Reduce Spectrum Defragmentation Latency in EONs with Effective Parallelization of Connection Reconfigurations

Changsheng You, Mingyang Zhang, Zuqing Zhu*

University of Science and Technology of China, Hefei, Anhui 230027, China, Email: zqzhu@ieee.org

Abstract: We investigate parallel defragmentation and propose a novel algorithm to achieve effective parallelization of the connection reconfigurations with a conflict graph. Simulation results show that the algorithm can effectively reduce the latency of traffic migrations.

OCIS codes: (060.1155) All-optical networks; (060.4251) Networks, assignment and routing algorithms.

1. Introduction

A new era of optical network can be foreseen with the advances of elastic optical networks (EONs) that adopt the optical orthogonal frequency-division multiplexing (O-OFDM) technology. EONs facilitate agile bandwidth management, but at the same time, also bring about new challenges. One important example is spectrum fragmentation, which refers to the existing of non-aligned, isolated and small-sized blocks of spectral segments in optical spectra due to fine spectrum allocation granularity and dynamic network operation. Since these segments can hardly be used for future connections, spectrum fragmentation causes low spectrum utilization and high blocking probability. In order to alleviate spectrum fragmentation, we need to develop cost-effective spectrum defragmentation (DF) approaches using network reconfiguration. Previous investigations have proposed DF algorithms [1, 2] and demonstrated physical-layer DF techniques [3, 4]. All these studies suggested that for practical implementation, DF operations should be finished as quickly as possible to minimize traffic disruptions. However, previous work only considered the sequential DF approach that tried to reconfigure the connections one at a time, which can introduce prolonged latency. Recently, the software-defined (SDN) architecture enabled by OpenFlow has been demonstrated for efficient optical network control [5]. The centralized network control provided by SDN opens up the possibility for parallel DF, which enables the operator to reconfigure multiple connections simultaneously and hence to reduce the latency significantly. In this paper, we investigate the parallel DF approach that uses the hop-tuning technique [3] in the physical-layer and propose a novel algorithm to achieve effective parallelization of the connection reconfigurations with an auxiliary conflict graph. The simulation results show that the proposed algorithm can effectively reduce the latency of traffic migrations.

2. Parallel Spectrum Defragmentation

In a DF operation, we need to first select the existing connections for reconfiguration. Since it is known that partial reconfiguration can achieve comparable blocking performance improvement as full configuration as long as the connections are carefully chosen [1], we select a fixed portion of existing connections to reconfigure in each DF operation. The portion is quantified as $\beta \in (0, 1]$, and we select connections according to the highest used slot-index first (HUSIF) strategy in [1]. Specifically, we sort the existing connections in descending order of the highest indices of their frequency slots (FS') and then select the top β portion of them. With the selected connections, we re-optimize their spectrum allocation on the routing paths and conduct traffic migrations for them to accomplish the DF operation. For each selected connection, the traffic migration tunes its spectrum allocation to the re-optimized one. However, when there is dependency between two or more selected connections, parallel DF will be impossible. For instance, Fig. 1(a) shows the spectrum utilization on a routing path in an EON, on which we select connections r_1 and r_2 and plan to reconfigure them to new spectrum locations r_1' and r_2'. Apparently, r_2' requires the FS' that are currently used by r_1 and hence making the traffic migrations $r_1 \rightarrow r_1'$ and $r_2 \rightarrow r_2'$ simultaneously as in Fig. 1(d) is not possible. We can only perform sequential DF as shown in Fig. 1(b). While as showed in Fig. 1(c), parallel DF is feasible as there is no dependency among r_1, r_2, r_1' and r_2'. If we assume that the latency of each traffic migration is unique as one time-unit, the latencies of the parallel and sequential DF operations in Figs. 1(d) and 1(b) are one and two time-units, respectively. Note that in addition to the saving on latency, parallel DF also brings down the complexity of implementing DF in the network control system. For instance, in the software-defined EON shown in Fig. 1(e), the OpenFlow controller needs to maintain a state-machine to facilitate the sequential DF since it is a stateful operation, while for the parallel DF, the controller can instruct all related OpenFlow agents to reconfigure in one step as the operation is stateless.

In order to realize parallel DF, we introduce two restrictions to the spectrum location reconfiguration phase for avoiding the dependency in Fig. 1(a): 1) a selected connection can only be tuned to unoccupied FS', and 2) the new spectrum location of a selected connection should possess lower FS indices than the original one. With these two restrictions, we can conduct parallel DF to consolidate the spectrum utilization in EONs. Fig. 2(a) illustrates an

978-1-55752-993-0/14/$31.00 ©2014 Optical Society of America

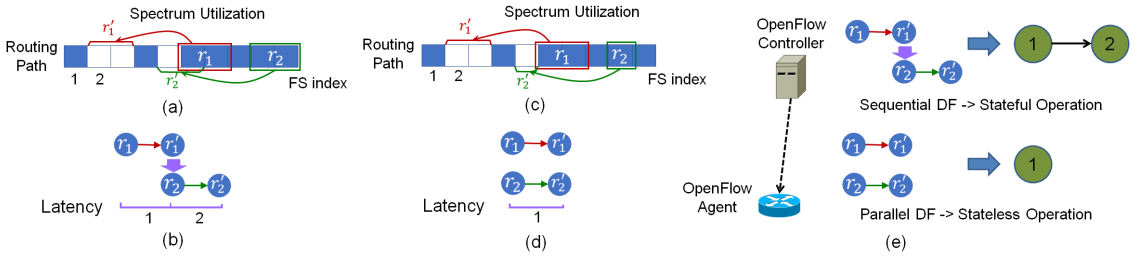

Fig. 1. Sequential and parallel DF operations in EON.

