Software Defined Multi-Path TCP Solution for Mobile Wireless Tactical Networks

Qi Zhao*, Pengyuan Du*, Mario Gerla*, Adam J. Brown† and Jae H. Kim†
*Department of Computer Science
University of California, Los Angeles, CA, USA
{qi.zhao, pengyuandu, gerla}@cs.ucla.edu
†Boeing Research & Technology, Seattle, WA, USA
{adam.j.brown2, jae.h.kim}@boeing.com

***Abstract*—Naval Battlefield Network communications rely on wireless network technologies to transmit data between different naval entities, such as ships and shore nodes. Existing naval battle networks heavily depend on the satellite communication system using single-path TCP for reliable, non-interactive data. While satisfactory for traditional use cases, this communication model may be inadequate for outlier cases, such as those arising from satellite failure and wireless signal outage. To promote network stability and assurance in such scenarios, the addition of unmanned aerial vehicles to function as relay points can complement network connectivity and alleviate potential strains in adverse conditions. The inherent mobility of aerial vehicles coupled with existing source node movements, however, leads to frequent network handovers with non-negligible overhead and communication interruption, particularly in the present single-path model. In this paper, we propose a solution based on multi-path TCP and software-defined networking, which, when applied to mobile wireless heterogeneous networks, reduces the network handover delay and improves the total throughput for transmissions among various naval entities at sea and littoral. In case of single link failure, the presence of a connectable relay point maintains TCP connectivity and reduces the risk of service interruption. To validate feasibility and to evaluate performance of our solution, we constructed a Mininet-WiFi emulation testbed. Compared against single-path TCP communication methods, execution of the testbed when configured to use multi-path TCP and UAV relays yields demonstrably more stable network handovers with relatively low overhead, greater reliability of network connectivity, and higher overall end-to-end throughput. Because the SDN global controller dynamically adjusts allocations per user, the solution effectively eliminates link congestion and promotes more efficient bandwidth utilization.**

***Index Terms*—Software-Defined Networking, Multi-path TCP, Mobile, Wireless, Tactical Networks**

I. Introduction

The Naval Battlefield Network (NBN), comprised of diverse, interconnected naval assets, communicate wirelessly, traditionally through reliance on governmental and commercial Satellite Communication (SATCOM) systems. Navy vessels may be equipped with one or more SATCOM terminals connected to a common service router, which arbitrates traffic through from shipboard Local Area Networks (LAN) to the SATCOM network through a dedicated uplink [1]. The relay satellite forwards incoming traffic to connected destination naval entities, thus completing the intended transmission, whether ship-to-ship, ship-to-shore, or ship-to-aircraft.

Shipboard architecture of NBN is limited by performance constraints in its current incarnation. Because different satellite terminals provide segregated connections to different user classes, a user may experience congestion overload on the assigned satellite link despite the availability of potentially under-utilized satellite links. Furthermore, while the naval network uses Internet Protocol (IP) service hardware routers, there may be barriers to adoption of new innovative network technologies (e.g., Information Centric Network) following initial deployment. Both concerns can be addressed through usage of Software Defined Networking (SDN) framework proposed by [2], to achieve the bandwidth sharing and load balancing across the NBN satellite communication links.

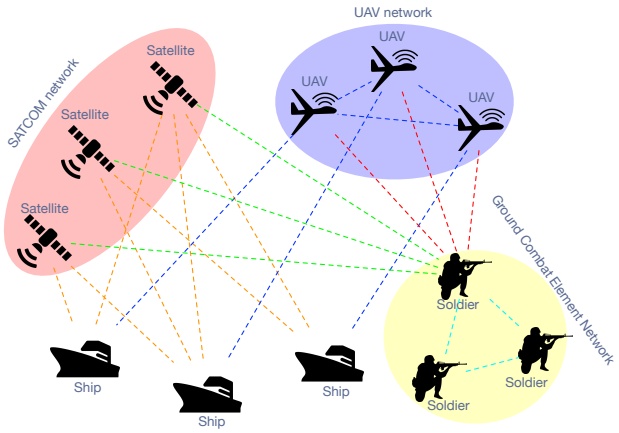


Fig. 1. Naval Battlefield Network.

