Uluslararası İleri Doğa Bilimleri ve Mühendislik Araştırmaları Dergisi
Sayı 8, S. 100-107, 11, 2024

Araştırma Makalesi

IJANSER
https://as-proceeding.com/index.php/ijanser
ISSN:2980-0811

International Journal of Advanced Natural Sciences and Engineering Researches
Volume 8, pp. 100-107, 11, 2024

Research Article

TCP for Data Centers: Short Flows and Incast

Omar Tareq[*1], Qutaiba Ibrahem Ali [2]

[1]*Department of Computer Techniques Engineering / Technical Engineering College for Computer and AI / Mosul, Northern Technical University, Iraq*
[2]*Department of Computer Engineering/College of Engineering, University of Mosul, Iraq*

[*]*(omar.zyad@ntu.edu.iq) Email of the corresponding author*

(Received: 04 December 2024, Accepted: 06 December 2024)

(3rd International Conference on Recent Academic Studies ICRAS 2024, December 03-04, 2024)

ATIF/REFERENCE: Tareq, O. & Ali, Q. I. (2024). TCP for Data Centers: Short Flows and Incast. *International Journal of Advanced Natural Sciences and Engineering Researches*, 8(11), 100-107.

***Abstract*—** Data center network challenges include meeting modern application demands due to limits of the traditional Transmission Control Protocol (TCP). This study evaluates the TCP Incast phenomenon, a common issue in data center environments where multiple servers respond simultaneously, leading to network congestion, packet loss, and retransmission delays. To explore the impact of TCP Incast on data center performance, the paper investigates key performance indicators including bandwidth utilization, server concurrency, and load balancing effectiveness. By modeling scenarios and evaluating mitigation strategies, the research proposes practical solutions to enhance TCP efficiency in high-bandwidth, low-latency data center environments. Results highlight the importance of bandwidth scaling, advanced load distribution algorithms, and optimized traffic scheduling mechanisms to mitigate the adverse effects of TCP Incast, thereby improving overall system performance.

Keywords- TCP Incast, Data Centers, Short Flows, Bandwidth Utilization, Server Concurrency, Load Balancing, Congestion Control, Traffic Scheduling, Low Latency, High Throughput.

I. INTRODUCTION

The TCP has been widely recognized as one of the standard protocols for reliable data transfer in traditional networks. However, current advanced demands for data center networks have become key challenges for TCP to realize low latency, high throughputs, and scalability. These environments range from supporting large numbers of simultaneous connections, dynamic traffic interactions and strict performance demands, which evidently makes the conventional TCP suboptimal seeing that it is very inefficient when it comes to bandwidth utilization, latency [1]. Another problem present in data centers is TCP Incast where many servers serve a single client at once and fill network buffers which lead to large numbers of packet loss and retransmission delays. This issue is particularly typical of many-to-one communication patterns that are typical for distributed storage systems and parallel computation environments. The worst effect of Incast is that it causes extreme

degradation in the performance of the real-time applications whenever the bandwidth is low [2]. Although various enhancements to TCP, such as Datacenter TCP (DCTCP), have been proposed, fundamental challenges remain, particularly in handling synchronized, large-scale traffic patterns [3][4]. In this research work, TCP Incast is evaluated in terms of its effects on data center utilization using key performance indicators such as bandwidth, concurrency of server, and load balancing effectiveness. By modelling key scenarios and evaluating mitigation strategies, the paper aims to propose practical solutions for improving TCP performance in modern data centers. To reduce latency and improve the level of performance insight we are going to determine major factors affecting data center performance.

Available bandwidth defines the data transfer capacity of the network in each time. Otherwise, where the transmission bandwidth is less than the total transmission rate the network becomes congested, and packets are dropped. This is because with high bandwidth implementation the probability of TCP Incast is decreased and there is associated general enhanced performance [5]. The overall number of servers that participate in handling the received requests. Thus, raising the number of servers increases the availability of servers and consequently brings down (λ per server) and, therefore, a chance of delay, while at the same time, when there are many small servers, there would be Incast [6]. The number of requests each server can handle per second, high processing rates also decrease the time taken in service on the server is also minimized. That is, if the processing rate is low compared to the request rate (λ per server), then the latency rises noticeably [7]. The number of requests reaching the data center per second, longer delays will occur if the arrival rate exceeds capacity of servers or network because then we will get queuing. The higher arriving rates raise the probability of packet loss and thus add retransmission delays. A method for distributing requests over servers depending on server current load. Efficient servers load balancing helps prevent overloads, which speeds up response times. If load balancing is not inefficient, which leads to some servers getting overburdened and increasing overall latency [8]. TCP Incast, this happens when several servers flood a certain point with response messages, for example, load balancer. This will affect the network because it will increase the measures of packet loss and retransmission delay. The above effect worsens especially where the number of servers is large relative to the available bandwidth.

