel and Distributed Processing Symposium (IPDPS '02)*, pages 434–443. IEEE, 2002.

[73] A. Hari, G. Varghese, and G. Parulkar. An Architecture for Packet-striping Protocols. *ACM Transactions on Computer Systems (TOCS)*, 17(4):249–287, November 1999.

[74] Y. Hasegawa, I. Yamaguchi, T. Hama, H. Shimonishi, and T. Murase. Improved data distribution for multipath TCP communication. In *Proceedings of the IEEE Global Telecommunications Conference (GLOBECOM '05)*, November 2005.

[75] Y. Hasegawa, I. Yamaguchi, T. Hama, H. Shimonishi, and T. Murase. Deployable multipath communication scheme with sufficient performance data distribution method. *Computer Communications*, 30(17):3285–3292, 2007.

[76] S. Hassayoun, J. Iyengar, and D. Ros. Dynamic window coupling for multipath congestion control. In *Proceedings of the 19th IEEE International Conference on Network Protocols (ICNP '11)*, pages 341–352. IEEE, 2011.

[77] T. Henderson, S. Floyd, A. Gurtov, and Y. Nishida. The NewReno modification to TCP's fast recovery algorithm. *IETF RFC 6582*, 2012.

[78] B. Hesmans, F. Duchene, C. Paasch, G. Detal, and O. Bonaventure. Are TCP extensions middlebox-proof? In *Proceedings of the 2013 workshop on Hot topics in middleboxes and network function virtualization*, pages 37–42, 2013.

[79] M. Honda, Y. Nishida, L. Eggert, P. Sarolahti, and H. Tokuda. Multipath congestion control for shared bottleneck. In *Proceedings of Workshop on Protocols for Fast Long-Distance Networks (PFLDNeT 09 ')*, pages 19–24, 2009.

[80] M. Honda, Y. Nishida, C. Raiciu, A. Greenhalgh, M. Handley, and H. Tokuda. Is it still possible to extend TCP? In *Proceedings of the 2011 ACM SIGCOMM conference on Internet measurement conference*, pages 181–194. ACM, 2011.

[81] H.-Y. Hsieh, K.-H. Kim, Y. Zhu, and R. Sivakumar. A

Receiver-centric Transport Protocol for Mobile Hosts with Heterogeneous Wireless Interfaces. In *Proceedings of the 9th Annual International Conference on Mobile Computing and Networking (MobiCom '03)*, pages 1–15, 2003.

[82] H.-Y. Hsieh and R. Sivakumar. A Transport Layer Approach for Achieving Aggregate Bandwidths on Multi-homed Mobile Hosts. In *Proceedings of the 8th Annual International Conference on Mobile Computing and Networking (MobiCom '02)*, pages 83–94. ACM, 2002.

[83] H.-Y. Hsieh and R. Sivakumar. pTCP: An end-to-end transport layer protocol for striped connections. In *Proceedings of the 10th IEEE International Conference on Network Protocols*, pages 24–33, 2002.

[84] H.-Y. Hsieh and R. Sivakumar. A transport layer approach for achieving aggregate bandwidths on multi-homed mobile hosts. *Wireless Networks*, 11(1-2):99–114, 2005.

[85] C.-M. Huang and C.-H. Tsai. WiMP-SCTP: Multi-path transmission using stream control transmission protocol (SCTP) in wireless networks. In *Proceedings of the 21st IEEE International Conference on Advanced Information Networking and Applications Workshops (AINAW '07)*, volume 1, pages 209–214, 2007.

[86] X. Huang and Y. Fang. Multiconstrained QoS multipath routing in wireless sensor networks. *Wireless Networks*, 14(4):465–478, 2008.

[87] C. Huitema. Multi-homed TCP. *IETF Internet-Draft (Expired)*, May 1995.

[88] IEEE. IEEE 802.ad Link Aggregation Task Force. 2000. http://www.ieee802.org/3/ad/ [Available Online].

[89] J. Iyengar, K. Shah, P. Amer, and R. Stewart. Concurrent multipath transfer using SCTP multihoming. In *Proceedings of the International Symposium on Performance Evaluation of Computer and Telecommunication Systems (SPECTS '04)*, 2004.