However, in a modern battlefield represented in Figure 1, the SATCOM links are not the only communication media. Unmanned Aerial Vehicles (UAVs) provide an alternative communication link for naval entities, such as ship and shore nodes. [2][1] only consider the ship-to-ship communication via SATCOM, such that the network topology is semi-static, i.e., there are no dynamic changes initiated once a model has begun execution. In that testbed environment, which is incapable of altering the network topology dynamically, the addition of mobile nodes, such as UAVs and Ground Combat Elements (GCEs), is not possible without enhancements.

978-1-5386-7185-6/18/$31.00 ©2018 IEEE

Composed primarily of infantry units, GCE requires connection to a Tactical Operation Center (TOC) on the ship. Whereas GCE uses single-path communication via SATCOM in the conventional architecture, in the modern scenario, GCE also utilizes the links provided by UAVs within range. Other media projected to soon be available include Geo-stationary Earth Orbit (GEO), Medium Earth Orbit (MEO) and Low Earth Orbit (LEO) satellites. For mobile service-providing units specifically, the network topology changes resulting from connection point migration gives rise to network handovers, which impact communication performance due to service interruptions and handover overhead. We will demonstrate the benefits from a dynamic controller capable of reconfiguring connections and reallocating bandwidth for users in response to topology changes and traffic demands.

To prepare NBN for expanding connection options, we will demonstrate that GCE's simultaneous utilization of multiple links using the Multi-Path Transmission Control Protocol (MPTCP) at the transport-layer for communication among naval entities promotes more efficient bandwidth utilization and reduces link congestion. We propose the use of an SDN controller to enable real-time traffic engineering and systematic management of diverse flows, including the MPTCP subflows, for fair rate allocations. MPTCP complements SDN by aggregating user bandwidth and smoothing network handover.

The paper is organized as follows: Section II presents the related work; Section III describes the network scenarios targeted by our solution; Section IV outlines the design of the solution; Section V describes the emulation testbed; Section VI presents and analyzes testbed results; and Section VII discusses the conclusions and implications of the research.

II. Related Work

A. Multi-path TCP

Multi-path TCP [3] extends the conventional Single-Path TCP (SPTCP) by leveraging the availability of multiple data paths across multiple interfaces between a pair of hosts, which increases the reliability at end-to-end communication. Dividing flows just above the transport layer enables MPTCP to present a logically single connection to upper layers while splitting data across several subflows sent through different communication paths. Benefits of MPTCP include efficient resource utilization, higher end-to-end throughput and smoother reaction to connection failures. MPTCP uses a path-manager to handle the subflows' creation and removal, reassemble to organize out-of-order packets from different subflows, scheduler to designate traffic to a subflow, and congestion control mechanism for the subflows [4]. [5] uses MPTCP to utilize multiple network interfaces for mobile devices and proposes a weighted MPTCP scheduler that allows transmission of certain controllable percentages of data per network interface. [6] proposes a network coding scheme that affords path diversity for each MPTCP subflow. [7] applies MPTCP in LEO satellite networks to improve throughput performance. In general cases, MPTCP can be used to improve end-to-end throughput for networks with multiple communication links.

B. Software Defined Networking

The SDN architecture [8] physically separates the network control plane from the data forwarding plane, to allow direct device control via the control plane. SDN divides the vertical integration of traditional IP networks and reassigns the network control functions from the underlying routers and switches, which promotes the logical centralization of network control and enables network programmability [9]. [2] proposes an SDN framework to improve the bandwidth sharing and load balancing for multiple satellite communication links. When a packet arrives at an SDN-enabled switch, preconfigured, firmware-level forwarding rules ensure delivery of the packet to the intended destination. SDN enabled infrastructure switches receive forwarding rules from the centralized controller via open standard protocols, such as OpenFlow [10]. OpenFlow enhanced network security, smooth wireless network handover, scalable data center networks, heterogeneous mobile networks, more energy-efficient networks and wide-area networks [2]. Implementing SDN over SATCOM systems is feasible because [11] has already successfully divided SATCOM channels into control and data plane.

