Parallel Routing on Multi-Core Routers for Big Data Transfers *

Ahmet Soran
Computer Science & Eng.
University of Nevada, Reno
Reno, NV 89557
asoran@cse.unr.edu

Furkan Mustafa Akdemir
Computer Engineering
Bilkent University
Ankara, Turkey
fmakdemir@gmail.com

Murat Yuksel
Computer Science & Eng.
University of Nevada, Reno
Reno, NV 89557
yuksem@cse.unr.edu

ABSTRACT

Over the last several years, the deployment of multi-core routers has grown rapidly. However, big data transfers are not leveraging the powerful multi-core routers to the extent possible, particularly in the key function of routing. Our main goal is to find a way to use these cores more effectively and efficiently in routing the big data transfers. We propose a novel approach to parallelize data transfers by using each core in the routers to calculate a separate shortest path. For each core, we generate a different "substrate" topology in order to allow shortest path calculations to find a different end-to-end (e2e) path. By abstracting a different topology for each core, we indirectly steer each core to calculate a different e2e path in parallel to each other. The e2e big data transfers can use these shortest paths obtained from each substrate topology to increase the total throughput. We present an initial evaluation of the concept.

Categories and Subject Descriptors

C.2.3 [**Network Operations**]: Network Management

Keywords

Multi-Core Routers, Multi-Path Routing, Load Balancing

1. INTRODUCTION

As datacenters are gaining more importance in the Internet, the need for bulk data (a.k.a. "big data") transfers intra- and inter-datacenter transfers is more paining. Most of the end-to-end (e2e) sessions are now going through a datacenter, and thus, the performance of large "big data" intra- and inter-datacenter transfers is very important to the overall Internet experience. More essentially, these big data transfers are crucial to the operation of the data centers due to the needs for large bulks of transfers for maintenance and backup of the data centers [3].

A successful approach to address the big data transfers is to use multi-path routing [6, 4]. The key focus of these techniques is to diversify and spread the paths available to the e2e transport while satisfying various constraints such as delay or loss. Since the problem is too complex most multi-path routing work boiled down to pre-computed techniques with heavy computations. Further, they typically involve non-shortest path calculations requiring considerable updates to legacy routers. Recently, these multi-path routing techniques, however, proved to be useful for scaling up the e2e reliable transfers. The paths generated by these multi-path routing methods were adopted and TCP sessions were successfully parallelized with effective solutions [1, 2]. However, these e2e transfers are still yet to utilize multi-core CPUs available in most routers.

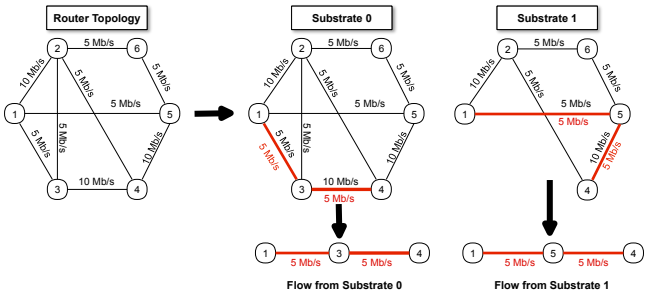


Figure 1: Motivating scenario with two cores.

We propose a "parallel routing" framework that explicitly considers *multi-core routers* and employs *shortest-path calculations* only. The basic idea is to virtually slice the router topology into "substrate" topologies and assign them to a separate router core, which calculates a shortest path on the assigned substrate. Rather than solving the multi-path routing problem all at once, our approach transforms it into two subproblems: (i) slicing out substrates from the router topology so that the collection of the shortest paths on each substrate diverse and non-overlapping e2e paths, and (ii) calculate shortest paths on each substrate. Since the latter problem is already being handled in legacy routers, our approach can easily be adapted to current routers if the former problem is solvable. In one point of view, our approach transforms the multi-path routing problem into a topology/substrate generation problem.

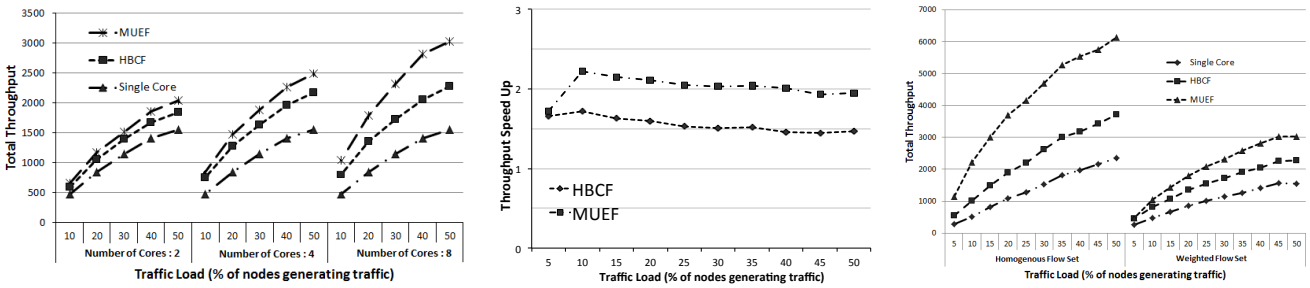
2. PARALLEL ROUTING

Parallel routing aims to provide multiple shortest-path routes over different substrates of the topology, as shown in Figure 1. The main idea is that different slices (sub-

*This work was supported in part by NSF award 1321069.

CoNEXT Student Workshop'13, December 9, 2013, Santa Barbara, CA, USA.

http://dx.doi.org/10.1145/2537148.2537159.