II. LITERATURE REVIEW

Understanding the TCP Incast problem in today's data center environments has been widely investigated especially because of the inability of TCP to offer the needed efficient transmission in present day environments. This issue manifests in many to one communication when a number of servers are responding at the same time overwhelming the network causing many packets to be wasted and many more to be sent repeatedly. Researchers have offered several proposed solutions aimed at reducing these challenges, which is based on enhancing the functionality of TCP mechanisms and the network.

A. TCP Enhancements or Replacement

J. Ousterhout interduce a foundational work that highlight TCP's limitations in datacentres and propose a complete replacement with a more tailored protocol that eliminates TCP's inherent inefficiencies, Homa, a new datacentre transport protocol is presented in this paper. The message based, connectionless design in Homa achieves low latency and better load distribution for this environment, providing a potential alternative to TCP [1]. C.Raiciu et al. proposed a study aims to examine MPTCP's effectiveness in improving bandwidth use and robustness in datacentres, and identify scenarios where MPTCP yields the most benefits, such as increased throughput and better load balancing, across different topologies and traffic patterns, the paper contributes a proposal for MPTCP in datacentres, showing through simulations that it outperforms single-path TCP in throughput and fairness. It also introduces a new "dual-homed" FatTree topology that, with MPTCP, achieves better performance without extra costs, illustrating how datacentres could benefit from multipath protocols in

existing and future architectures [2]. Sen Liu et al. intend to propose a task aware TCP variant, TaTCP, to minimize task completion time by letting flows inside a task share tardiness information. We want to coordinate the flows to balance traffic effectively and for stalled flows. The study introduces TaTCP, which dynamically coordinates flow rates based on task progress and stalled flows, leading to a reduction of task completion times by up to 70% compared to conventional protocols, with demonstrated effectiveness across various datacentre topologies [3]. Yu Tianfang and Qiu Xuesong proposed a paper introduces a Software-Defined Networking (SDN)-based congestion control mechanism, STCC, to reduce congestion in datacentre networks by monitoring network metrics and dynamically adjusting congestion windows, aiming to enhance throughput and reduce retransmissions, they describe an SDN based congestion control model deployed at STCC, which monitors network state in real time and identifies congestion points and optimal routing paths, dynamically applies congestion control on the network state and outperforms existing approaches by increasing through put and reducing packet retransmissions [4]. H. Lim et al. introduce a Timeout-Less Transport (TLT) mechanism using commodity switches to eliminate transport layer timeouts by differentiating packets based on priority, TLT shows significant latency improvements, reducing 99%. Flow Completion Time by up to 97.2% in incast scenarios and improving response times across various transport protocols [9]. T. Zhang et al. propose a paper discusses challenges for datacentre networks (DCNs) and present requirements including fast convergence in high bandwidth environments and compatibility between the new DCN and traditional TCP for older applications, Fast and Friendly Converging (FFC) protocol is introduced, which combines Explicit Congestion Notification (ECN) and Round-Trip Time (RTT) measurements to achieve faster convergence, while remaining TCP friendly. Simulations and real-world implementations of FFC are tested [10].

