Cut-Through Network Designs for High-Throughput E2E Networking

MARAT ZHANIKEEV[1,a)]

Abstract: In the age of Big Data, emulation of circuits on top of packet switching is once again a reasonable option from the viewpoint of performance. The biggest feature of such circuits is that the cut-through mode in Ethernet or optical networks is enforced end-to-end. This paper improves on the idea of circuits with a more general formulation which adds network design to the traditional scheduling problem.

Keywords: cut-through, circuit emulation, BigData networking, cloud internetworking, e2e networking

1. Introduction

This paper is an upgrade on the earlier work with roughly the same scope published at [1]. This paper continues the discussion on data center interworking in clouds and starts a new discussion on network topologies which can support circuit emulation without the necessity to build a brand new infrastructure – this concept is referred to as *circuits-over-packets* emulation.

Circuits vs packets is a very old subjects arguably first published in [4]. Roughly at that time, it was proven statistically that packet switching is more efficient. The evaluation was performed on multiplexing multiple flows each modeled with an ON/OFF model where OFF periods are slightly longer than ON periods. Since that time, the process of migrating to all-IP networks started slowly but lately picked put the pace and today one can assume that almost 100% of networking uses packet switching. Circuit switching is still present in some legacy networking technologies such as SONET/SDH, ATM, MPLS, etc [8].

Migration to packet switching brought new problems. The most important of which is the problem of *contention* – a natural state where multiple concurrent packet flows have to contend to get access to the uplink. As the next section shows, contention is not easy to resolve in shared environments. By extension, the *cut-through* mode in the title of this paper crucially depends on the ability to enforce exclusive access to uplink.

One can say that circuits are having a small comeback, especially in the area of BigData networking where it is necessary to squeeze out all the physically possible performance margins [1]. Naturally, given that global networks has only recently migrated to all-IP networks, the circuits here are not physical but are emulated over packets. Enabling the cut-through mode is switching devices is very important to the success of such emulations.

[1] Computer Science and Systems Engineering
Kyushu Institute of Technology
Kawazu 680-4, Iizuka-shi, Fukuoka-ken, 820–8502 Japan
[a)] maratishe@gmail.com

The cut-through discussion itself is not new. It is common to compare it to the alternative – *store-and-forward*, where it is repeatedly stated that the former is 10-15 times faster than the latter [5]. There is a small volume of research that considers cut-through mode for clouds, specifically for networking inside DCs [6].

Contention is also common in modern optical networks, where Optical Burst Switching (OBS) is becoming the de-facto standard [9]. Note that the justification here is the same as was presented for the old *circuits vs packets* argument – OBS is the most efficient way to operate an optical network. As a parallel research topic, wavelength switching is also discussed [10]. Note that wavelength switching is, but definition, a type of circuit switching. All the findings in this paper that pertain to circuits are also applicable to wavelength switching.

When focusing on cloud internetworking – the main final destination for the circuits in the paper – one cannot miss the active research on Fiber Channel over Ethernet (FCoE) technology [11]. The technology is currently considered the best candidate for future intra-DC networking. Standards process is also active under the code name T11 [12]. Note that this research is not related to circuits. In fact, FCoE is functionally the same as Ethernet and therefore suffers from the same problems as are found for the traditional Ethernet. This means that FCoE literature and recent research on cut-through inside DCs [6] do not overlap.

This paper continues the discussion on circuits. First, the *cut-through* mode is explained in the context of L2/L3 QoS provisioning. Then, traffic engineering problems are defined for all the major technologies including the proposed circuits. These technologies are then analyzed in simulation over a single line. Finally, the paper considers network topologies over which circuit emulation is feasible at current level of technology.

