15-780 Project
Use ILP to Evaluate Indirection Scheduling Techniques for Hybrid Circuit/Packet Networks

Conglong Li (andrew ID conglonl), Thomas Kim (andrew ID tskim)
{*conglonl*,*thomas.kim*} @*cs.cmu.edu*

1 Introduction

Traditionally, circuit switching and packet switching have been considered as alternative design choices, each with their own advantages and disadvantages. Packet switches are efficient at multiplexing traffic across a large number of ports, while circuit switches can often service higher line rates in a more cost-effective manner, especially when combined with optical links. However, as optical circuit switching technology advances, the reconfiguration time (i.e., the amount of time it takes to alter the input-to-output mapping) is swiftly decreasing, thus increasingly blurring the division between the packet and circuit regimes.

Indeed, a range of new datacenter switch designs take advantage of this trend, and propose to schedule appropriately large traffic demands via a high-bandwidth circuit switch and handle any remaining traffic with a slower packet switch. However, all recent proposals for such hybrid designs presume the existence of an omniscient scheduling oracle that can compute switch configurations and map traffic to them in an optimal fashion. Recently, Liu *et al.* [7] proposed a hybrid switch scheduling algorithm, Solstice, that is both highly effective at scheduling datacenter-like traffic workloads and has practical computational overheads.

In this project, we improved the Solstice scheduling algorithm by introducing an indirection technique that reduces the number of configurations needed for scheduling. This proposed technique aims to redirect traffic through underutilized links. Using this technique, traffic demand can be served by pairs of existing links, utilizing multi-hop routes to increase usage of underutilized links. We evaluated this technique by both integer linear programming (ILP) formulations and simulations. Results show that indirection techique could reduce total time required to serve 100% demand by reducing the number of configurations needed for scheduling.

This paper is organized as follows. Section 2 discusses Solstice and other related work. Section 3 describes the ILP formulation without indirection technique. Section 4 describes the indirection technique and its ILP formulation. Section 5 presents the evaluation results. Finally, Section 6 summarizes our conclusions.

2 Related Work

Recently, researchers have proposed hybrid datacenter network architectures that offer higher throughputs at lower cost by combining switching technologies. In particular, recent proposals

suggest employing highspeed optical [1, 2, 9] or wireless [4, 5, 10] networks configured to service the heavy flows, while passing the remainder of the traffic through a traditional, relatively underprovisioned packet-switched network.

As technology trends usher in dramatically faster reconfiguration times, the distinction between packet and circuit is blurred, and ever smaller flows can take advantage of a hybrid fabric. This trend will soon allow servicing the bulk of the traffic through a rapidly reconfigurable optical switch [8], leaving a relatively minor portion to be serviced by the packet network [6]. While the potential cost savings that hybrid technologies could realize is large, the design space for scheduling resources in the hybrid regime is not yet well understood. What range of traffic demands can be scheduled for a given switch design and how can this schedule be computed efficiently? These questions are not addressed by the existing switch scheduling literature.

Many of the classical approaches to scheduling for switches with non-trivial reconfiguration delays divide the offered demand into two parts: an initial, heavy-weight component that is served by O(N) highly utilized configurations with significant durations, and a second, residual component that is serviced by a similar number of short, under-utilized schedules. The recent Solstice scheduling algorithm [7] exploits the skewed nature of datacenter traffic patterns to create a small number of configurations with long durations that minimize the penalty for reconfiguration and leave only a small amount of residual demand to be serviced by a low-speed (and lowcost) unconstrained packet switch.

3 Scheduling Without Indirection

Solstice introduces a method of calculating a schedule for the circuit switch and determine what data to send to the packet switch in order to satisfy all demand while maximizing utilization.

3.1 Notation

The switch in use is a **hybrid switch** composed of a circuit switch and a packet switch. The circuit switch has a link rate of r_c bits/second with a reconfiguration time of δ. The packet switch has a link rate of $r_p \ll r_c$, bits/second.