B. Network-Based Solutions

Beyond protocol-level enhancements, network infrastructure plays a critical role in mitigating TCP Incast. M. K. Mukerjee et al. address the challenge of adapting TCP performance to cope with rapid bandwidth fluctuations in reconfigurable datacentre networks (RDCNs), especially as circuits and packet networks operate on different timescales, the paper introduces dynamic buffer resizing and explicit network feedback mechanisms for TCP in RDCNs, along with the Etalon emulator platform, enabling realistic performance testing and showcasing the proposed solutions [11]. A. M.Abdelmoniem et al. addresses TCP incast congestion in datacentres when synchronized short lived TCP flows experience packet loss from lack of buffer space. The cause of this congestion leads to delays and affects latency sensitive applications, the paper introduces SICC, an SDN-based congestion control framework that leverages SDN controllers and hypervisors to manage incast congestion, the approach is tested in simulations and a real testbed, showing reduced latency for short flows [12]. J. Huang et al. proposed a paper tackles TCP Incast in datacentre networks using commodity switches, where synchronized TCP connections cause packet loss and throughput collapse due to buffer overflows in shallow-buffered switches, this work introduces Packet Slicing, which adjusts IP packet sizes to reduce incast probability, it is compatible with various TCP protocols and has shown, through simulation and real tests, significant performance improvements in high-concurrency settings [13]. A. M. Abdelmoniem et al. proposed a paper addresses the issue of excessive delays in datacentres caused by TCP timeout mechanisms, which lead to throughput collapse and negatively affect latency sensitive applications, the paper presents T-RACKs, a lightweight cross-layer approach that enhances flow performance by allowing customized handling of timeout values for different flows, significantly improving flow completion times and link utilization without altering TCP [14].

C. Traffic Scheduling and Load Balancing

Advanced traffic scheduling mechanisms have also been explored to minimize the effects of TCP Incast. This approach reduces the likelihood of congestion caused by large-scale synchronized traffic bursts. S. Zou et al. introduced a paper which presents AP (Adaptive Pacing), a protocol that automatically adjusts the pacing

intervals of TCP flows based on network congestion, demonstrated through simulations and testbed implementation. This approach effectively reduces Incast related issues and improves goodput, especially in high concurrency scenarios in datacentres [15]. W. Bai introduced a new mechanism, Buffer Congestion Control (BCC), that operates with minimal modifications to hardware and effectively manages shared buffer utilization in DCNs to reduce latency and maintain throughput [16].

III. MODELLING

The proposed model examines a scenario where a user requests a large file, such as a video, from a data center composed of multiple servers. The user's request triggers data transmission from several servers, with each server sending a block of the requested file to a load balancer. The load balancer collects the data before passing on the outcomes to the user. However, this process leads to what is commonly referred to as TCP Incast where multiple server responses overwhelm the network's resource and results in congestion, packets drop and lots of retransmission time. Figure 1 shows the topology that is suggested for the work.

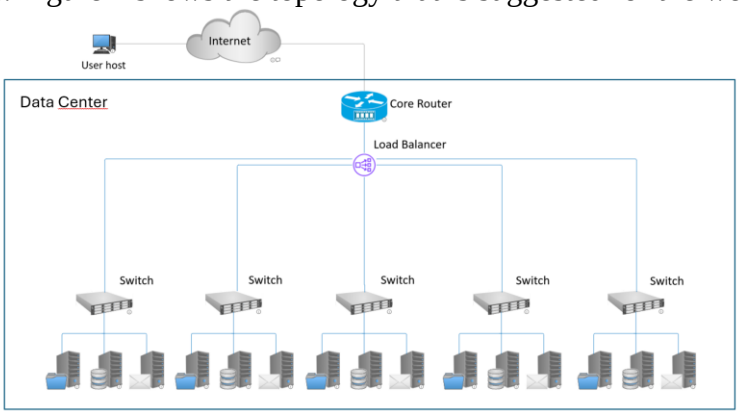

Fig. 1. Model Topology

The model's topology starts with a user's request that travels over the Internet to the data center (i in ms). The path also encompasses the core router within it that adds a latency time of (r in ms) and the load balancer that regulates the requests. The average arrival rate of the requests is λ requests/second, where there are N servers available for processing these requests (N= 5), observing the average processing rate of the servers at μ requests/second. The servers pass their responses through network switches and the round trip incurs an extra delay of roughly (s in ms). At the load balancer before passing the entire data back to the user, using the M/M/1 queueing model [17], the delay per server (LD) can be calculated as:

$$LD = 1 / (\mu - \lambda \text{ per server}) \qquad (1)$$

When the servers send their responses simultaneously, every server transmits information at rates (R) and the total bandwidth (B), the total sending rate is N*R, which exceeds the B (bandwidth limit), this congestion causes packet loss and retransmissions, introducing an additional delay called TCP Incast delay:

$$\text{Congestion Occur when } N * R > B \qquad (2)$$

$$\text{Congestion Factor} = (N * R) / B \qquad (3)$$

The TCP incast delay (TID) can be modeled as:

$$\text{TID} = \begin{cases} \alpha \left(\frac{N*R}{B} - 1\right) & if\ N*R > B \\ 0 & if\ N*R \leq B \end{cases} \quad (4)$$

Where α: Scaling constant (ms), representing the retransmission penalty per unit congestion.