2. Cut-Through on End-to-End Paths

The benefit of the *cut-through* mode is not a secret. Basically any switching equipment can be viewed as a device which can

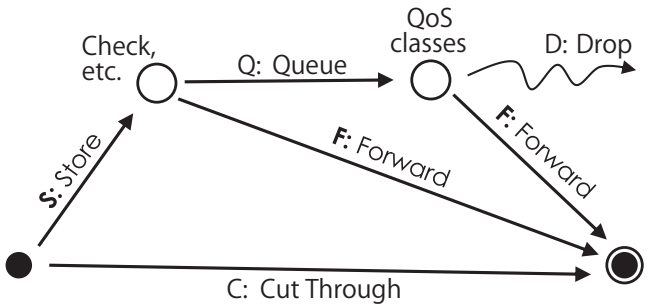

Fig. 1 Cut-Through as one of the possible modes a switch can offer to a packet.

only be in one of two modes at any given point of time: *cut-through* or *store-and-forward* [5]. In the cut-through mode, the device does not store the packet but transmits it to the next destination immediately. In store-and-forward mode, the entire packet has to be received first, placed into a queue, and only then can be transmitted to the next destination. The obvious difference here is the time it takes at least to receive one complete packet. Note that there is a small body of literature which attempts to exploit the benefit of the cut-through mode in data center networks [6].

Now, the above distinction is a bit more complicated when the details are revealed. You can refer to the full description of the modern cut-through mode at [5]. This paper only briefly summarizes it.

Clearly, even in the cut-through mode the packet cannot me transmitted to the next destination without at least reading the L2 header of the arriving packet. This means that the device has to receive at least several first bytes before making a routing decision and starting to transmit the packet. In terms of implementation, the device implements a short buffer for each packet in order to be able to make such low-level routing decisions. However, note that the delay is very small given that the buffer never exceeds several bytes.

However, as is pointed out in [5], the cut-through mode continues to be developed into an interesting technology. Apparently, major device manufacturers like Cisco have started implementing a *tradeoff* between increasing the cut-through buffer by 1-2 bytes and the routing flexibility such an increase can bring. Note that such a technology is fully compatible with legacy packet structure and all the traditional headers. The cut-through part of the L2 switch simply converts the higher-level routing table into a structure which reveals the above tradeoff, and the cut-through processor then makes the necessary decisions for individual packets.

While this paper cannot accommodate the full discussion of the above tradeoff, Fig.1 helps by modeling all the alternatives to the cut-through mode. Note that store-and-forward is the larger name for all the alternatives but also is the specific second-best choice after the cut-through. In this case the packet is stored but, if the queue is completely empty, is transmitted into the network immediately. All the alternative modes further up in the model gradually cause more and more delay for each packet.

Now, why is this important? Fig.2 is a standard way to represent the problem of traffic engineering, or Open Shortest Path First (OSPF) algorithm specifically [7]. Obviously, when many

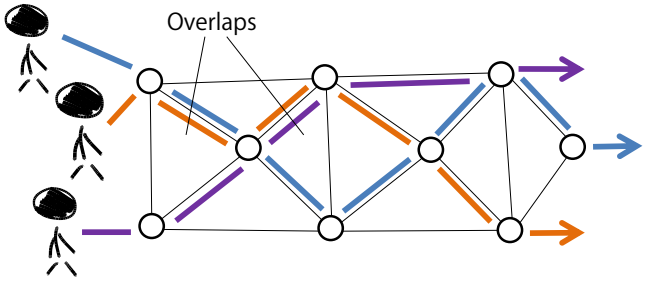

Fig. 2 Cut-through in the context of traffic engineering.

users access the same network at the same time, some paths are bound to overlap, at least partially. If the connections on such paths are concurrent, then *cut-through* becomes physically difficult to achieve given that packets from two or more connections come at mixed rates and, therefore, disable the cut-through mode for each other. Ability to implement cut-through even in such conditions is the key point of this paper.

A note is due on *virtual networking*. While many people draw an equal sign between Software Defined Networking (SDN) and Network Virtualization (NV), these are different viewpoints. In fact, SDN without NV can potentially implement circuits in software – for example, as part of a DC-DC high-throughput networks [6]. The problems of NV, specifically the per-packet cost incurred by virtualization, have been reported in [2].

3. The TE Problem for Circuits

It is difficult to compare OSPF formulations across different technologies, mostly because formulations of utility functions are different. This section presents a new notation for OSPF problems which emphasizes the mapping part without focusing on the utility function. This helps by making it possible to compare OSPF problems in various practical implementation scenarios while the utility definition and algorithms used to find the optimal solution are placed outside of the notation.