[90] J. R. Iyengar, P. D. Amer, and R. Stewart. Retransmission policies for concurrent multipath transfer using SCTP multihoming. In *Proceedings of the 12th IEEE International Conference on Networks (ICON '04)*, volume 2, pages 713–719. IEEE, 2004.

[91] 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*, 14(5):951–964, 2006.

[92] Jason Lee and Dan Gunter and Brian Tierney and Bill Allcock and Joe Bester and John Bresnahan and Steve Tuecke. Applied Techniques for High Bandwidth Data Transfers across Wide Area Networks. In *Proceedings of the International Conference on Computing in High Energy and Nuclear Physics*, 2001.

[93] P. Jokela. Using the encapsulating security payload (ESP) transport format with the host identity protocol (HIP). *IETF RFC 5202*, April 2008.

[94] Juniper. Aggregated Ethernet. 2014. http://www.juniper.net/documentation/en_US/junos14.1/topics/task/configuration/link-aggregation-cli.html [Available Online].

[95] S. Kandula, K. C.-J. Lin, T. Badirkhanli, and D. Katabi. FatVAP: Aggregating AP Backhaul Capacity to Maximize Throughput. In *Proceedings of the 5th USENIX Symposium on Networked Systems Design and Implementation*, volume 8, pages 89–104, 2008.

[96] D. Kaspar, K. Evensen, P. Engelstad, and A. F. Hansen. Using HTTP pipelining to improve progressive download over multiple heterogeneous interfaces. In *Proceedings of the IEEE International Conference on Communications (ICC '10)*, pages 1–5. IEEE, 2010.

[97] D. Kaspar, K. Evensen, P. Engelstad, A. F. Hansen, P. Halvorsen, and C. Griwodz. Enhancing video-on-demand playout over multiple heterogeneous access networks. In *Proceedings of the 7th IEEE Consumer Communications and Networking Conference (CCNC '10)*, pages 1–5. IEEE, 2010.

[98] V. Kawadia and P. R. Kumar. A cautionary perspective on cross-layer design. *IEEE Wireless Communications*, 12(1):3–11, 2005.

[99] S. Keshav. A control-theoretic approach to flow control. *ACM SIGCOMM Computer Communication Review*, 25(1):188–201, 1995.

[100] P. Key, L. Massoulié, and D. Towsley. Combining multipath routing and congestion control for robustness. In *Proceedings of the 40th Annual Conference on Information Sciences and Systems*, pages 345–350. IEEE, 2006.

[101] R. Khalili, N. Gast, M. Popovic, and J.-Y. Le Boudec. MPTCP is Not Pareto-optimal: Performance Issues and a Possible Solution. *IEEE/ACM Transactions on Networking (TON)*, 21(5):1651–1665, October 2013.

[102] R. Khalili, N. Gast, M. Popovic, U. Upadhyay, and J.-Y. Le Boudec. MPTCP is Not Pareto-optimal: Performance Issues and a Possible Solution. In *Proceedings of the 8th International Conference on Emerging Networking Experiments and Technologies (CoNEXT '12)*, pages 1–12. ACM, 2012.

[103] H. A. Kim, B. hwan Oh, and J. Lee. Improvement of MPTCP Performance in heterogeneous network using packet scheduling mechanism. In *Proceedings of the 18th Asia-Pacific Conference on Communications (APCC '12)*, pages 842–847, October 2012.

[104] K.-H. Kim and K. G. Shin. Improving TCP performance over wireless networks with collaborative multi-homed mobile hosts. In *Proceedings of the 3rd international conference on Mobile systems, applications, and services*, pages 107–120. ACM, 2005.

[105] K.-H. Kim and K. G. Shin. PRISM: improving the performance of inverse-multiplexed TCP in wireless networks. *IEEE Transactions on Mobile Computing*, 6(12):1297–1312, 2007.

