On the Benefits of Multipath Routing for Distributed Data-intensive Applications with High Bandwidth Requirements and Multidomain Reach

Xiaomin Chen†, Mohit Chamania†, Admela Jukan†, André C. Drummond*, Nelson L. S. da Fonseca*
Technische Universität Carolo-Wilhelmina zu Braunschweig†
Institute of Computing, State University of Campinas*
{chen, chamania, jukan}@ida.ing.tu-bs.de, {andred, nfonseca}@ic.unicamp.br

Abstract

We investigate and quantify the benefits of multipath routing in a wide-area distributed environment which includes inter-domain routing issues. In this context, we discuss two possible multipath routing schemes and focus on the viable solution for distributed data-intensive applications with high bandwidth and delay requirements. The network topology aggregation is extended for end-to-end multipath computation. An ILP-based algorithm and a heuristic algorithm are proposed with multiple constraints, including bandwidth, delay and memory size. Numerical and simulation results show that the proposed multipath routing algorithms are feasible, and especially well-suited for emerging applications with extremely high bandwidth requirements.

1. Introduction

Multipath routing is a transmission technique which allows more aggregate bandwidth accommodated within networks by sending data into multiple parallel paths between the source and destination. Multipath routing has been primarily used to reduce blocking probability [8] as well as to improve the network resource utilization by an appropriate splitting scheme [13]. As the distributed *tera*- and *peta*-scale applications are emerging, which are pushing the bandwidth demands to the limits despite the enormous bandwidth in optical network, the multipath routing remains an attractive proposition [5]. Although services over the current Internet are getting cheaper and more bandwidth abundant, they still lack of mechanisms to support bandwidth allocation at specific data bandwidth granularities and advance reservation before job execution. This makes the existing connection-oriented networks, such as carrier-grade Ethernet [9], viable candidates for the advance computing applications with extreme bandwidth demands.

Data-intensive applications can benefit from better network resource availability, since multipath routing usually results in a lower blocking probability. However, multipath routing bears some challenges. First, multipath routing carries differential delay problem [2] which can present a potentially high requirement for memory size at the sink node. Differential delay is caused by multiple paths with different delays which leads to the situation where flows arriving earlier at the destination have to be buffered until all the remaining flows arrive. Second, geographically distributed computing resources may require that data transmission is provisioned over multiple administrative domains. This is illustrated by a simple two-domain network shown in Fig. 1. Between source $s(r)$ and destination $d(r)$, there are four paths P_1, P_2, P_3, and P_4, each with a different available capacity B and an end-to-end delay d. For instance, if 17 bandwidth units are requested between $s(r)$ and $d(r)$, it is obvious that the demand can only be fulfilled by splitting traffic into multiple paths. The path with the highest delay here is P_1, therefore, the memory size required at the destination would be given by $(d_1 - d_2) \cdot B_2 + (d_1 - d_3) \cdot B_3 + (d_1 - d_4) \cdot B_4 = 26$.

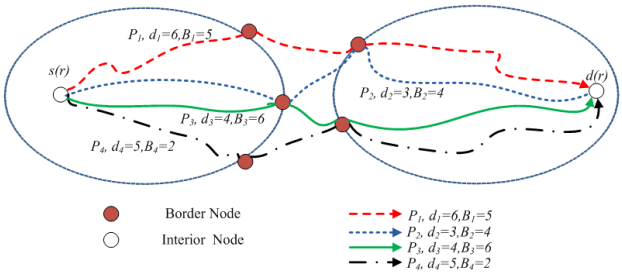

Figure 1. Multipath Routing with Multi-domain Reach.