Demand is the amount of data that must be sent between any two ports. It is expressed as a matrix D of size $n \times n$, where rows are sources and columns are destinations. The amount of demand from port a to port b is $D_{a,b} \in \mathbf{R}_0^+$

A **Schedule** is a set of configurations $\{P_1, P_2, \ldots, P_m\}$, durations $\{t_1, t_2, \ldots, t_m\}$, and packet switch utilization matrices $E_1, E_2, \ldots, E_m$. $P_{i_{a,b}}$ is an $n \times n$ binary matrix, where $P_{i_{a,b}} = 1$ indicates that port a can second to port b when the switch is using the configuration P_i. Each row and each column in P_i has exactly one 1. For each configuration there is an associated set of data E_i to be sent via the packet switch during that configuration. $E_{i_{a,b}} = k$ means k bits are sent from a to b through the packet switch instead of the circuit switch during configuration i.

3.2 Considerations

We seek to minimize transmission time, which is equivalent to maximizing utilization. Additionally, we wish to fully schedule existing demand before scheduling new demand to prevent starvation.

We define *total time* to be the amount of time scheduled on the circuit switch plus the amount of

$$\min\left(\sum_{i=1}^{n!} t_i\right) + m\delta$$

subject to:

1) $E + \sum_{i=1}^{n!} r_c t_i P_i \geq D$

2) $\forall i : \sum_{j=1}^{n} E_{i,j} \leq r_p T$

3) $\forall j : \sum_{i=1}^{n} E_{i,j} \leq r_p T$

4) $\forall i : t_i \leq max(D) l_i$

Figure 1: An ILP to find optimal schedules for a hybrid switch.

time spent switching.

$$T = \left(\sum_{i=1}^{m} t_i\right) + m\delta.$$

In the hybrid switch, the packet switch can only be utilized during times when the packet switch is utilized. Thus, all outbound links i can admit the following amount of data:

$$\sum_{j=1}^{m} E_{i,j} \quad \leq \quad r_p T.$$

Inbound links are similarly constrained.

Demand is satisfied when the total data served by the hybrid switch is at least as large as the demand:

$$E + \sum_{i=1}^{m} r_c t_i P_i \quad \geq \quad D.$$

We formulate this problem as an ILP, shown in figure 1.

4 The Indirection Technique

We improved the Solstice scheduling algorithm by introducing a indirection technique that reduces the number of configurations needed for the scheduling. Originally, Solstice schedules each traffic demand by a direct path, which means that a new configuration is required to fulfill a demand that has different sources or destinations. Our technique schedules part of demand traffic indirectly. For example, port a has 5 demand to port b and 20 demand to port c; port c has 20 demand to port b. Solstice will require 3 configurations for these demands. However, we could let port a send all 25 demand to port c and let port c send 25 demand to port b. In this way we reduce one configuration time, which is nontrivial for optical switches.

We introduce a three-dimensional *indirection matrix* to represent all possible one-hop indirections, and use it to extend the original ILP to support indirection. Indirection matrix I has an entry $I_{a,b,c} = v$, where v is the amount of traffic sent indirectly to c from a through b. The amount of traffic sent directly and indirectly over any link must be less than the capacity of the link.

The updated ILP including indirection is shown in figure 2.

$$\min\left(\sum_{i=1}^{n!} t_i\right) + m\delta$$

subject to:

1) $\forall a,b : (D_{a,b} - (\Sigma_c nI_{a,c,b}) - H_{a,b}) + (\Sigma_c nI_{a,b,c} + I_{c,a,b}) <= (\Sigma_i ot_i * P_{i_{a,b}})$

2) $E + \sum_{i=1}^{n!} r_c t_i P_i \geq D$

3) $\forall i : \sum_{j=1}^{n} E_{i,j} \leq r_p T$

4) $\forall j : \sum_{i=1}^{n} E_{i,j} \leq r_p T$

5) $\forall i : t_i \leq max(D) l_i$

Figure 2: Modified ILP for Indirection

5 Evaluation

To evaluate the indirection technique, we implemented the formulations described in Section 3 and 4 , and then used the Gurobi ILP solver [3] to find the optimal solution. To get a nearly optimal solution within a reasonble time, we used a callback function which will terminate the ILP solver if the optimality gap (MIPGap) is less than 0.1% and the current best solution didn't change for 5 minutes.