[106] K.-H. Kim, Y. Zhu, R. Sivakumar, and H.-Y. Hsieh. A receiver-centric transport protocol for mobile hosts with heterogeneous wireless interfaces. *Wireless Networks*, 11(4):363–382, 2005.

[107] R. Koetter and M. Médard. An algebraic approach to network coding. *IEEE/ACM Transactions on Networking*, 11(5):782–795, 2003.

[108] M. Kun, Y. Jingdong, and R. Zhi. The research and simulation of multipath-OLSR for mobile ad hoc network. In *IEEE International Symposium on Communications and Information Technology (ISCIT)*, volume 1, pages 540–543. IEEE, 2005.

[109] K.-c. Lan and C.-Y. Li. Improving TCP performance over an on-board multi-homed network. In *IEEE Wireless Communications and Networking Conference (WCNC)*, pages 2961–2966. IEEE, 2012.

[110] T.-A. Le and L. X. Bui. Forward Delay-based Packet Scheduling Algorithm for Multipath TCP. *arXiv preprint arXiv:1501.03196*, 2015.

[111] Y. Lee, I. Park, and Y. Choi. Improving TCP performance in multipath packet forwarding networks. *Journal of Communications and Networks*, 4:148–157, 2002.

[112] W. Lei, W. Zhang, and S. Liu. A Framework of Multipath Transport System Based on Application-Level Relay (MPTS-AR). *IETF Internet-Draft (Experimental), July*, 2015.

[113] K. Leonard. An early history of the Internet . *IEEE Communications Magazine*, 48(8):26–36, 2010.

[114] M. Li, L. Andrey, and Y. Cui. Network coding based multipath TCP. In *2012 IEEE Conference on Computer Communications Workshops (INFOCOM workshop)*, pages 25–30, March 2012.

[115] M. Li, L. Andrey, S. Tarkoma, Y. Cui, and A. Ylä-Jääski. Tolerating path heterogeneity in multipath TCP with bounded receive buffers. *Computer Networks*, 64:1–14, 2014.

[116] M. Li, L. Andrey, S. Tarkoma, and A. Ylä-Jääski. The Delayed ACK evolution in MPTCP. In *Global Communications Conference (GLOBECOM), 2013 IEEE*, pages 2282–2288. IEEE, 2013.

[117] M. Li, L. Andrey, S. Tarkoma, and A. Ylä-Jääski. MPTCP incast in data center networks. *Communications, China*, 11(4):25–37, 2014.

[118] J. Liao, J. Wang, and X. Zhu. cmpSCTP: An extension of SCTP to support concurrent multi-path transfer. In *Proceedings of the IEEE International Conference on Communications (ICC '08)*, pages 5762–5766, 2008.

[119] Y.-s. Lim, Y.-C. Chen, E. M. Nahum, D. Towsley, and K.-W. Lee. Cross-layer path management in multi-path transport protocol for mobile devices. In *Proceedings of the Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, pages 1815–1823. IEEE, 2014.

[120] J. Liu, H. Zou, J. Dou, and Y. Gao. Rethinking Retransmission Policy In Concurrent Multipath Transfer. In *Proceedings of the International Conference on Intelligent Information Hiding and Multimedia Signal Processing (IIHMSP '08)*, pages 1005–1008. IEEE, 2008.

[121] L. Magalhaes and R. Kravets. Transport level mechanisms for bandwidth aggregation on mobile hosts. In *Proceedings of the 9th IEEE International Conference on Network Protocols*, pages 165–171, 2001.

[122] K. Manousakis and D. Famolari. INTELiCON: A Framework for the Simultaneous Utilization of Multiple Interfaces and its Application on TCP. In *International Wireless Communications and Mobile Computing Conference, 2008 (IWCMC '08)*, pages 976–981, Aug 2008.

[123] S. Mao, D. Bushmitch, S. Narayanan, and S. S. Panwar. MRTP: a multiflow real-time transport protocol for ad hoc networks. *IEEE Transactions on Multimedia*, 8(2):356–369, 2006.

[124] S. Mascolo, L. A. Grieco, R. Ferorelli, P. Camarda, and G. Piscitelli. Performance evaluation of Westwood+ TCP congestion control. *Performance Evaluation*, 55(1):93–111, 2004.