In this paper, we investigate the benefits of multipath routing with specific emphasis on multi-domain scenarios. Two possible inter-domain multipath routing schemes

978-0-7695-3649-1/09 $25.00 © 2009 IEEE
DOI 10.1109/CNSR.2009.26

are discussed, i.e., *Segmental* and *End-to-end Multipath Routing*, where a *Path Computation Element* (PCE) [10] is deployed for path computation between domains. In the *Segmental Multipath Routing* scheme, traffic splits and merges inside the same domain regardless of the number of domains traversed. In the *End-to-end Multipath Routing* scheme, traffic splitting and merging are performed only once. Multiple paths are computed between source and destination for all incoming requests and the optimum path (which can be also the resulting single path) is chosen. Although *Segmental Multipath Routing* requires less inter-domain cooperation and is easy to be implemented, it cannot guarantee end-to-end delay and optimal paths. Our paper has the goal to investigate the benefits of multipath routing, in particular in the context of data-intensive applications where bandwidth- and delay-guaranteed service is required. Therefore, we focus on the *End-to-end Multipath Routing schemes* and propose a new inter-domain path computation scheme with extending the network topology aggregation for multipath routing. To this end, we develop algorithms for traffic splitting and path selection and use an *Integer Linear Programming* (ILP) approach to derive optimal solutions. Our formulation is comprehensive as it uses bandwidth, delay and differential delay as constraints. We also propose an accompanying heuristic algorithm as a more practical solution and show the benefits of our methods via numerical examples.

The remainder of the paper is organized as follows. In Section 2, we give an overview of related work. Section 3 describes the PCE-based inter-domain multipath computation mechanisms. Section 4 presents the ILP-based algorithm as well as the proposed heuristic algorithm. Section 5 is the performance evaluation for the proposed algorithms and Section 6 draws the conclusions.

2. Related Work

Multipath routing has been extensively studied in single domain networks, while little has been reported within the multi-domain scope. Cidon et al. [4] analyzed the performance of multipath routing theoretically and found significant advantages of multi-path routing over single path routing by comparing the throughput and the time required to establish connections. Banner et al. [3] formulated the multipath routing problem for minimization of the network congestion. The work in [3] focused on constraint-based path selection, without the consideration of traffic distribution. Ahuja et al. [2] studied the problem of minimizing the differential delay in the context of Ethernet over SONET. The algorithms proposed in [2] select a path for a Virtually Concatenated Group (VCG) which had the minimum differential delay. A few works have addressed inter-domain multipath computation problem with focus on survivable routing, whereby a single alternate path between source and destination over multiple domains is searched. Sprintson et al. [12] studied a path computation element (PCE) based scheme to find a pair of inter-domain disjoint MPLS paths. An inter-domain routing algorithm was proposed to find two disjoint paths based on the aggregated multi-domain topology.

3. Interdomain Multipath Computation Schemes

Previous multipath computation schemes in a single domain are based on the full knowledge of the network states. In a multi-domain scenario, this is not a realistic assumption which makes path computation a challenging task. Due to the constraints related to scalability, security and administrative policies, the intra-domain information cannot be fully advertised to the other domains, which naturally leads to the limited view of the entire network. We postulate that a collaborative path computation is required between domains.

3.1. Segmental Multipath Computation

When multiple domains are considered, it is possible that some transit domains are either heavily loaded or a single border node cannot support the incoming request. In that case, instead of *crankback* [11] mechanism to find an alternative domain, simple splitting of traffic in multiple paths can be beneficial in transit domains where a single path is unavailable.

In the *Segmental Multipath Computation* scheme, upon receiving connection request, domain PCE carries out path computation based on its domain *Traffic Engineering Database* (TED) in which domain information is stored. As exemplified in Fig. 2, PCE_{k-2} succeeds to compute a single path for the request between border nodes e_1 and e_3. The index k denotes the order of domains along the domain chain, where $k = 1$ is the source domain. Path information is sent back to e_1 and the request is forwarded to the next domain D_{k-1} which cannot serve the traffic by a single path. In conventional *crankback* routing, the connection request will be rejected to go through D_{k-1}. Instead, PCE_{k-2} has to find an alternative domain. In *Segmental Multipath Computation* scheme, PCE_{k-1} calculates multiple paths by running *k-shortest-path algorithm* [7] instead of sending *crankback* message to the previous domain. As shown in Fig. 2, $\{P_1, P_2, .., P_n\}$ is calculated between border nodes e_5 and e_7. Then path selection algorithm is run to adapt the required bandwidth as well as the network service requirements. If multipath computation and selection are successful, the request is forwarded to the next domain D_k.