(a) Effect of cores (b) Speedup for 8 cores (c) Effect of traffic pattern

Figure 2: Effect of various parameters on parallel routing performance

strates) of the router topology are given to each core and the e2e data transfer is split onto shortest paths calculated on each substrate topology. It is up to the e2e transport protocol's decision to which one of these paths from the substrate topologies to use with what rate. Figure 1 illustrates a motivating scenarios where there are two virtual substrates are produced. Substrate 0 is equivalent to the real router topology, whilst Substrate 1 is generated by removing node 3 from Substrate 0. Each substrate's shortest path is different and the two collectively yield a total of 10Mb/s throughput from node 1 to 4. This could be done via 2 cores on each router. Note that Substrate 0 is the current single shortest-path approach which only yields 5Mb/s.

The design goal of parallel routing is to *generate substrates that yield the most diverse and non-overlapping shortest paths possible*. For a network with E edges, 2^E different substrates could be generated with no constraints on nodes or connectedness of the network. This is clearly a hard problem, and thus we develop intuitive heuristics to generate the substrates. To assure all-to-all connectivity, Substrate 0 should be the complete router topology. When generating the subsequent substrates, however, some nodes or edges are going to be omitted. A simple heuristic step could be to omit the nodes/edges that are being used the most by the shortest paths in the existing substrates. We try two approaches:

Graph-Based: The graph-based heuristic could be done purely based on the graph properties of the topology. When generating a substrate, we remove the node(s) that are the most "central" to the topology of substrate 0. This increases the chance of finding a non-overlapping shortest path in the new substrate. Betweenness centrality is a graph metric that measures the number of all pairs shortest paths traversing a node. Thus, we name this heuristic as Highest Betweenness Centrality First (HBCF).

Flow-Based: The graph-based properties cannot capture the dynamism in the network traffic. Thus, designing a heuristic that considers the *current* utilization of an edge is useful to adapt the substrate generation process to traffic dynamics. We count the active flows traversing each edge and omit the most used edge in substrate i to generate substrate $i + 1$, which leads us to Most Used Edge First (MUEF).

Results: We evaluated our heuristics on the Sprintlink topology of the Rocketfuel [5] dataset. We compared the heuristics against the single shortest path case according to the total throughput achieved from the generated substrates. We generated the flows based on a uniform or gravity-based traffic matrices, named as "homogeneous" and "weighted" flow sets, respectively, in our results. To generate the gravity-based weighted flows, we used the population of the routers' cities, which better models the inter-datacenter traffic. The homogeneous flow set is better for modeling intra-datacenter traffic. We assumed link capacities inversely proportional to the link weights provided by Rocketfuel. We used max-min allocation to determine the e2e rates the flows will attain. We normalized the flow rates based on the smallest flow rate, and, so our throughput results are shown in units.

In the simulations, we tuned HBCF and MUEF so that they omit 10% of nodes and edges, respectively, every time they generate a new substrate. Under weighted traffic, Figure 2(a) shows that parallel routing heuristics achieve higher total throughput in comparison to the single core routing. Also, they outperform even further as the number of cores and the offered load (i.e., the number of source-destination flows) increase.

Figure 2(b) shows the speedup our heuristics achieve with 8 cores. MUEF clearly outperforms HBCF since it works at the finder granularity of edges rather than nodes. However, as the number of flows and the network get larger, MUEF is computationally harder. Lastly, Figure 2(c) shows the results when the traffic is homogeneous, which increases the performance difference between the heuristics and the single core cases further, with MUEF speedup exceeding 3.

3. SUMMARY AND FUTURE WORK

We presented a new framework to parallelize e2e big data transfers and developed simple heuristics to solve the substrate topology generation problem within the framework. Our initial evaluations showed that parallel routing heuristics over multi-core routers achieve higher total throughput and perform better under uniformly distributed heavier traffic loads. It will be interesting to explore various tradeoffs in designing heuristics. For instance, rather than basing the substrates to Substrate 0, using a cumulative approach which removes the nodes or edges from Substrate i to generate Substrate i + 1 is worthy to try. Because, new non-overlapping shortest paths in Substrate i+1 should consider the cumulative set of all existing shortest paths in Substrate 0 through Substrate i.

We will experiment our heuristics with e2e reliable transport protocols on a larger test base. Our initial evaluations showed that higher cores in the routers yield sizably more throughput. We considered routers up to 8 cores, but the number of cores at which the marginal aggregate throughput saturates remains to be explored. Further, heterogeneity of the number of cores across routers is another dimension that is worthy of exploration.

4. REFERENCES

[1] S. Barre, O. Bonaventure, C. Raiciu, and M. Handley. Experimenting with multipath tcp. In *Proceedings of ACM SIGCOMM*, pages 443–444, 2010.

[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 ACM SIGCOMM*, pages 266–277, 2011.

[3] N. Laoutaris, M. Sirivianos, X. Yang, and P. Rodriguez. Inter-datacenter bulk transfers with netstitcher. In *Proceedings of ACM SIGCOMM*, 2011.

[4] S. Nelakuditi and Z. Zhang. On selection of paths for multipath routing. In *Proceedings of IEEE/IFIP International Workshop on Quality of Service (IWQoS)*, pages 170–186, 2001.

[5] N. Spring, R. Mahajan, and D. Wetherall. Measuring isp topologies with rocketfuel. In *Proc. of ACM SIGCOMM*, pages 133–145, 2002.

[6] S. Vutukury and J. Garcia-Luna-Aceves. Mdva: A distance-vector multipath routing protocol. In *Proceedings of IEEE INFOCOM*, 2001.