III. Communication Scenarios

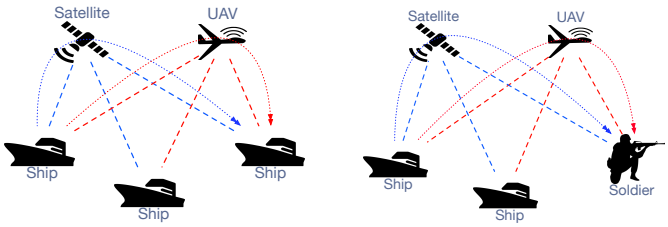
In our NBN architecture, we consider two mobile naval network scenarios in Figure 2: ship-to-ship and ship-to-shore.

A. Ship to Ship

Ships communicate with each other to exchange battlefield information, and the existing framework may be inadequate when (i) a SATCOM link fails and (ii) an urgent combat imperative necessitates real-time communication. UAVs, shown in Figure 2(a), can provide a relay communication link supplementing SATCOM links with lower delay to construct connections with naval entities in range.

B. Ship to Shore

Shore nodes, such as GCEs, military bases and combat vehicles, communicate with ship to exchange battle command information. UAVs, shown in Figure 2(b), may be desirable when (i) complex terrain frustrates available SATCOM links; (ii) shore and ship nodes require live media streaming; (iii) connectible SATCOM links are congested or have failed.



(a) Ship-to-Ship (b) Ship-to-Shore

Fig. 2. Naval Heterogeneous Networks Communication Scenarios.

IV. Solution Design

We propose combining SDN and MPTCP for optimized communications in a heterogeneous SATCOM and UAV network. A contemplated network consists of host ships each equipped with an SDN switch for managing traffic, data service providers in the form of SATCOM systems and UAVs, GCEs as data customers, and a global SDN controller to serve as a bandwidth broker. GCEs utilize MPTCP across multiple interfaces to communicate with ships over diverse links as directed by the SDN controller, which collects network statistics, such as link capacities, topology changes. The controller deploys and updates traffic flow distribution against the GCE demand according to instructions generated by the Flow Deviation Method (FDM) algorithm [12]. Extensions to the original FDM algorithm enable handling of dynamic traffic flow allocation and minimization of the impact from flow reallocation. By smoothly distributing traffic flow, the algorithm avoids congestion for more efficient load balancing.

1) Multi-path TCP: We deploy MPTCP to achieve smoother reaction to network changes due to UAV node migration. A link is instantly established between UAVs and GCEs within range, joining the existing SATCOM link and allowing immediate utilization of low-latency links with low overhead and without interruption of existing sessions.

2) Software-Defined Networking: SDN centralizes network logic and topology information in the control plane, converting commodity hardware switches into packet forwarding devices [13]. In the event the network encounters congestion when UAV links have insufficient bandwidth to meet total demand, without SDN, MPTCP must mitigate congestion with the default scheduler and flow controller only. The default traffic scheduler of MPTCP is Lowest Round Trip Time First (LowRTT), which designates the subflow with lowest RTT as the primary path and compels MPTCP traffic to empty the queue to that link first, potentially creating unfairness. SDN alleviates fairness concerns by computing the optimal flow for each MPTCP subflow and sets a limiter at the source SDN switch to enforce subflow compliance. SDN trades throughput to improve fair flow allocation. Without SDN, MPTCP greedily fills the pipes according to statistic fluctuations in demands. Another advantage of SDN is guidance in the choice of links. MPTCP chooses the first link found, whereas FDM guides SDN to allocate optimal link usage over many options, GEO, MEO, LEO satellites or UAVs.