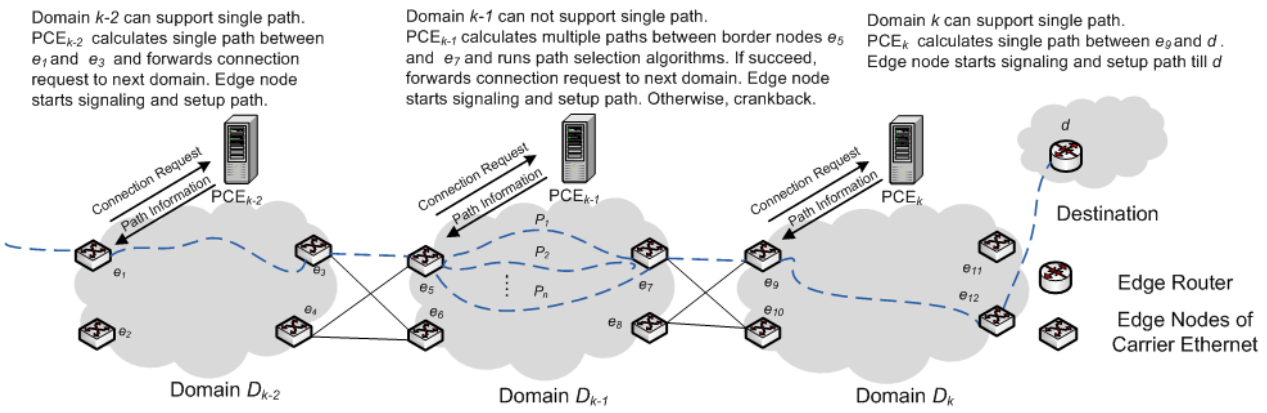

Figure 2. Segmental Multipath Computation.

In such a scheme, each domain makes its own decision on the routing algorithms for the incoming requests regardless of other domains, which alleviates inter-domain cooperation. However, drawbacks exist when *quality of service* (QoS) guaranteed services are required. First the *crankback* routing may lead to high signaling load and larege service-provisioning time. As shown in Fig. 2, when domain D_{k-1} cannot find paths from e_5 to any egress border nodes (e_7 and e_8), a *crankback* message is sent back to the previous domain D_{k-2} to find a new path to D_{k-1} where path computation is carried out again. Similar procedure is carried out until the request reaches the destination. Second, the *Segmental Multipath Computation* scheme cannot guarantee end-to-end delay since path setup is decided in a per-domain fashion. For this reason, our focus will be drawn on the viable multipath routing scheme for data-intensive applications requiring delay- and bandwidth-guaranteed services.

3.2. End-to-End Multipath Computation

Most of the existing path computation schemes are designed to support a single path setup at a time. PCE based inter-domain path computation algorithms such as the BRPC [6] are also restricted to the computation of a limited number of paths in the network. In the scheme proposed in [6], a sequential signaling is initiated from the source domain PCE to the destination domain PCE, and the path computation is carried out in the response cycle, where each intermediate domain uses the data from the previous domains to calculate a *Virtual Shortest Path Tree* from the destination and send it to the next previous domain towards the source. This mechanism is likely to reject a lot of possible paths which may not be optimal from the source to the destination and has to constructs virtual tree for each connection request.