Although ILP solver gives optimal solution to the scheduling problem, it can't scale since there are O($n!$) different configuratons to be considered for scheduling. To the end of this, we propose a post-scheduling merging heuristic. Given a scheudule computed by a scheudling algorithm (e.g. Solstice), this merging heuristic uses indirection technique to reduce the number of distinct configurations needed, and thus reduce the total time to serve all demand. We received the simulator of the Solstice algorithm from the authors, and introduced our merging techinque into the simulator to compare it with the original Solstice algorithm.

We built 100 skewed matrices (12 hosts) as the input demand matrices. For each input demand matrix, we use ILP solver (with and without indirection), Solstice algorithm, Solstice+Merging to schedule the demand. Figure 3 and 4 depcit the results. Comparing the ILP results, indirection technique reduces the average number of configurations from 8.12 to 4.35, which leads to a 3.125% reduction on the total time. For the simulation results, the merging heuristic helps the Solstice to reduce the average number of configurations from 14.40 to 10.63, which leads to a 2% reduction on the total time. This shows that our merging hueristic provides similar improvement on Solstice as the ILP results. We believe that the merging heuristic could provide increasing improvement at a larger scale since Solstice will use even more configurations to schedule the demand.

6 Conclusion

Recent advances in optical switch technology has created a need for new scheduling algorithms which take advantage of shorter reconfiguration times while maintaining reasonable computational overheads. The recent Solstice scheduling algorithm accomplishes these goals, but does not achieve optimal utilization due to the fact that it only supports traffic sent over a direct link. It is possible to meet traffic demands without utilizing a direct link between each source and destination port

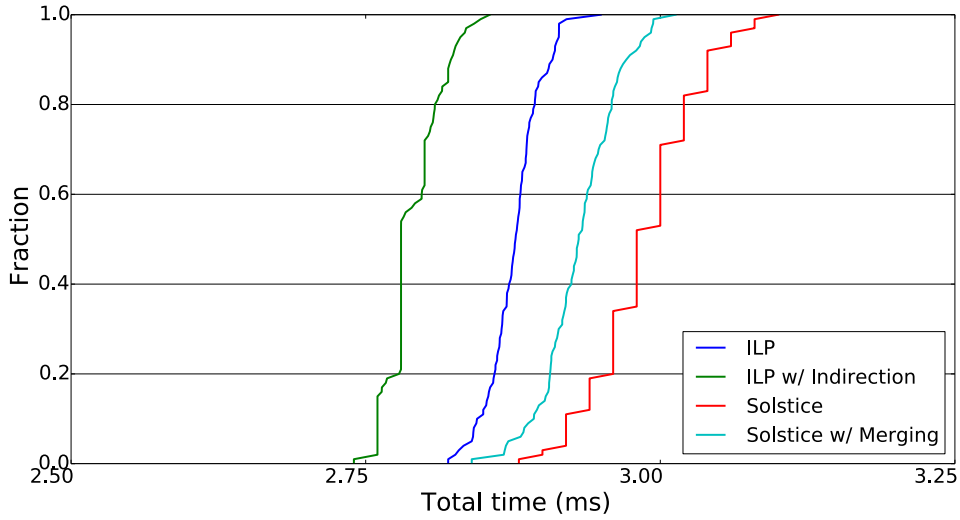

Figure 3: CDF of total time required when serving 100% demand. Average: 2.88, 2.79, 2.99, 2.93.