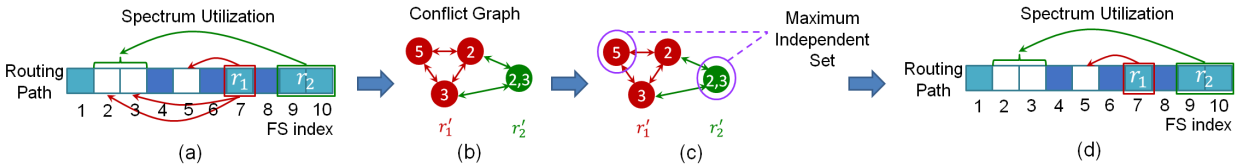

Fig. 2. An example of the proposed parallel DF algorithm.

example of determining the new spectrum locations under the two restrictions. In this case, r_1 can be tuned to three spectrum locations while r_2 can only be tuned to one. Then, we employ a conflict graph (CG), as shown in Fig. 2(b), to describe the conflicts among all the feasible new spectrum allocations of the selected connections. For each selected connection, we insert nodes into the CG to represent the feasible new spectrum allocations on its current routing path. Note that here we only consider the available spectrum locations on the same routing path to reduce the CG's size. All the nodes for the same connection are connected with links, since the connection can only be moved to one new spectrum location. As shown in Fig. 2(b), all the nodes for r_1' are connected. Then, if there is a conflict between two feasible new spectrum allocations of two different connections, the two corresponding nodes in the CG are connected with a link. For example, if r_1 is tuned to FS 2 or FS 3, r_2 would be stuck and hence the corresponding nodes should be connected in the CG in Fig. 2(b). With the CG, we propose a parallel DF algorithm that can reconfigure the maximal number of selected connections in parallel, under the aforementioned restrictions.

Step 1: Select β portion of existing connections with the HUSIF strategy in [1].

Step 2: Build CG to show the conflicts among all the feasible new spectrum allocations of the selected connections.

Step 3: Find the maximal independent set in the CG and the nodes in it represent the new spectrum locations where the parallel DF can move the selected connections to (as in Fig. 2(c)). Basically, the problem of maximizing the number of selected connections that the parallel DF can reconfigure is equivalent to finding the maximum independent set in the CG, as an independent set of a graph is a subset of nodes in which any two nodes are not connected. However, it is known that finding a maximum independent set in a random graph is NP-hard. Hence, we implement the maximal independent set search algorithm in [6] to reduce the computational complexity.

Step 4: Perform parallel traffic migrations based on the maximal independent set (as in Fig. 2(d)).

Step 5: Since compared with sequential DF, parallel DF can migrate less connections due to the restrictions, we conduct parallel DF multiple times in one operation to achieve better consolidation of the spectrum utilization. We define the number of parallel DF times in one operation as γ and repeat **Steps 1** to **5** γ times in one DF operation.

3. Performance Evaluation

We evaluate the performance of the proposed parallel DF algorithm in simulations using the 14-node NSFNET topology. We assume that the EON is deployed in the C-Band and each fiber link can accommodate 358 subcarrier slots. The dynamic requests are generated according to the Poisson traffic model with random source and destination for each one. The bandwidth requirement of each request is uniformly distributed within $[1, 16]$ FS'. The requests are originally served with the K-shortest paths and balanced load spectrum assignment (KSP-BLSA) algorithm [7], and all the DF algorithms trigger a DF operation when the number of expired requests exceeds 80. For performance comparisons, we also implement the scheme without any DF operation and the sequential DF as the benchmark algorithms. We evaluate the algorithms with three performance matrices, 1) bandwidth blocking probability (BBP), which is defined as the ratio of blocked to total requested bandwidth, 2) DF latency per operation, which measures the average time consumed for performing one DF operation, and 3) average number of connection reconfigurations per operation, which counts the average number of traffic migrations in a DF operation. Basically, BBP reduction indicates the benefit of DF, DF