In the *End-to-End Multipath Computation* scheme, we propose to insert representation of all domains to construct a virtual topology which can be used for end-to-end path computation. Topology aggregation mechanisms [14] are used for information dissemination. It advertises limited information about the domain topology, which is assimilated by other domains and used to construct an abstract inter-domain topology. Path computation schemes are then applied using the knowledge of these abstract schemes to determine actual paths in the network. An example is shown in Fig. 3. Domains $\{D_1, D_2, .., D_k\}$ constitute a domain chain between source and destination, which is known in advance. E_{IN_k} is the set of ingress border nodes of domain D_k while E_{OUT_k} is the set of egress border nodes. PCE$_k$ represents paths advertised for inter-domain traffic between E_{IN_k} and E_{OUT_k} in an aggregate topology. Upon receiving connection request, PCE$_2$ sends the aggregate topology of domain D_2 to PCE$_1$ and forward connection request to the next domain D$_3$. The similar procedure is processed until the connection request arrives at D_k and the PCE$_1$ receives aggregate topology of the whole domain chain. PCE$_1$ in source domain runs multipath routing algorithms for incoming request with QoS requirements. If the connection can be served by an end-to-end single path, the inter-domain multipath computation problems resort to inter-domain single path computation problems.

However, as current aggregate topology representations are modeled for single paths, they do not indicate if any physical resources are shared by different advertised multiple paths. Therefore, a new aggregate virtual topology representation is needed for multipath routing which can indicate the sharing of any common resources. This is illustrated in Fig. 4(a) where four paths are advertised with their available capacity and delay between border nodes. Conventional aggregate topology is represented in Fig. 4(b); this topology representation is commonly used in single path routing. However, path P_2 and P_3 have shared segment $D-J-G$ with available capacity 7. For multipath routing,

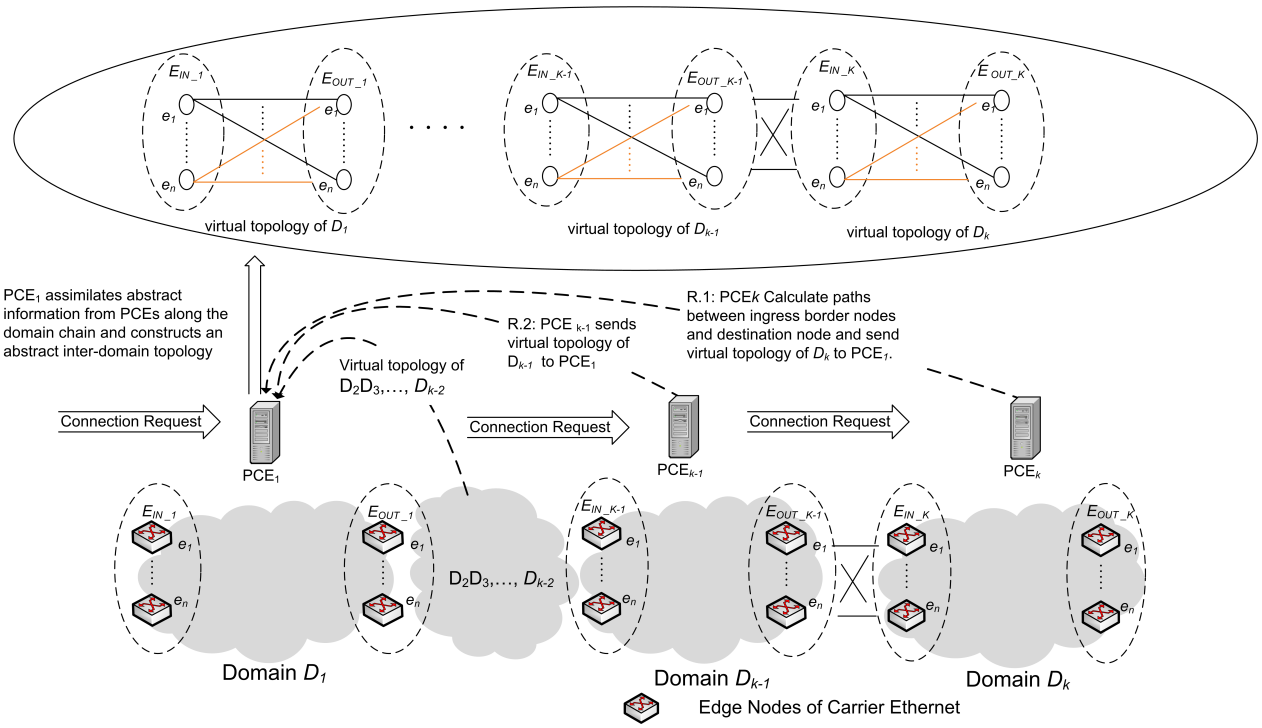

Figure 3. End-to-end Multipath Computation.