V. Emulation Environment

A. Testbed Implementation

We setup a testbed, diagramed in Figure 3, to evaluate our proposed solution using the Mininet-WiFi emulator [14] in Linux Ubuntu 14.04. Mininet-WiFi, an extension to the Mininet SDN network emulator [15], emulates WiFi mobile stations and access points with virtualized WiFi interface modules. We choose Mininet-WiFi instead of the Common Open Research Emulator (CORE) [16] is because Mininet-WiFi is more friendly to SDN paradigm and we have flexible choices of SDN controller, including open source controllers and our own implemented controller. Moreover, it will be easier to achieve integration with our previous work based on Mininet. To utilize MPTCP, we install the Linux kernel implementation of MPTCP v0.92. Open vSwitch (OVS) [17] is also installed to emulate SDN edge devices, which communicate with the SDN controller using the OpenFlow protocol. For the SDN controller, we implement the whole control logic, which closely interacts with Mininet-WiFi using python APIs to collect network and user information and to deploy the control configuration onto the SDN edge devices. We integrate an FDM implementation [12], detailed in our previous work [1], to repeatedly compute the optimal bandwidth allocation. The SDN controller reconfigures flow tables and queues on SDN edge devices in response to bandwidth allocation updates.

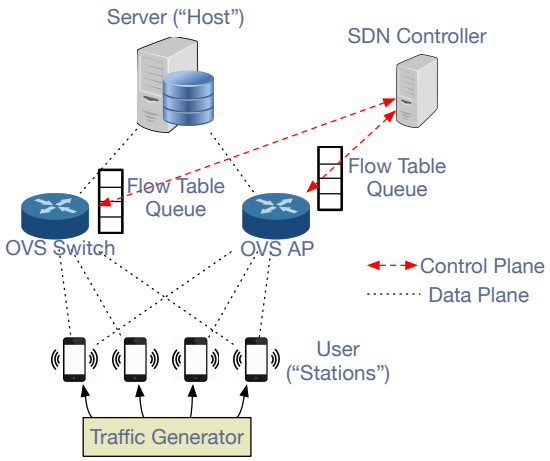


Fig. 3. Architecture Overview of the Mininet-WiFi Testbed.

B. Mininet emulated networks

We leverage Mininet-WiFi to build the mobile wireless heterogeneous naval network environment in which ships are static “hosts” from (to) which users download (upload) contents. An “OVS”, functioning as a SATCOM or UAV switch, handles packet routing and forwarding. Scenario I contemplates a ship-to-ship communication model in which ships serve as hosts and users. In Scenario II, shore nodes, which can be static or mobile “stations”, are the user nodes communicating with ship hosts.

1) UAV network: The core of the emulated heterogeneous naval network is an OVS to act as the wireless access point (AP). Mininet-WiFi connects virtualized wireless interfaces to an SDN switch and simulates the wireless medium using the $mac802\_11hwsim$ module. The switches become wireless AP through `hostapd` daemon. The model implements propagation and shadowing to simulate wireless channel characteristics (e.g., path loss, interference). Each UAV switch, equipped with SDN capabilities and configured using the OpenFlow API, is emulated as a virtualized AP, with one WLAN and multiple Ethernet interfaces connected to the backhaul network.

2) SATCOM network: We emulate the SATCOM network with the OVS switch, which has only Ethernet interfaces. For greater fidelity, the testbed uses Linux traffic control utilities to configure link parameters, such as bandwidth, link latency, loss, and jitter, upon instantiation.

3) Mobile hosts: Mininet-WiFi extends the base emulator to support WLAN connection construction by extending switches. For each host device, WLAN (UAV) and Ethernet (SATCOM) interfaces are created to enable communication through both networks. The emulator supports host mobility and movement customization using a predefined mobility model, such as random walk and random way point. For mobile host interfaces, the python API `net.nameToNode` can configure the host object using Linux shell commands. The routing table for each host is defined using `ip route` for concurrent communication across multiple interfaces. In this manner, the host object acts as a handler used by the SDN controller for instruction of a mobile hosts on interface creation and removal and network switching.

C. SDN controller

The SDN controller persistently monitors the network status, calculates the optimal bandwidth allocation, and updates the allocation to corresponding devices. To collect necessary network information, the controller utilizes the Mininet-WiFi python API to query all naval entities and service providers. The overhead for information gathering and bandwidth allocation deployment is negligible because the information transmission is directly through the python API in the testbed, but in reality there is a transmission cost in the control plane.