[125] M. Mathis, J. Mahdavi, S. Floyd, and A. Romanow. TCP selective acknowledgment options. Technical report, October 1996.

[126] N. F. Maxemchuk. *DISPERSITY ROUTING IN STORE-AND-FORWARD NETWORKS*. PhD thesis, University of Pennsylvania, 1975. Available as http://repository.upenn.edu/dissertations/AAI7524101.

[127] M. Menth, A. Stockmayer, and M. Schmidt. LISP Hybrid Access. *draft-menth-lisp-ha-00 (Experimental), Internet-Draft, IETF*, July 2015.

[128] J. Milbrandt, K. Humm, and M. Menth. Adaptive bandwidth allocation: impact of routing and load balancing on tunnel capacity requirements. In *Next Generation Internet Design and Engineering (NGI '06)*, pages 8–pp. IEEE, 2006.

[129] F. H. Mirani, N. Boukhatem, and M. A. Tran. A data-scheduling mechanism for multi-homed mobile terminals with disparate link latencies. In *Proceedings of the 72nd IEEE Vehicular Technology Conference Fall (VTC '10)*, pages 1–5. IEEE, 2010.

[130] E. Miyazaki and M. Oguchi. Evaluation of Middleware for Bandwidth Aggregation using Multiple Interface in Wireless Communication. *International Journal On Advances in Networks and Services*, 4(3 and 4):343–352, 2012.

[131] R. Moskowitz, P. Nikander, P. Jokela, and T. Henderson. Host identity protocol. *IETF RFC 5201*, April 2008.

[132] D. Nagle, D. Serenyi, and A. Matthews. The Panasas ActiveScale storage cluster: Delivering scalable high bandwidth storage. In *Proceedings of the ACM/IEEE conference on Supercomputing*, page 53. IEEE Computer Society, 2004.

[133] S. C. Nguyen and T. M. T. Nguyen. Evaluation of multipath TCP load sharing with coupled congestion control option in heterogeneous networks. In *Proceedings of the Global Information Infrastructure Symposium (GIIS '11)*, pages 1–5, 2011.

[134] S. C. Nguyen, X. Zhang, T. M. T. Nguyen, and G. Pujolle. Evaluation of throughput optimization and load sharing of multipath tcp in heterogeneous networks. In *Proceedings of the 8th International Conference on Wireless and Optical Communications Networks (WOCN '11)*, pages 1–5. IEEE, 2011.

[135] T. Nguyen-Duc, H. Tran-Viet, K. Nguyen, Q. T. Minh, S. H. Ngo, and S. Yamada. Investigating the Performance of Link Aggregation on OpenFlow Switches. In *Testbeds and Research Infrastructure: Development of*

Networks and Communities, pages 194–202. Springer, 2014.

[136] C. Nicutar, C. Paasch, M. Bagnulo, and C. Raiciu. Evolving the Internet with connection acrobatics. In *Proceedings of the 2013 workshop on Hot topics in middleboxes and network function virtualization*, pages 7–12. ACM, 2013.

[137] E. Nordmark and M. Bagnulo. Shim6: Level 3 multihoming shim protocol for IPv6. *IETF RFC 5533*, June 2009.

[138] OVH company. Overthebox. 2015. https://www.ovhtelecom.fr/overthebox [Available Online].

[139] C. Paasch, S. Ferlin, O. Alay, and O. Bonaventure. Experimental evaluation of multipath TCP schedulers. In *Proceedings of the 2014 ACM SIGCOMM workshop on Capacity sharing workshop*, pages 27–32. ACM, 2014.

[140] C. Paasch, R. Khalili, and O. Bonaventure. On the Benefits of Applying Experimental Design to Improve Multipath TCP. In *Proceedings of the 9th ACM Conference on Emerging Networking Experiments and Technologies (CoNEXT '13)*, pages 393–398, 2013.

[141] A. Pathak, H. Pucha, Y. Zhang, Y. Hu, and Z. Mao. A Measurement Study of Internet Delay Asymmetry. In *Passive and Active Network Measurement*, volume 4979 of *Lecture Notes in Computer Science*, pages 182–191. Springer, 2008.