this represents a conflict. When P_2 and P_3 are to be used simultaneously, the total capacity advertised as 10 violates the actual available capacity of shared segments. We therefore propose an extension to the virtual topology representation, by representing shared segments in the virtual topology with their available capacity and delay accordingly, as shown in Fig. 4(c).

4. Traffic splitting and Path selection algorithms

4.1. Definition and Notations

Given is an aggregate topology of a multi-domain network which is represented by a directed graph $G(V, E)$, where V is the set of nodes and E is the set of links. The network consists of a set of domains D, denoted by $D^1, D^2, .., D^n$. Each domain $D^i(V^i, E^i)$ is a sub-graph of G and $V^i \subset V, E^i \subset E$. Given a connection request r, $s(r), d(r)$ are source and destination, and $b(r), \epsilon(r)$ are the requested bandwidth and end-to-end delay constraint. Define $M(d)$ as the memory size constraint at the sink node and M_r is the resulting memory size required by the connection request r after multipath routing. The path set for multipath routing represented by $\mathcal{P} = \{P_1, P_2, .., P_N\}$ is solution path set precomputed by the *End-to-End Multipath Computation* scheme proposed in Section 3.

The delay of one path $P \in \mathcal{P}$ is defined as:

$$d_P = \sum_{e \in P} d_e$$

Where d_e is the delay of link e in Path P.

The differential delay between two paths P and P' can be defined as [2]:

$$dd(P, P') = |d_P - d_{P'}|$$

Assume the path with highest delay in the solution path set $\mathcal{P}$ is $\tilde{P}$ and traffic distributed into path P is denoted by t_P, then the memory size M_r required by r is:

$$M_r = \sum_{P \in \mathcal{P}} t_P(d_{\tilde{P}} - d_P)$$

Each link $e \in E$ is associated with three parameters: c_e, b_e and w_e, where c_e, b_e are the total link capacity and available link capacity of link e respectively. w_e is the weight of link e, reflecting the delay of underlying paths in the physical topology.

4.2. ILP-based algorithm

The ILP-based algorithm aims to find solutions which can select paths and split traffic optimally with the objective of minimizing bandwidth usage. The ILP relies on the following variables:

- $\mathbf{x_e}$ - an integer variable for each link $e \in E$ denoting the current flow on link e.
- $\mathbf{t_P}$ - an integer variable for each path $P \in \mathcal{P}$ denoting the current traffic on the path P.

The ILP is formulated as follows:

$$\text{Minimize} \sum_{e \in E} w_e \cdot \mathbf{x_e}$$

Subject to:

$$\forall e \in E: \mathbf{x_e} = \sum_{P \in \mathcal{P} \wedge e \in P} \mathbf{t_P} \quad (1)$$

$$\forall e \in E: \mathbf{x_e} \leq b_e \quad (2)$$

$$\sum_{P \in \mathcal{P}} \mathbf{t_P} |d_{\tilde{P}} - d_P| \leq M(d) \quad (3)$$

$$\forall P \in \mathcal{P}: \mathbf{t_P} = 0 \, if \, d_P > d_{\tilde{P}} \quad (4)$$

$$\forall P \in \mathcal{P}: \mathbf{t_P} \leq b_P \text{ with } b_P = \min\{b_e, e \in P\} \quad (5)$$