1) Network solicitation: FDM calculation for optimized bandwidth allocation requires a set of inputs including network topology, switch bandwidth capacity, and wireless channel measurements, such as link delay and capacity. The Mininet-WiFi python API `sta.params['associatedTo']`, `sta.intfList()` can be called to gather a list of networks to describe the overall topology. For wireless channel measurements, Mininet-WiFi provides a python class `wirelessLink`, which can be used to retrieve values for bandwidth, latency, and signal strength to indicate link quality.

2) Bandwidth allocation: The FDM algorithm is implemented as a bandwidth allocation optimizer in the SDN controller using a python module. If total demand exceeds total bandwidth capacity, indicating an oversaturated network and an infeasible allocation, we first manually temper bandwidth demands to reach a feasible network that can be optimized by FDM. To support host mobility and cases of network failure, the SDN controller periodically solicits network information to detect changes in user connectivity. Upon change, the FDM optimizer recalculates and updates bandwidth allocations for SDN edge devices.

3) Bandwidth allocation deployment: The SDN controller uses the `ovs-ofctl` and `ovs-vsctl` commands to establish and configure the OpenFlow flow table and queues as shown in Figure 3. Each entry in the flow table contains the input and output port number and the corresponding action. An SDN edge switch references the flow table for the matching flow entry before processing a packet according to the associated set of actions. Possible actions include (i) designating the port number to which the packet should be forwarded, (ii) associating the packet with an OpenFlow queue at the designated output port, and (iii) dropping the packet.

OVS uses a queuing mechanism to control traffic rates. Once FDM returns bandwidth allocation, we create the queue for each interface of each user with the appropriate bandwidth limit. The SDN controller can specify the queue with `Set-Queue` to forward packets to a particular queue based on the matching flow entry. To lookup matching flow entries, we rely on the IP address of the associated network interface.

VI. Experiment and Evaluation

In this section, we describe the design of our experiments for evaluating the performance of our proposed solution. Based on the scenarios in Section III, we mainly focus on the mobility management for utlization of UAV links. To simulate the mobility for both UAVs and shore nodes, without loss of generality, we choose to enable the mobility on users rather than the APs. We use iPerf3 [18] to generate fixed rates of TCP traffic per user.

A. Evaluation Scenario I: Direct Move Experiment

As Figure 4 shows, we create two mobile users and one host server, each enabled with MPTCP. We create one OVS switch (SATCOM) and one OVS AP (UAV WLAN). The SATCOM link is configured with $250ms$ delay and $50Mbps$ backhaul network capacity while the UAV link is configured with $10ms$ delay and $1Mbps$ backhaul network capacity. We restrict $3Mbps$ sending rate for each user and set the total emulation time to be $100s$. The model emulates a network handover from SATCOM to UAV as each user enters the range of the UAV at time $60s$.

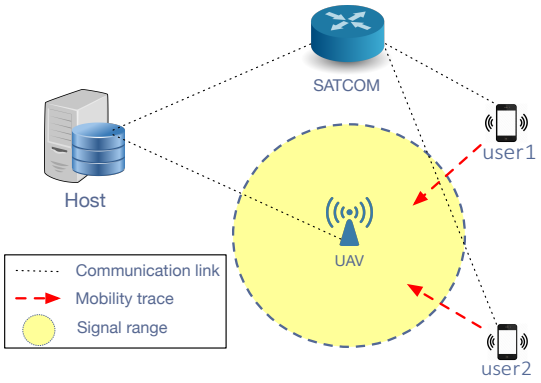


Fig. 4. Evaluation Experiment Scenario I.

Figure 5 graphs the instantaneous throughput gathered from the experiment for two different protocols: SPTCP and MPTCP with FDM (i.e., SDN-controlled). Where the throughput using SPTCP drops to $0Mbps$ for about 4 seconds during the network handover, the throughput using MPTCP with FDM remains stable, ensuring Quality of Service (QoS).