[142] Q. Peng, A. Walid, and S. H. Low. Multipath TCP Algorithms: Theory and Design. *ACM SIGMETRICS Performance Evaluation Review*, 41(1):305–316, 2013.

[143] Q. Peng, A. Walid, and S. H. Low. Multipath TCP: Analysis, Design and Implementation. *CoRR*, abs/1308.3119, 2013.

[144] C. Perkins. IP Encapsulation within IP. *IETF RFC 2003*, October 1996.

[145] C. Perkins. IP Mobility Support for IPv4, Revised. *IETF RFC 5944*, November 2010.

[146] C. E. Perkins. Mobile IP. *IEEE Communications Magazine*, 35(5):84–99, 1997.

[147] A. Phanishayee, E. Krevat, V. Vasudevan, D. G. Andersen, G. R. Ganger, G. A. Gibson, and S. Seshan. Measurement and Analysis of TCP Throughput Collapse in Cluster-based Storage Systems. In *Proceedings of the 6th USENIX Conference on File and Storage Technologies (FAST '08)*, volume 8, pages 1–14, 2008.

[148] D. S. Phatak and T. Goff. A novel mechanism for data streaming across multiple IP links for improving throughput and reliability in mobile environments. In *Proceedings of the 21st Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '02)*, volume 2, pages 773–781, 2002.

[149] D. S. Phatak, T. Goff, and J. Plusquellic. IP-in-IP tunneling to enable the simultaneous use of multiple IP interfaces for network level connection striping. *Computer Networks*, 43(6):787–804, 2003.

[150] S. Pierrel, P. Jokela, and J. Melen. Simultaneous Multi-Access extension to the Host Identity Protocol. *draft-pierrel-hip-sima-00 (work in process), Internet-Draft, IETF*, June 2006.

[151] T. Polishchuk and A. Gurtov. Improving TCP-friendliness and Fairness for mHIP. *Inforcommunications Journal*, 3(1), 2011.

[152] J. Postel and J. Reynolds. File transfer protocol. *IETF RFC 959*, October 1985.

[153] S. Prabhavat, H. Nishiyama, N. Ansari, and N. Kato. On load distribution over multipath networks. *IEEE Communications Surveys & Tutorials*, 14(3):662–680, 2012.

[154] J. Qadir, A. Ali, Y. Kok-Lim, A. Sathiaseelan, and J. Crowcroft. Exploiting the power of multiplicity: a holistic survey of network-layer multipath. *IEEE Communications Surveys Tutorials*, 2015.

[155] A. Qureshi, J. Carlisle, and J. Guttag. Tavarua: Video Streaming with WWAN Striping. In *Proceedings of the 14th annual ACM international conference on Multimedia*, pages 327–336. ACM, 2006.

[156] M. Radi, B. Dezfouli, K. A. Bakar, and M. Lee. Multipath routing in wireless sensor networks: survey and research challenges. *Sensors*, 12(1):650–685, 2012.

[157] C. Raiciu, S. Barré, C. Pluntke, A. Greenhalgh, D. Wischik, and M. Handley. Improving datacenter performance and robustness with multipath TCP. In *Proceedings of the ACM SIGCOMM conference*, volume 41, pages 266–277, 2011.

[158] C. Raiciu, M. Handley, and D. Wischik. Coupled Multipath-Aware Congestion Control. *IETF Internet-Draft*, October 2009. https://tools.ietf.org/html/draft-raiciu-mptcp-congestion-00.

[159] C. Raiciu, M. Handley, and D. Wischik. Coupled congestion control for multipath transport protocols. *IETF RFC 6356*, October 2011.

[160] C. Raiciu, C. Paasch, S. Barré, A. Ford, M. Honda, F. Duchene, O. Bonaventure, and M. Handley. How hard can it be? designing and implementing a deployable multipath TCP. In *Proceedings of the 9th USENIX conference on Networked Systems Design and Implementation (NSDI '12)*, volume 12, pages 29–29, 2012.