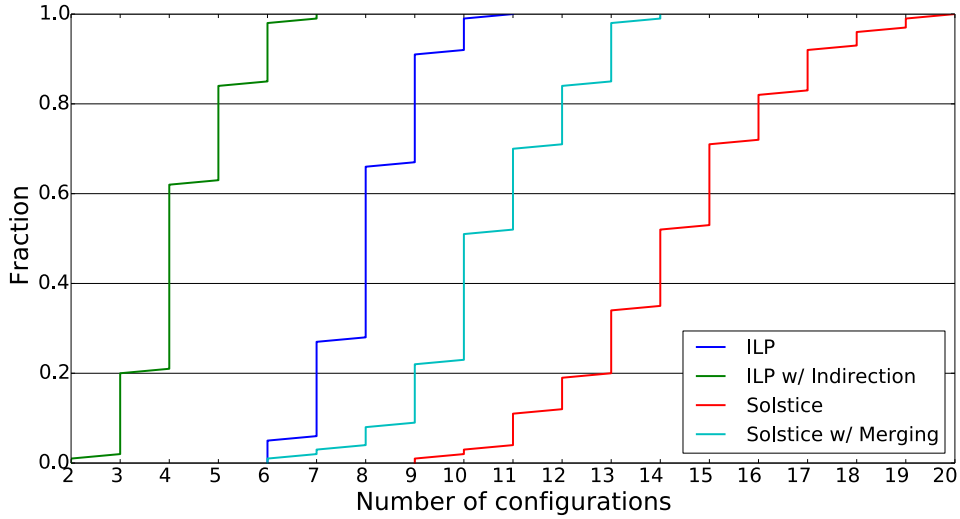

Figure 4: CDF of number of configurations required when serving 100% demand. Average: 8.12, 4.35, 14.40, 10.63.

by routing traffic over multiple links. Utilizing these indirect links can reduce the number of reconfigurations necessary and reduce the amount of unused bandwidth on each link.

In this paper, we have presented an extension to the Solstice packet scheduler allowing for multi-hop indirect serving of traffic. Through comparison with the original algorithm in simulation, we have demonstrated that this indirection technique reduces the number of reconfigurations necessary to serve all traffic demand by increasing the utilization of each link. Our evaluation shows that the indirection technique does not have prohibitively high computational overhead, and decreases the total amount of time necessary to fulfill all traffic demands.

References

[1] K. Chen, A. Singla, A. Singh, K. Ramachandran, L. Xu, Y. Zhang, X. Wen, and Y. Chen. OSA: An optical switching architecture for data center networks with unprecedented flexibility. In *USENIX NSDI*, 2012.

[2] N. Farrington, G. Porter, S. Radhakrishnan, H. H. Bazzaz, V. Subramanya, Y. Fainman, G. Papen, and A. Vahdat. Helios: a hybrid electrical/optical switch architecture for modular data centers. In *SIGCOMM 2010*.

[3] Gurobi. Gurobi Optimization. http://www.gurobi.com/.

[4] D. Halperin, S. Kandula, J. Padhye, P. Bahl, and D. Wetherall. Augmenting data center networks with multi-gigabit wireless links. In *SIGCOMM 2011*.

[5] S. Kandula, J. Padhye, and P. Bahl. Flyways to de-congest data center networks. In *HotNets 2009*.

[6] H. Liu, F. Lu, A. Forencich, R. Kapoor, M. Tewari, G. M. Voelker, G. Papen, A. C. Snoeren, and G. Porter. Circuit Switching Under the Radar with REACToR. In *USENIX NSDI*, 2014.

[7] H. Liu, M. K. Mukerjee, C. Li, N. Feltman, G. Papen, S. Savage, S. Seshan, G. M. Voelker, D. G. Andersen, M. Kaminsky, G. Porter, and A. C. Snoeren. Scheduling Techniques for Hybrid Circuit/Packet Networks. In *CoNEXT*, 2015.

[8] G. Porter, R. Strong, N. Farrington, A. Forencich, P. Chen-Sun, T. Rosing, Y. Fainman, G. Papen, and A. Vahdat. Integrating microsecond circuit switching into the data center. In *SIGCOMM 2013*.

[9] G. Wang, D. G. Andersen, M. Kaminsky, K. Papagiannaki, T. Ng, M. Kozuch, and M. Ryan. c-Through: Part-time optics in data centers. In *SIGCOMM 2010*.

[10] X. Zhou, Z. Zhang, Y. Zhu, Y. Li, S. Kumar, A. Vahdat, B. Y. Zhao, and H. Zheng. Mirror mirror on the ceiling: flexible wireless links for data centers. In *SIGCOMM 2012*.