$$\sum_{P \in \mathcal{P}} \mathbf{t_P} = b(r) \quad (6)$$

$$\mathbf{t_{\tilde{P}}} > 0 \quad (7)$$

Equation (1) relates the link flow to the traffic split into the computed paths. Since the link e may be shared by multiple paths, x_e is given by the addition of flows on all computed paths going through link e. Constraint (2) states the available link capacity constraint. Constraint (3) captures the memory size limitation at the destination node with the assumption that $\tilde{P}$ is the *highest delay* path in the current solution set $\mathcal{P}$. Constraint (4) shows that traffic is only split into the paths with lower delay than $\tilde{P}$. Constraint (5) gives the path capacity constraint of P which depends on the free capacity of the bottleneck link along the path. Constraint (6) states that the sum of traffic distributed into the selected path set should be equal to the required bandwidth of the inter-domain connection. Constraint(7) ensures that there is always traffic distributed into the path $\tilde{P}$ with the highest delay.

It can be seen that the requested memory size in Equation (3) depends on the choice of the highest delay path. To solve this ILP, we choose each P in $\mathcal{P}$ as a potential value of $\tilde{P}$ with respect to the maximal end-to-end delay $d_{\tilde{P}} \leq \epsilon(r)$. Each choice of $\tilde{P}$ implies that certain paths may not be taken into consideration as their delay is greater than the delay of $\tilde{P}$. We run the ILP with maximal $N = |\mathcal{P}|$ possible input scenarios, with every path from $\mathcal{P}$ that fulfills all requirements to the end-to-end delay set as $\tilde{P}$ once. As $\tilde{P}$ is the path with the highest delay, flows for paths with delay greater than $\tilde{P}$ are set to zero. Therefore the input to the ILP is a subset of paths belonging to $\mathcal{P}$, where each path has a delay less than or equal to $\tilde{P}$. The ILP is solved for each possible value of $\tilde{P}$, and the solution with the minimum value of the objective function from all the possible solutions is chosen as the optimal solution.

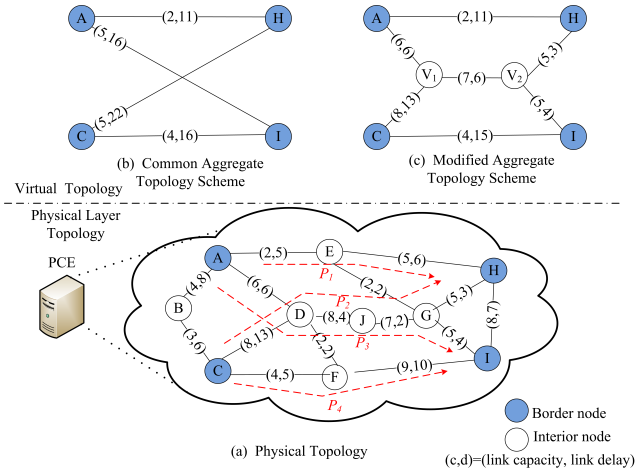

Figure 4. Aggregate Topology for Inter-domain Multipath Routing.

The time complexity of an ILP is known to be exponential. In the scenario that all paths have different delay, the ILP would be run N times, and the i-th iteration has an input path set of size i. Therefore the time complexity of the ILP is in the order of $O(2^1 + 2^2 + .. + 2^N)$ which is equal to $O(2^{N+1})$.

4.3. Heuristic Algorithm

The heuristic is developed around making an educated guess of the value of the path $\tilde{P}$, and then determining a corresponding solution set. In the heuristic, we do not find all possible solutions but stop at the first solution which can satisfy the path request. Upon an inter-domain connection request arrival, find a path set $\mathcal{P} = \{P_1, P_2, .., P_N\}$ from $s(r)$ to $d(r)$ within the end-to-end delay bound $\epsilon(r)$. Arrange $\mathcal{P}$ in increasing order of end-to-end path delay d_P. For two paths $\mathcal{P}_i$, $\mathcal{P}_j$ with $i > j$ it holds that the delay of the j-th path is always less or equal to the i-path. In order to make an educated guess about the choice of $\tilde{P}$, we create another set $\mathcal{P}'$, such that each element of $\mathcal{P}'$ has a one-to-one mapping with $\mathcal{P}$, and vice versa. The elements in set $\mathcal{P}'$ are arranged in a decreasing order of available path capacity. The one-to-one mapping between sets $\mathcal{P}$ and $\mathcal{P}'$ is utilized as follows: the path with the highest available capacity is chosen as the first guess for the highest delay path. The proposed heuristic algorithm is shown in Algorithm 1. It aims to distribute traffic into selected paths set proportionally. The outer loop starts from the path with biggest capacity in path set P and assumes it is also the highest de-