[161] C. Raiciu, C. Pluntke, S. Barré, A. Greenhalgh, D. Wischik, and M. Handley. Data center networking with multipath TCP. In *Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks*, page 10, 2010.

[162] C. Raiciu, D. Wischik, and M. Handley. Practical congestion control for multipath transport protocols. *University College of London Technical Report*, 2009.

[163] A. L. Ramaboli, O. E. Falowo, and A. H. Chan. Bandwidth aggregation in heterogeneous wireless networks: A survey of current approaches and issues. *Journal of Network and Computer Applications*, 35(6):1674–1690, 2012.

[164] K. Ramakrishnan, S. Floyd, D. Black, et al. The addition of explicit congestion notification (ECN) to IP. *IETF RFC 3168*, September 2001.

[165] P. Rodriguez and E. W. Biersack. Dynamic parallel access to replicated content in the Internet. *IEEE/ACM*

Transactions on Networking (TON), 10(4):455–465, 2002.

[166] P. Rodriguez, R. Chakravorty, J. Chesterfield, I. Pratt, and S. Banerjee. MAR: A commuter router infrastructure for the mobile Internet. In *Proceedings of the 2nd international conference on Mobile systems, applications, and services*, pages 217–230. ACM, 2004.

[167] K. Rojviboonchai and A. Hitoshi. An evaluation of multi-path transmission control protocol (M/TCP) with robust acknowledgement schemes. *IEICE transactions on communications*, 87(9):2699–2707, 2004.

[168] K. Rojviboonchai, T. Osuga, and H. Aida. RM/TCP: protocol for reliable multi-path transport over the Internet. In *Proceedings of the 19th International Conference on Advanced Information Networking and Applications (AINA '05)*, volume 1, pages 801–806, 2005.

[169] K. Rojviboonchai, N. Watanabe, and H. Aida. One-way-trip time (OWTT) measurement and retransmission policy for congestion control in M/TCP. In *Proceedings of the Annual Conference of IPSJ*, 2002.

[170] G. Rossini and D. Rossi. Evaluating CCN multi-path interest forwarding strategies. *Computer Communications*, 36(7):771–778, 2013.

[171] H. Sakakibara, M. Saito, and H. Tokuda. Design and implementation of a socket-level bandwidth aggregation mechanism for wireless networks. In *Proceedings of the 2nd annual international workshop on Wireless internet*, page 11. ACM, 2006.

[172] D. Sarkar. A Concurrent Multipath TCP and Its Markov Model. In *Proceedings of the IEEE International Conference on Communications (ICC '06)*, pages 615–620, June 2006.

[173] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson. RTP: A Transport Protocol for Real-Time Applications. *IETF RFC 3550*, July 2003.

[174] K. Sha, J. Gehlot, and R. Greve. Multipath Routing Techniques in Wireless Sensor Networks: A Survey. *Wireless Personal Communications*, 70(2):807–829, may 2013.

[175] S. Shailendra, R. Bhattacharjee, and S. K. Bose. Improving congestion control for Concurrent Multipath Transfer through bandwidth estimation based resource pooling. In *Proceedings of the 8th International Conference on Information, Communications and Signal Processing (ICICS '11)*, pages 1–5. IEEE, 2011.

[176] S. Shakkottai, E. Altman, and A. Kumar. The Case for Non-Cooperative Multihoming of Users to Access Points in IEEE 802.11 WLANs. In *Proceedings of the 25th IEEE International Conference on Computer Communications (INFOCOM '06)*, 2006.

[177] Z. Shamszaman, S. Ara, and I. Chong. Feasibility considerations of multipath TCP in dealing with big data application. In *International Conference on Information Networking (ICOIN '13)*, pages 708–713, Jan 2013.

[178] V. Sharma, S. Kalyanaraman, K. Kar, K. Ramakrishnan, and V. Subramanian. MPLOT: A transport protocol exploiting multipath diversity using erasure codes. In *Proceedings of the 27th IEEE Conference on Computer Communications (INFOCOM '08)*, pages 121–125, 2008.