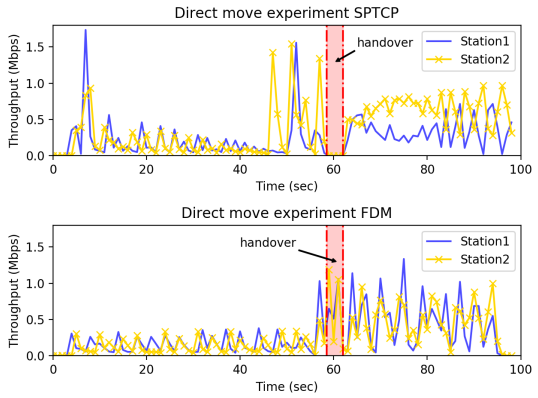


Fig. 5. Instantaneous Throughput Curve as a Function of Time.

Figure 6 shows average throughput for the two stations using SPTCP, MPTCP only, and MPTCP with FDM. The average throughput is $0.4875Mbps$ for SPTCP, $0.3728Mbps$ for MPTCP, and $0.4188Mbps$ for MPTCP with FDM. Similar average throughput for each protocol indicates that the stations can fairly share bandwidth resource. As expected, with only one network handover lasting approximately $4s$, SPTCP has the highest average throughput. With throughput dropping to $0Mbps$ during handovers, SPTCP is expected to perform less well in terms of throughput when handover time is of relatively greater proportion with respect to total experiment duration.

Comparing MPTCP in isolation and MPTCP with FDM in Figure 6, the latter achieves higher throughput (as represented by the rightmost bar) and slightly lower throughput variation (as signified by the error bar), which demonstrates that the FDM optimizer allocates the total bandwidth more efficiently than greedy heuristics using the LowRTT scheduler. In SATCOM and UAV heterogeneous networks, MPTCP prefers the UAV link, which has lower delay, potentially beyond the point of saturation to the detriment of overall performance. The FDM optimizer eliminates potential congestion in advance and assigns fairer allocations, which improve performance.

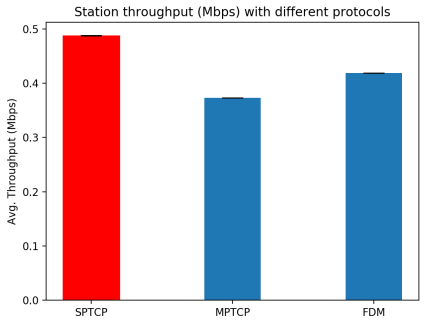


Fig. 6. Average Throughput for Two Stations.

Our solution reallocates bandwidth according to FDM optimization for reduction of network handover impact. When handovers are infrequent or are guaranteed to impact only low priority transmissions, SPTCP may be adequate. For other use cases, our solution better guarantees a continuous communication environment with controlled impact to throughput.

B. Evaluation Scenario II: Random Walk Experiment

Where Scenario I contemplates direct movement of users to enter UAV range, Scenario II, diagramed in Figure 7, contemplates random movement near UAV range. Each user may enter or exit the UAV range during the emulation, potentiating multiple network handovers. Sending rate $3Mbps$ and emulation time $100s$ remain unchanged from Scenario I.

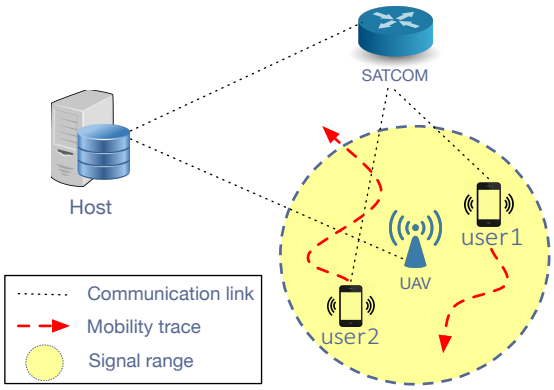


Fig. 7. Evaluation Experiment Scenario II.