The total round trip delay is the sum of the forward and backward path delays. Forward path delay includes the Internet latency (i in ms), router to load balancer latency (r in ms) and load balancer processing delays (LD in ms) and load balancer to switches (s in ms):

$$\text{Forward path delay} = i + r + LD + s \quad (5)$$

The backward path delay can be calculated by summing the Internet latency (i ms), router to load balancer latency (r ms), switches (s ms), and TCP incast delay:

$$\text{Backward path delay} = i + r + s + \text{TID} \quad (6)$$

Therefore, the total round trip delay can be calculated by:

$$\text{Total round trip delay} = \text{forward path delay} + \text{backward path delay} \quad (7)$$

$$\text{Total round trip delay} = 2i + 2r + 2s + 1 / (\mu - \lambda \text{ per server}) + \alpha ((N*R)/B-1) \quad (8)$$

IV. RESULTS AND DISCUSSION

A higher processing rate can help eliminate bottlenecks and prevent performance degradation as peaks loads. In addition, dynamic scaling methods that adjust μ in response to real time load conditions further improve system responsiveness. By applying request throttling measures one can limit the number of requests coming to the system in order to avoid an overload. Real time load balancing algorithms which aim at distributing the requests depending on the load at that time can also help reduce the time delays and still help increase the system's performance during the high traffic times. The figure 2 shows how arrival rate (λ) affects load balancer delay in a data center configuration. The horizontal axis presents the arrival rate (λ) in the measure of requests per second and the vertical axis is the delay in seconds experienced by the load balancer. Change in response time of the curve manifests clear non-linear relationship where the delay rises exponentially as the arrival rate nears the load balancer's utilization limit. This behavior corresponds to the model of M/M/1 queues as the numerator approaches μ, so that the denominator decreases, and the delay escalates rapidly. This underscores the need to keep λ as low as possible, but always below μ to reduce on the waiting time as much as possible.

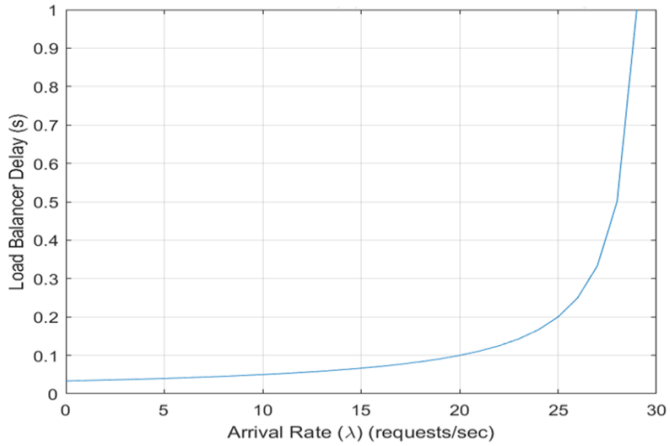

Fig. 2. Effect of Arrival Rate on load balancer delay

The figure 3 illustrates the effect of processing rate (μ) on the load balancer delay in a data center. Here, on the x-axis number of requests per second as processing rate μ is represented and on the y-axis, the load balancer delay in seconds is represented. We are also able to identify from the plot of the graph a direct proportionality between the processing rate and the load balancer delay. At the lowest processing rate, the requests' work rate is slow, which leads to bottlenecks and thus higher delays. However, the rate increases ($30<\mu\leq40$), delays drop off drastically, and at high rates (>40), although the delays continue to decrease, they are at a diminishing rate, proving efficient system operation.