Let us define a *unit demand* as source s, destination d, volume v, time t, and sometimes optical wavelength λ. The demand tuple for demand item i can be written as

$$T_i = \langle s, d, v, t \rangle. \quad (1)$$

Repeating the earlier statement, the points of this notation is to define the mapping between demand and path mapping while utility function remains undefined in this new notation. However, all the below formulations can be solved using the standard OSPF problem in [7].

Based on the above demand tuple, the core formulation is in the form of *in/demand tuple → path mapping* and retains this form for all the distinct technologies below.

Traditional Ethernet formulation is the most standard case written as:

$$T_i = \langle s, d, v \rangle \rightarrow \langle s, a, b, ..., d \rangle, \quad (2)$$

where $a, b, ...$ are intermediate nodes. Note that demand tuple lacks the time t which is because traditional OSPF is not aware of time. In practice, such optimizations are conducted at regular time intervals, changing configurations for each switch based on the new mapping.

Optical Networks without Switching is possibly the simplest notation written as

$$T_i = \langle s, d, v \rangle \rightarrow \langle s, \lambda \rangle, \quad (3)$$

where λ is the wavelength of the lightpath. Again, this formulation does not take time into consideration. Also, it should be obvious from notation that capacity of such networks is fairly low. For example, with 24 or 48 wavelengths, depending on the hardware, there can only be 24 or 48 concurrent flows.

Optical Networks with Switching can solve the problem of low capacity by intoducing wavelength switching:

$$T_i = \langle s, d, v \rangle \rightarrow \langle s, \lambda_s, \lambda_a, \lambda_b, ... \rangle, \quad (4)$$

where $\lambda_a, \lambda_b, ...$ are different wavelengths assigned at intermediate nodes along the path. Obviously, this has a great potential to increase the overall capacity of the network.

Finally, the newly introduced **Tall Gate** model can be written as:

$$T_i = \langle s, d, v, t_1, t_2 \rangle \rightarrow \langle s, \lambda, t \rangle, \quad (5)$$

where t_1 and t_2 define the intended time frame for a circuit and t is the time that comes from the best solution and guarantees exclusive access to the line. The mapping without λ applies to Ethernet networks. In fact, this shows how optical lines are simply multiple Ethernet lines from the viewpoint of this formulation. Also, as far as the Tall Gate model is concerned, this is exactly the case. In fact, the Tall Gate model completely guarantees exclusive access because lines on the highway do not overlap or interact with any other lines, by definition.

4. One Hop Performance

This subsection compares performance across all the practical designs defined at the beginning of this section. Single 10Gbps Ethernet line is accessed by multiple traffic sources in all models. Traffic sources themselves are defined using the hotspot model described in [1]. Each distribution is first randomized and passed to each design for decision making on how to perform the bulk transfer. The single metric on the output is the total time it takes for all traffic sources to complete bulk transfers.

Practical numeric setup is as follows. Big Data size ranges (maps hotspots to this range) are *10M..100M*, *10M..100M*, *100M..500M*, *100M..1G*, *1G..10G*, *10G..50G*, and *10G..100G*, all in bytes. The ranges are picked by hand as representative of various environments. The three metrics in the table above have to be put to numeric form. Interference is represented by the set $\{10, 20, 30, 40, 50\}$, representing percentage of decrease in throughput. Zero interference is translated as 0%, low is selected from values below 30% inclusive, and high is selected from values above 30% inclusive. Overhead is selected from the set $\{10ms, 100ms, 1s, 10s, 30s\}$, defining the time period it takes to negotiate and establish a circuit. Similarly to interference, zero is the special case of $0ms$, and high and very high are selected from values up to or above $1ms$, both inclusive. Isolation is selected from the set $\{20, 40, 60, 80, 100\}$ representing percentage of decrease in throughput. Again, *no interference* is the special case of 0% while *yes* translates into selection of one of the values from