Figure 8 graphs the instantaneous throughput gathered from the experiment for SPTCP and MPTCP with FDM. Random user movement prompted numerous network handovers. From the upper figure in Figure 8, multiple network handovers emerge since two stations randomly walk around near the UAV's signal range. Every time when the station moves in (out) the UAV's range, it has to switch the network connection to UAV's (SATCOM's) link since the single-path communication. As a consequence, at least $3s$ to $5s$ communication interruption is introduced as indicated by the $0Mbps$ throughput red zone. MPTCP with FDM is much better in this case. During the experiment time, there is no communication interruptions even though sometimes the throughput is very close to $0Mbps$. The throughput of MPTCP with FDM shows more dynamical fluctuations than SPTCP because SATCOM and UAV networks have a great disparity in link delay, so that the throughput drops rapidly due to the large out-of-order overhead introduced by MPTCP whenever the station starts to utilize both links simultaneously.

Figure 9 shows the average throughput for the two stations for SPTCP, MPTCP only, and MPTCP with FDM to be $0.3121Mbps$, $0.4738Mbps$ and $0.4602Mbps$, respectively. As predicted, the overall throughput of SPTCP decreases due to the frequent communication interruptions from multiple handovers. Both MPTCP experiments yielded comparatively higher average throughput due to reduced handover overhead and continuous SATCOM link connectivity.

FDM allocates the available bandwidth more fairly than the LowRTT scheduler, which can be seen from the smaller

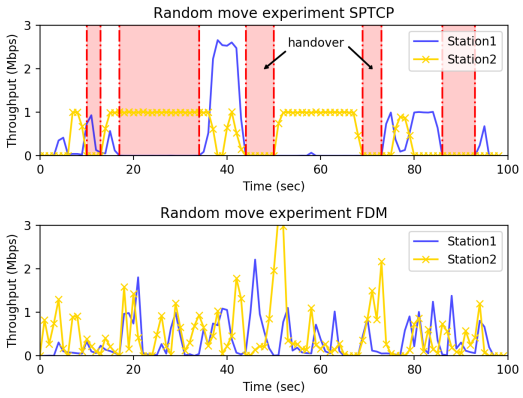


Fig. 8. Instantaneous Throughput Curve as a Function of Time.

error bar in Figure 9. The overall throughput of MPTCP, however, is slightly better than MPTCP with FDM presumably due to the frequency of the handover. If a network handover occurs prior to network congestion, the ramifications of greedy consumption may not be fully experienced.

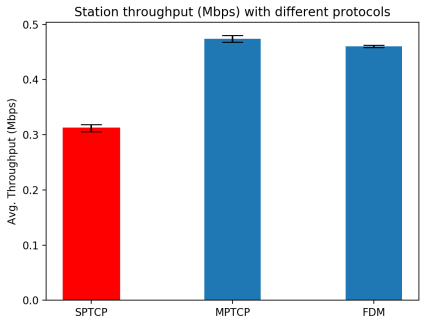


Fig. 9. Average Throughput for Different Stations.

In summary, our solution handles dense network handovers with improved overall throughput. Though MPTCP with FDM had slightly lower average throughput than MPTCP only, our solution achieves fairer bandwidth allocation for greater QoS.

VII. Conclusion

This paper is a component of our extended work regarding naval multi-SATCOM network architectures, which adds separate relay communication links provided by mobile UAVs. Our solution utilizes SDN and MPTCP to support dynamic bandwidth allocation for both static and mobile users. To adjust bandwidth allocation in response to network topology changes, the FDM algorithm periodically collects network information to dynamically reallocate bandwidth. Our proposed solution has been demonstrated to be capable of handling the mobility management of heterogeneous naval networks for both sparse and dense network handover cases. Our solution in dense network handover case outperforms the sparse network handover case in terms of overall throughput. Future work includes development of a large scale emulation testbed capable of supporting more users and networks to evaluate the scalability and robustness of our solution.

Acknowledgment

This work was supported by the Office of Naval Research (ONR) Contract N00014-15-C-5038 Software Defined Naval Battlefield Network (SD-NBN). The authors would like to thank Dr. Santanu Das (ONR Program Manager) for his support and guidance.

References

[1] P. Du, F. Pang, T. Braun, M. Gerla, C. Hoffmann, and J. H. Kim, "Traffic optimization in software defined naval network for satellite communications," in *MILCOM 2017 - 2017 IEEE Military Communications Conference (MILCOM)*, Oct 2017, pp. 459–464.