[179] V. Sharma, K. Kar, K. Ramakrishnan, and S. Kalyanaraman. A transport protocol to exploit multipath diversity in wireless networks. *IEEE/ACM Transactions on Networking (TON)*, 20(4):1024–1039, 2012.

[180] H. Shojania and B. Li. Parallelized progressive network coding with hardware acceleration. In *Fifteenth IEEE International Workshop on Quality of Service*, pages 47–55. IEEE, 2007.

[181] W. Simpson. The Point-to-Point Protocol (PPP). *IETF RFC 1661*, July 1994.

[182] A. Singh, M. Xiang, A. Konsgen, C. Goerg, and Y. Zaki. Enhancing fairness and congestion control in multipath TCP. In *Proceedings of the 6th Joint IFIP Wireless and Mobile Networking Conference (WMNC '13)*, pages 1–8, 2013.

[183] S. Singh, T. Das, and A. Jukan. A Survey on Internet Multipath Routing and Provisioning. *IEEE Communications Surveys Tutorials*, 2015.

[184] H. Sivakumar, S. Bailey, and R. L. Grossman. PSockets: The case for application-level network striping for data intensive applications using high speed wide area networks. In *Proceedings of the ACM/IEEE conference on Supercomputing (CDROM '00)*, page 37. IEEE Computer Society, 2000.

[185] K. Sklower, B. Lloyd, G. McGregor, D. Carr, and T. Coradetti. The PPP Multilink Protocol (MP). *IETF RFC 1990*, August 1996.

[186] A. C. Snoeren. Adaptive inverse multiplexing for wide-area wireless networks. In *Proceedings of the Global Telecommunications Conference (GLOBECOM '99)*, volume 3, pages 1665–1672. IEEE, 1999.

[187] R. Stewart. Stream control transmission protocol. *IETF RFC 4960*, September 2007.

[188] T. N. Subedi, K. K. Nguyen, and M. Cheriet. OpenFlow-based in-network Layer-2 adaptive multipath aggregation in data centers. *Computer Communications*, 61:58–69, 2015.

[189] J. Sun, Y. Wen, and L. Zheng. On file-based content distribution over wireless networks via multiple paths: Coding and delay trade-off. In *Proceedings of the IEEE INFOCOM*, pages 381–385, 2011.

[190] A. S.-W. Tam, K. Xi, Y. Xu, and H. J. Chao. Preventing TCP incast throughput collapse at the initiation, continuation, and termination. In *Proceedings of the 20th IEEE International Workshop on Quality of Service*, page 29. IEEE Press, 2012.

[191] C.-L. Tsao and R. Sivakumar. On effectively exploiting multiple wireless interfaces in mobile hosts. In *Proceedings of the 5th international conference on Emerging networking experiments and technologies (CoNEXT '09)*, pages 337–348. ACM, 2009.

[192] S. Tullimas, T. Nguyen, R. Edgecomb, and S.-c. Cheung. Multimedia streaming using multiple TCP connections. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMCCAP)*, 4(2):12, 2008.

[193] I. Van Beijnum, J. Crowcroft, F. Valera, and M. Bagnulo. Loop-freeness in multipath BGP through propagating the longest path. In *IEEE International Conference on Communications (ICC) Workshops*, pages 1–6, 2009.

[194] R. van der Pol, M. Bredel, A. Barczyk, B. Overeinder, N. van Adrichem, and F. Kuipers. Experiences with MPTCP in an intercontinental OpenFlow network. In *Proceedings of the 29th TERENA Network Conference (TNC '13)*, 2013.

[195] B. Wang, W. Wei, Z. Guo, and D. Towsley. Multipath Live Streaming via TCP: Scheme, Performance and Benefits. In *Proceedings of the 3rd ACM International Conference on emerging Networking EXperiments and Technologies (CoNEXT '07)*, pages 11:1–11:12. ACM, 2007.

[196] B. Wang, W. Wei, Z. Guo, and D. Towsley. Multipath live streaming via TCP: scheme, performance and benefits. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMCCAP '09)*, 5(3):25, 2009.