Algorithm 1 Heuristic for multi-path traffic splitting.

Input: $\mathcal{P}, \mathcal{P}', G(V,E), r$
Output: $\{t_P, P \in \mathcal{P}\}$
for $i = 1, .., N$ **do**
  $\tilde{P} = \mathcal{P}'_i$
  $bandwidth = 0$
  for $j = i, .., 1$ **do**
    $P' = \mathcal{P}_j$
    $bandwidth = bandwidth = b_{P'}$
    if $bandwidth \geq b(r)$ **then**
      for $k = i, .., j$ **do**
        $P = \mathcal{P}_k$
        $t_P = b_P \cdot \frac{b(r)}{bandwidth}$
      end for
      Convert calculated t_P to integers
      Distribute remaining bandwidth to minimize delay
      if $\sum_{P \in \mathcal{P}_{(i,..,j)}} t_P(d_{\tilde{P}} - d_P)) \leq M(d)$ **then**
        if *simultaneous reservation possible* **then**
          Output calculated flows and exit
        end if
      end if
    end if
  end for
end for

lay path $\tilde{P}$. The inner loop first calculates the sum of free capacity of paths and compares it with the required bandwidth. If it fails to find a group of paths with sufficient free capacity, the inner loop breaks and the outer loop picks the next $\mathcal{P}'_i$ as $\tilde{P}$. The algorithm continues until a group of paths is found with $bandwidth \geq b(r)$. The proportional traffic splitting ratio t_P is calculated and compared to the path capacity b_P (cf. Equation 4). The required memory size M_r is computed and compared with destination memory size limitation $M(d)$. If all paths can be reserved simultaneously and $M_r \leq M(d)$ is true, the algorithm stops and outputs the magnitude of the sub-flows. Otherwise, it goes to the outer loop and starts with another $\mathcal{P}'_i$. The heuristic algorithm can only stop when a solution is found or all paths in $\mathcal{P}$ are exhausted. The former will output the flows while the latter case will reject the connection request r.

The proposed heuristic has a polynomial time complexity in the order of $O(N^3)$ where N is the total number of paths. In case of larger networks where the time complexity is too large for practical purposes, the complexity can be reduced by making only a few educated predictions for a possible value of $\tilde{P}$. Various algorithms have been proposed to calculate multiple paths between a pair of nodes. The algorithm proposed in [7] can be used to perform constraint based k-shortest path computation with a time complexity of $O(kn \log n + e)$, where n is the number of nodes and e is the number of links.

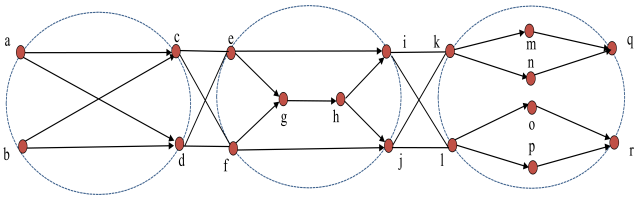

Figure 5. Aggregate Topology for Simulation.

5. Performance Evaluation

The multi-domain network used in the simulation consists of three concatenated domains. NSFnet topology is used as domain topology with four border nodes (Seattle, Houston, Ann Arbor, and Princeton). Paths are chosen between border nodes for transit traffic, and the choice of these paths leads to three different types of aggregate topology as shown in Fig. 5. All intra-domain virtual links are assumed to have a capacity of 40 Gbps while all inter-domain virtual links have a capacity of 80 Gbps. The capacity of shared segment in the intermediate domain is assumed to be 60 Gbps. For the purpose of this study, it is also assumed that the available capacity in the aggregated topology is not affected by intra-domain traffic inside any domain. We use an event driven simulator made in Java to test the performance of the proposed algorithms. To solve the ILP instance, the XPRESS solver program [1] is used.