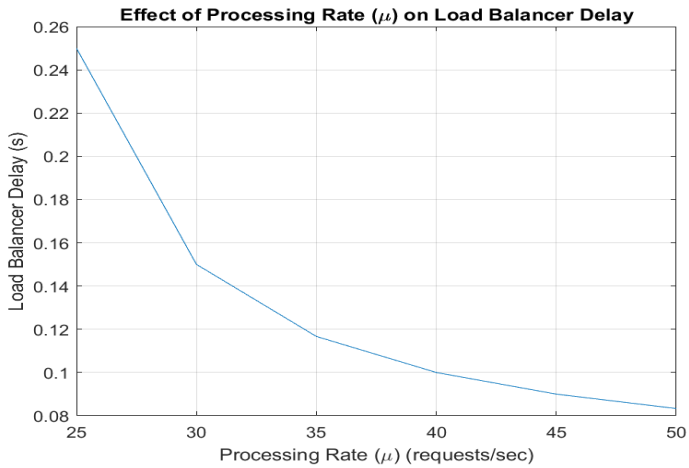

Fig. 3. Effect of processing rate on load balancer delay

The figure 4 also shows how TCP Incast delay changes with the number of servers (N) and bandwidth (B) in Data center network. As the number of servers increases the TCP Incast delay increases at all bandwidth conditions. This behaviour is expected since several servers responding concurrently raises the risk of network contention. It has been observed that the delay is substantially more for the lower bandwidth (for instance, 20 Mbps) and increases almost directly with a number of servers. For the higher bandwidth, say 70 Mbps, the delay rises with the moderate slope and so there is a clear illustration

of how the larger bandwidth can reduce congestion impacts. Delay is very much affected by the amount of bandwidth available. For example, increased bandwidths reduce a delay because of higher transmission capacity that can handle the impact of simultaneous server responses. For lower bandwidths, the increase in the number of servers puts higher load on the network and consequently the delay increases rapidly.

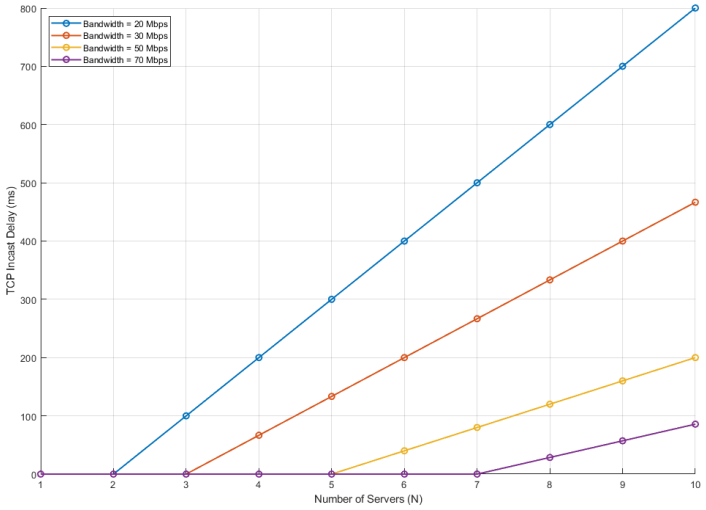

Fig. 4. Effect of processing rate on load balancer delay

V. CONCLUSION

The results of this work showed that to reduce the impacts of TCP Incast, bandwidth must be increased, especially when *N* is small. Nevertheless, for a large number of servers, it is likely necessary to employ further approaches like the sophisticated load distribution, refined algorithms for connection control and pacing mechanisms at the level of the server. Such measures can lead to a reduction of further responses from other servers at the same time, giving a possible result to eradicate the delay and prevent congestion of the networks. As argued earlier in this analysis, networks always have the capacities, bandwidths, and server usages need to be optimally used in data centers.

REFERENCES

[1] J. Ousterhout, "It's Time to Replace TCP in the Datacenter," Stanford University, Jan. 18, 2023. [Online]. Available: https://homa-transport.atlassian.net/wiki/spaces/HOMA/overview

[2] C. Raiciu, S. Barre, C. Pluntke, A. Greenhalgh, D. Wischik, and M. Handley, "Improving Datacenter Performance and Robustness with Multipath TCP," in *Proceedings of the ACM SIGCOMM 2011 Conference*, Toronto, Canada, Aug. 2011, pp. 265–276. [Online]. Available: https://doi.org/10.1145/2018436.2018472

[3] Liu, J. Huang, Y. Zhou, J. Wang, and T. He, "Task-aware TCP in Data Center Networks," in *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*, Atlanta, GA, USA, Jun. 2017, pp. 1417–1424. [Online]. Available: https://doi.org/10.1109/ICDCS.2017.175

[4] T. Yu and X. Qiu, "STCC: A SDN-oriented TCP congestion control mechanism for datacenter network," *IET Networks*, vol. 10, no. 1, pp. 13–23, Jan. 2021. [Online]. Available: https://doi.org/10.1049/ntw2

[5] Q. I. Ali, "Design and implementation of an embedded intrusion detection system for wireless applications," IET Information Security, vol. 6, no. 3, pp. 171–182, 2012, doi: 10.1049/iet-ifs.2011.0152.