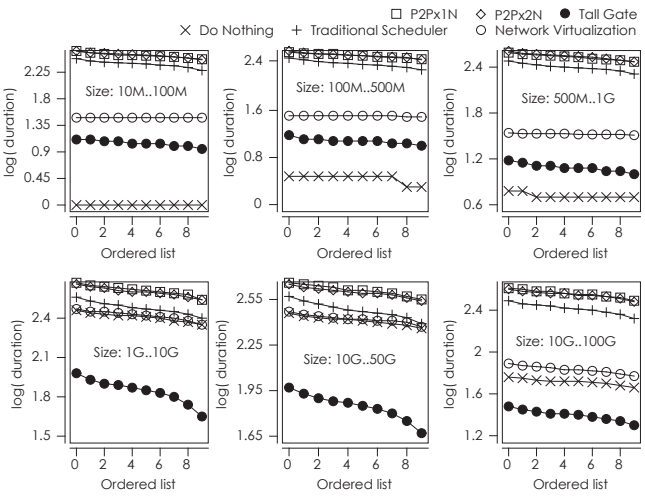

Fig. 3 Performance for all models, separated in separate charts for each size range. The Tall Gate models is marked as filled-in bullets.

the entire set. All the selections from the above sets (or subsets when low, high or very high) is done randomly.

Note that these values are compiled from raw logs observing each configuration metric in practice. Granted the split in each set is arbitrary and was performed by hand, the range of values comes from practical experience.

The only result metric is the *duration* of all bulk transfers from a given hotspot distribution. Randomness in configuration parameters and order of bulks in hotspot distributions is removed by having 1000 runs for each design, each with a new hotspot distribution (but same configuration). Since each run results in 4 values for 4 sets of hotspot distributions, each run is represented by the average over the four duration values. Avoiding crowded plots (1000 runs = 1000 points), the resulting values for each method are listed in decreasing order and converted into 10 points for each design, where each point is an average of 100 points. The result of this procedure is a 10-point curve for each design on each plot.

Fig.3 shows the results of analysis. Plots are generated using the above method, while each plot represents a separate size range. We can see that the *Do Nothing* design does well until the bulk exceeds 1Gbyte range. *Network Virtualization* also does relatively well up to 1G past which it performs about the same as *Do Nothing*. The *Tall Gate* model is 2nd best up to 1Gbytes and the best for all the size ranges above that.

Note that in 1G..50G range (to bottom plots), the Tall Gate model results in an interesting curve in which we can find sources at both extremes (much better and much worse) while majority of sources experience the average performance. Given that vertical scales on all plots are logs, this range indicates a big difference between the two extremes. This is potentially a good indicator that the distribution of results under the Tall Gate model reflected the distribution of bulk in the hotspot model. The issue here is fairness, where it desirable that fairness across traffic sources would be distributed in accordance with their relative bulk.

The remaining designs all resulted in bad performance which did not improve even for large size ranges. Note that it did not help much for the P2P design to have a separate network for control traffic because the time delay itself from negotiating the cir-

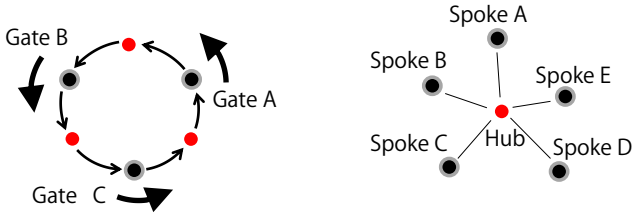

Fig. 4 Two simple topologies encountered in cloud internetworking.

cuits made a non-negligible contribution to the overall duration.

Granted the results in Fig.3 are fuzzy (for example, performance for some models suddenly improves for the biggest size range) due to randomness and irregularity in size ranges, the overall performance as well as the effect of bulk size on performance is evident. The take-home lesson here is that performance varies depending on the size range of Big Data. This means that for some cases, it might even be better not to implement circuits – the case for all the small size ranges. On the other hand, if we know that our sources have to exchange considerable bulk, it might be beneficial to implement one of the circuit designs and benefit from increased throughput.

5. Network Designs in Clouds

Network designs for circuits is part of the larger topic on L2 vs L3 QoS provisioning [3]. More on the various technologies that exist around this topic can be found in [1]. This section summarizes some of the main points in this discussion and presents the two specific topologies which are feasible in clouds today, under the requirement of providing legacy compatibility. Compatibility here simply means that circuits have to be implemented on top of existing networks. In other words, if a cloud provider wants to implement circuits across several of its data centers, it needs to make sure that it is the only party that has full access to the switching equipment.