[2] S. Nazari, P. Du, M. Gerla, C. Hoffmann, J. H. Kim, and A. Capone, "Software defined naval network for satellite communications (sdn-sat)," in *MILCOM 2016 - 2016 IEEE Military Communications Conference*, Nov 2016, pp. 360–366.

[3] A. Ford, C. Raiciu, M. Handley, S. Barre, and J. Iyengar, "Architectural guidelines for multipath tcp development," Tech. Rep., 2011.

[4] A. Ford, C. Raiciu, M. Handley, and O. Bonaventure, "Tcp extensions for multipath operation with multiple addresses," Tech. Rep., 2013.

[5] T. De Schepper, J. Struye, E. Zeljkovic, S. Latré, and J. Famaey, "Software-defined multipath-tcp for smart mobile devices," in *2017 13th International Conference on Network and Service Management (CNSM)*. IEEE, 2017, pp. 1–6.

[6] G. Giambene, D. K. Luong, M. Muhammad *et al.*, "Network coding and mptcp in satellite networks," in *Advanced Satellite Multimedia Systems Conference and the 14th Signal Processing for Space Communications Workshop (ASMS/SPSC), 2016 8th*. IEEE, 2016, pp. 1–8.

[7] P. Du, S. Nazari, J. Mena, R. Fan, M. Gerla, and R. Gupta, "Multipath tcp in sdn-enabled leo satellite networks," in *Military Communications Conference, MILCOM 2016-2016 IEEE*. IEEE, 2016, pp. 354–359.

[8] N. McKeown, "Software-defined networking," *INFOCOM keynote talk*, vol. 17, no. 2, pp. 30–32, 2009.

[9] B. A. A. Nunes, M. Mendonca, X.-N. Nguyen, K. Obraczka, and T. Turletti, "A survey of software-defined networking: Past, present, and future of programmable networks," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 3, pp. 1617–1634, 2014.

[10] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "Openflow: enabling innovation in campus networks," *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 2, pp. 69–74, 2008.

[11] J. Bao, B. Zhao, W. Yu, Z. Feng, C. Wu, and Z. Gong, "Opensan: a software-defined satellite network architecture," in *ACM SIGCOMM Computer Communication Review*, vol. 44, no. 4. ACM, 2014, pp. 347–348.

[12] L. Fratta, M. Gerla, and L. Kleinrock, "The flow deviation method: An approach to store-and-forward communication network design," *Networks*, vol. 3, no. 2, pp. 97–133, 1973.

[13] D. Kreutz, F. M. Ramos, P. E. Verissimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-defined networking: A comprehensive survey," *Proceedings of the IEEE*, vol. 103, no. 1, pp. 14–76, 2015.

[14] R. R. Fontes, S. Afzal, S. H. Brito, M. A. Santos, and C. E. Rothenberg, "Mininet-wifi: Emulating software-defined wireless networks," in *Network and Service Management (CNSM), 2015 11th International Conference on*. IEEE, 2015, pp. 384–389.

[15] R. L. S. De Oliveira, A. A. Shinoda, C. M. Schweitzer, and L. R. Prete, "Using mininet for emulation and prototyping software-defined networks," in *Communications and Computing (COLCOM), 2014 IEEE Colombian Conference on*. IEEE, 2014, pp. 1–6.

[16] J. Ahrenholz, "Comparison of core network emulation platforms," in *2010 - MILCOM 2010 MILITARY COMMUNICATIONS CONFERENCE*, Oct 2010, pp. 166–171.

[17] B. Pfaff, J. Pettit, T. Koponen, E. J. Jackson, A. Zhou, J. Rajahalme, J. Gross, A. Wang, J. Stringer, P. Shelar *et al.*, "The design and implementation of open vswitch." in *NSDI*, 2015, pp. 117–130.

[18] C.-H. Hsu and U. Kremer, "Iperf: A framework for automatic construction of performance prediction models," in *Workshop on Profile and Feedback-Directed Compilation (PFDC), Paris, France*. Citeseer, 1998.