[197] F. Wang, Z. M. Mao, J. Wang, L. Gao, and R. Bush. A Measurement Study on the Impact of Routing Events on End-to-end Internet Path Performance. In *Proceedings of the 2006 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM '06)*, pages 375–386. ACM, 2006.

[198] X. Wang, Z. Feng, D. Fan, Y. Xue, and V. Le. A Segment-Based Adaptive Joint Session Scheduling Mechanism in Heterogeneous Wireless Networks. In *Proceedings of the 70th IEEE Vehicular Technology Conference Fall (VTC '09)*, pages 1–5. IEEE, 2009.

[199] Y. Wen and V. Chan. Ultra-reliable Communication over Vulnerable All-Optical Networks via Lightpath Diversity. *IEEE Journal on Selected Areas in Communications, Optical Communications and Networking*, 23(8):1572–1587, August 2005.

[200] D. Wischik, M. Handley, and M. B. Braun. The resource pooling principle. *ACM SIGCOMM Computer Communication Review*, 38(5):47–52, 2008.

[201] F. Yang and P. Amer. Non-renegable Selective Acknowledgments (NR-SACKs) for MPTCP. In *In proceedings of the 27th International Conference on Advanced Information Networking and Applications Workshops (WAINA)*, pages 1113–1118, March 2013.

[202] F. Yang and P. Amer. Work in progress: Using one-way communication delay for in-order arrival MPTCP scheduling. In *Proceedings of the 9th International Conference on Communications and Networking in China (CHINACOM)*, pages 122–125, Aug 2014.

[203] M. Zhang, B. Karp, S. Floyd, and L. Peterson. RR-TCP: a reordering-robust TCP with DSACK. In *Proceedings of the 11th IEEE International Conference on Network Protocols*, pages 95–106, 2003.

[204] M. Zhang, J. Lai, A. Krishnamurthy, L. L. Peterson, and R. Y. Wang. A Transport Layer Approach for Improving End-to-End Performance and Robustness Using Redundant Paths. In *Proceedings of the General Track: USENIX Annual Technical Conference*, pages 99–112, 2004.

[205] W. Zhang, W. Lei, S. Liu, and G. Li. A general framework of multipath transport system based on application-level relay. *Computer Communications*, 51:70–80, 2014.

[206] D. Zhou, W. Song, and M. Shi. Goodput improvement for multipath TCP by congestion window adaptation in multi-radio devices. In *Proceedings of the IEEE Consumer Communications and Networking Conference (CCNC '13)*, pages 508–514. IEEE, 2013.

Appendix
List of Acronyms

AOLIA	Adapted Opportunistic Linked Increases Algorithm
ATLB	Arrival-Time matching Load-Balancing
BERP	Bandwidth Estimation Based Resource Pooling
BMP	Buffer Management Policy
CMT	Concurrent Multipath Transfer
DAC	Delayed ACK for CMT
DBAS	Deplorable Bandwidth Aggregation System
ECMP	Equal Cost Multipath
ECT	Equal Cost Tree
EDPF	Earliest Delivery Path First
FOSM	Flow-Oriented Scheduling Mode
FPS	Forward Prediction Scheduling
HIP	Host Identity Protocol
HLB	Head-of-Line Blocking
HTTP-RP	HTTP Request Pipelining
HTTP-RRR	HTTP Range Retrieval Request
LACP	Link Aggregation Control Protocol
LIA	Linked Increase Algorithm
LISP	Locator/Identifier Separation Protocol
MPTCP	Multipath TCP
NAT	Network Address Translation
OLIA	Opportunistic Linked Increases Algorithm
PCA	Per-Conversation Allocation
PET	Packet-Pair based EDPF for TCP applications
PFA	Per-Flow Allocation
POSM	Packet-Oriented Scheduling Mode
RP	Resource Pooling
RR	Round Robin
RTT	Round-Trip Time
SACK	Selective Acknowledgment
SCTP	Stream Control Transmission Protocol
SPB	Shortest Path Bridging
TRILL	Transparent Interconnection of a Lot of Links
WRR	Weighted Round Robin