Let us consider the toolkit available to use at L2 and L3 of switching [1]. Note that the below technologies do not really distinguish between L2 and L3. Circuits are the same in theory, however, in practice most switches today implement the cut-through mode only at L2.

Policy Edges are switching devices that are officially the entry or exit points of a given QoS policy. They are important because, by definition, QoS can only be enforced on the inside of these devices. In respect to circuits, it is important that contention is avoided in incoming traffic while outgoing traffic can be properly policed.

Traffic Shaping is one way to maintain the cut-through mode. If traffic on a given port is shaped to stay below a given rate, then cut-through mode can be guaranteed. Otherwise, the device will have to switch into the store-and-forward mode.

Taking into consideration the above two technologies, it is clear that circuits cannot be implemented in the wild (global network). However, it is still possible for implement circuits for small-scale networks connecting DCs in clouds. Fig.4 shows two specific topologies that are found in practice today.

Ring topology is easily found in optical networks. For example, the JGN network is a ring. A cloud provider would normally have a contract with the network that would allow it access to entry points at several physical locations. Since the network is a ring, uplink and downlink are separated and can be used in opposite directions. However, it is still convenient to describe the ring as a one-directional network.

Hub-and-Spokes topology is also possible provided a cloud provider agrees with the network provider that the hub is physical isolated and fully dedicated to networking between data centers. Note that OBS/OPS networks use this topology automatically but suffer from contention under multiple users [9].

When used in the wild, the Virtual Network Embedding (VNE) problem can help provide isolation across multiple virtual networks, provided in the end they are isolated using L2 policing and strictly enforce the cut-through mode [13]. The VNE problem itself was originally created for virtual networks. However, the core of VNE is a mathematical optimization problem which is abstract from its physical implementation.

The main reason why VNE is compatible with circuits is that requests in VNE are full graphs. Therefore, provided physical isolation is included as one of the constraints, VNE can output circuit-compatible solutions. Time separation is also a valid option.

References

[1] M.Zhanikeev, “Circuit Emulation for Big Data Transfers in Clouds”, Networking for Big Data, CRC (in print), 2015.
[2] M.Zhanikeev, “Experiences from Measuring Per-Packet Cost of Software Defined Networking”, IEICE Technical Report on Service Computing (SC), vol.113, no.86, pp.31–34, June 2013.
[3] M.Zhanikeev and Y.Tanaka, “Analytical Models for L2 versus L3 QoS Provisioning”, IEICE Technical Report on Photonic Networks, Vol.112, No.276, pp.13–18, November 2012.
[4] J.McDonald, Fundamentals of Digital Switching. Applications of Communications Theory, Springer, 1983.
[5] “Cut-Through and Store-and-Forward Ethernet Switching for Low-Latency Environments”, Cisco White Paper, 2014.
[6] G.Wang, D.Andersen, M.Kaminsky, K.Papagiannaki, T.Ng, M.Kozuch, M.Ryan, “c-Through: Part-time Optics in Data Centers”, ACM SIGCOMM, pp.327–338, October 2010.
[7] B. Fortz and M. Thorup, “Internet Traffic Engineering by Optimizing OSPF Weights”, INFOCOM, pp.519–528, March 2000.
[8] Harry G. Perros, Connection-Oriented Networks : SONET/SDH, ATM, MPLS, and Optical Networks. Wiley and Sons, 2005.
[9] R.Jankuniene, P.Tervydis, “The Contention Resolution in OBS Network”, Elektronika ir Electrotechnika, vol.20, no.6, pp.144–149, 2014.
[10] Hui Zang, J.P.Jue, L.Sahasrabuddhe, R.Ramamurthy, B.Mukherjee, “Dynamic Lightpath Establishment in Wavelength-Routed Networks”, IEEE Communications Magazine, vol.39(9), pp.100–108, 2001.
[11] S.Gai, Data Center Networks and Fibre Channezl over Ethernet (FCoE). Lulu.com, 2008.
[12] INCITS T11 Standardization Body. [Online]. Available: www.t11.org/FCoE (February 2015)
[13] I.Houidi, W.Louati, D.Zeghlache, “A Distributed Virtual Network Mapping Algorithm”, International Conference on Computers and Communications (ICC), pp.5634–5641, 2008.