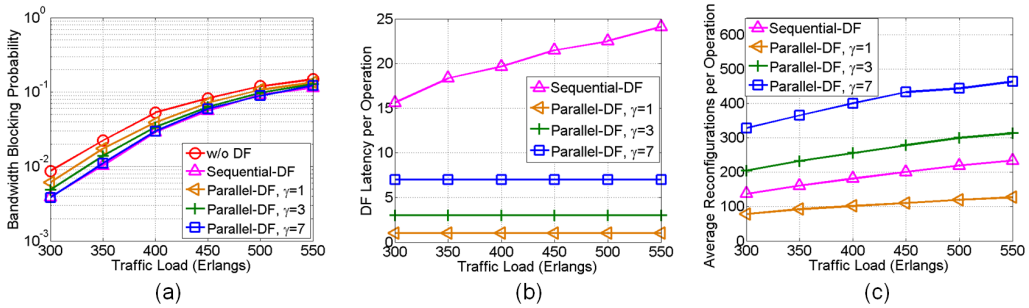

Fig. 3. Simulation results for $\beta = 0.5$, (a) Bandwidth blocking probability, (b) DF latency per operation, and (c) Average number of connection reconfigurations per operation.

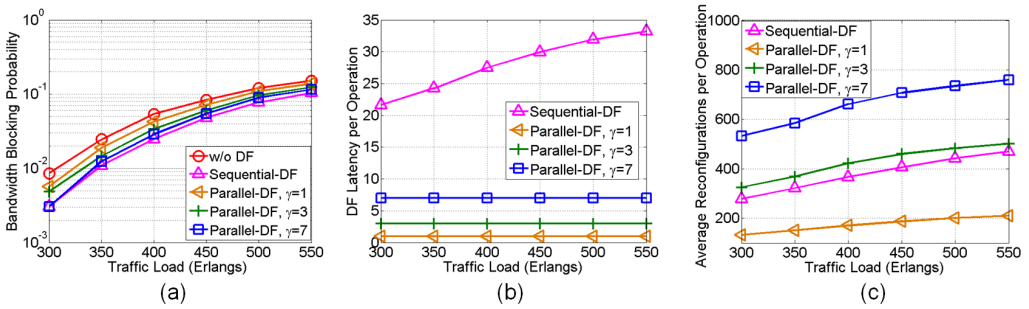

Fig. 4. Simulation results for $\beta = 1$, (a) Bandwidth blocking probability, (b) DF latency per operation, and (c) Average number of connection reconfigurations per operation.

latency is the performance characteristic of the DF algorithm, and average number of reconfigurations is the cost of the DF algorithm.

We first fix $\beta = 0.5$, *i.e.*, each DF operation tries to reconfigure 50% of the existing connections, and perform a series of simulations. For the parallel DF algorithms, we try $\gamma = 1,\ 3$, or 7. Fig. 3(a) shows the simulation results on BBP, and we observe that all DF algorithms achieves BBP reduction when compared with the one without DF. Since the sequential DF has no restrictions on traffic migrations, it achieves the largest BBP reduction. When γ is 1 or 3, the BBP performance of the parallel DF is worse than that of the sequential one, but when $\gamma = 7$, the BBP performance of the parallel DF is comparable to that of the sequential one. The results on DF latency is in Fig. 3(b), and it can be seen that the parallel DF algorithms provide much less latency than the sequential one. In the parallel DF, all the traffic migrations can be conducted in one step that consumes one time-unit and hence the DF latency per operation simply equals to γ and it does not change with the traffic load. But for the sequential DF, it needs to migrate traffic sequentially according to the dependency, which leads to more latency and causes the latency to increase with the traffic load since when the traffic load is higher, the dependency is more complicated. Fig. 3(c) illustrates the results on the average number of reconfigurations per operation, and we observe that when $\gamma > 1$, the numbers from the parallel DF are larger than those from the sequential one. This phenomenon can be understood as that in the parallel DF, a selected connection may be migrated to the "optimal" spectrum location in multiple steps due to the restrictions and hence the average numbers of reconfigurations are larger. We then fix $\beta = 1$ to perform a series of simulations for full reconfiguration, and Fig. 4 shows the results. It can be seen that the overall trends are similar.

4. Conclusion

We investigated parallel DF approach and proposed a novel algorithm to achieve effective parallelization of the connection reconfigurations with a conflict graph. Simulation results showed that the proposed algorithm could effectively reduce the latency of traffic migrations.

References

[1] M. Zhang, *et al.*, in *Proc. of ICC 2013*, pp. 1-5, Jun. 2013.
[2] R. Wang, *et al.*, *J. Lightw. Technol.*, vol. 31, pp. 2491-2500, Aug. 2013.
[3] R. Proietti, *et al.*, *Opt. Express*, vol. 20, pp. 26958-26968, Nov. 2012.
[4] F. Cugini, *et al.*, *J. Lightw. Technol.*, vol. 31, pp. 125-133, Jan. 2013.
[5] V. Gudla, *et al.*, in *Proc. of OFC 2010*, paper OTuG2, Mar. 2011.
[6] S. Balaji, *et al.*, *Adv. Modeling Optimization*, vol. 12, pp. 107-118, 2010.
[7] S. Y. Wang, *et al.*, *Proc. of INFOCOM 2013*, pp. 1503-1511, Apr. 2011.