[6] Q. I. Ali, "Performance Evaluation of WLAN Internet Sharing Using DCF & PCF Modes," International Arab Journal of e-Technologies (IAJET), vol. 1, no. 1, pp. 38–45, 2009.

[7] S. L. Qaddoori and Q. I. Ali, "An embedded and intelligent anomaly power consumption detection system based on smart metering," IET Wireless Sensor Systems, vol. 13, no. 2, pp. 75–90, 2023, doi: 10.1049/wss2.12054.

[8] H. M. Mohammed and Q. I. Ali, "E-proctoring systems: A review of design techniques, features, and abilities against threats and attacks," Quantum Journal of Engineering, Science, and Technology, vol. 3, no. 2, pp. 14–30, 2022.

[9] H. Lim, W. Bai, Y. Zhu, Y. Jung, and D. Han, "Towards Timeout-less Transport in Commodity Datacenter Networks," in *Proc. of the 16th European Conf. on Computer Systems (EuroSys '21)*, 2021, pp. 1–16. [Online]. Available: https://doi.org/10.1145/3447786.3456227

[10] T. Zhang et al., "Rethinking Fast and Friendly Transport in Data Center Networks," *IEEE/ACM Transactions on Networking*, vol. 29, no. 2, pp. 1–13, Apr. 2021. [Online]. Available: https://doi.org/10.1109/TNET.2020.3012556
[11] M. K. Mukerjee, C. Canel, W. Wang, D. Kim, S. Seshan, and A. C. Snoeren, "Adapting TCP for Reconfigurable Datacenter Networks," in *Proceedings of the 17th USENIX Symposium on Networked Systems Design and Implementation (NSDI '20)*, Santa Clara, CA, USA, Feb. 2020, pp. 651–666. [Online]. Available: https://www.usenix.org/conference/nsdi20/presentation/mukerjee
[12] A. M. Abdelmoniem, B. Bensaou, and A. J. Abu, "Mitigating incast-TCP congestion in data centers with SDN," *Annals of Telecommunications*, 2017. [Online]. Available: https://doi.org/10.1007/s12243-017-0608-1
[13] J. Huang, Y. Huang, J. Wang, and T. He, "Adjusting Packet Size to Mitigate TCP Incast in Data Center Networks with COTS Switches," IEEE Transactions on Cloud Computing, 2018. [Online]. Available: https://doi.org/10.1109/TCC.2018.2810870
[14] W. Bai, S. Hu, K. Chen, K. Tan, and Y. Xiong, "One More Config Is Enough: Saving (DC)TCP for High-Speed Extremely Shallow-Buffered Datacenters," *IEEE/ACM Transactions on Networking*, vol. 29, no. 2, pp. 1–10, Apr. 2021. [Online]. Available: https://doi.org/10.1109/TNET.2020.3032999
[15] S. Zou, J. Huang, J. Wang, and T. He, "Flow-Aware Adaptive Pacing to Mitigate TCP Incast in Data Center Networks," *IEEE/ACM Transactions on Networking*, vol. 29, no. 2, pp. 1–14, Apr. 2021. [Online]. Available: https://doi.org/10.1109/TNET.2020.3027749
[16] A. M. Abdelmoniem and B. Bensaou, "T-RACKs: A Faster Recovery Mechanism for TCP in Data Center Networks," *IEEE/ACM Transactions on Networking*, vol. 29, no. 2, pp. 1–16, Apr. 2021. [Online]. Available: https://doi.org/10.1109/TNET.2021.3059913
[17] R. Alshahrani and H. Peyravi, "Modeling and Simulation of Data Center Networks," in *Proc. of SIGSIM-PADS '14*, Denver, CO, USA, 2014, pp. 1–7. [Online]. Available: https://doi.org/10.1145/2601381.2601389