Connection requests arrive between randomly selected source destination pairs with negative exponentially distributed inter-arrival times with a mean inter-arrival time of 1 second. Network load is varied by varying the mean holding time. The requested bandwidth by a connection is distributed uniformly between a pre-specified range, represented by (*Lower bound*, *Upper Bound*). We observe that the system reaches equilibrium at around 1000 seconds and then measure the performance of the system for another 8000 seconds.

We first study the variation of the blocking probability for different algorithms with the incoming requested bandwidth varying in the segment (*Lower Bound*, *Upper Bound*). Each node is assumed to have a constant available memory per connection request $M(d) = 100Mb$. To benchmark the performance of the ILP and the heuristic algorithms, we also simulate a single path routing algorithm under the same operating conditions. The single path routing algorithm also uses the constrained shortest path first algorithm in the aggregated topology to calculate a path between the source and the destination.

We compare the blocking probability of all proposed algorithms with dynamic requested capacity ranges under a

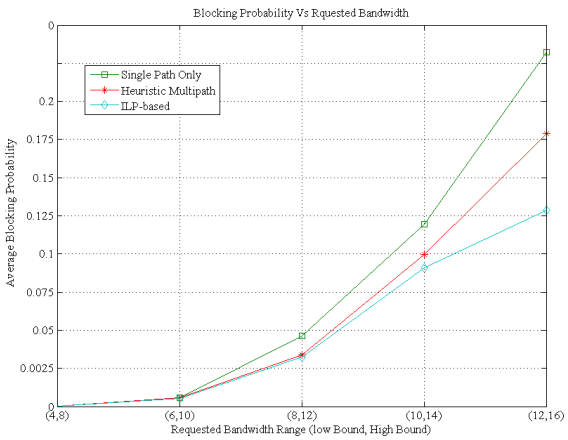

Figure 6. Blocking Probability vs Different Requested Bandwidth Range.

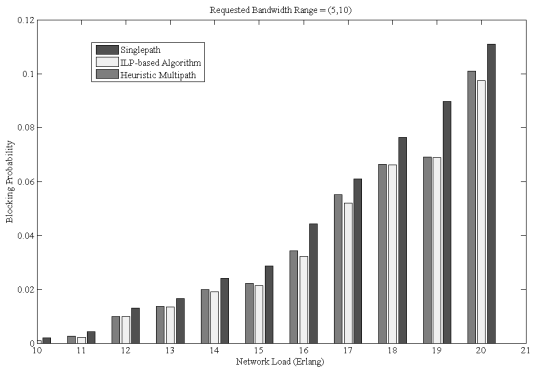

Figure 7. Blocking Probability vs Network Load.

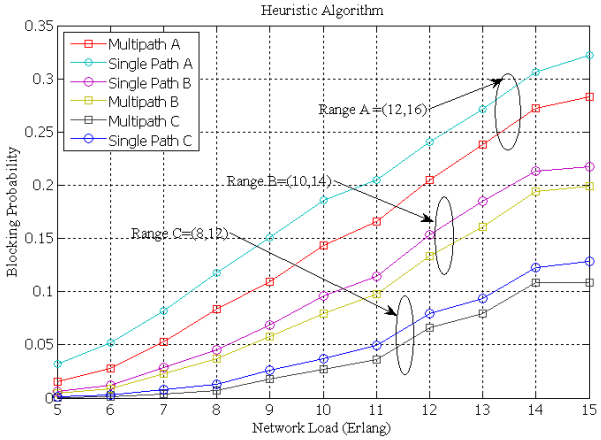

Figure 8. Blocking Probability vs Network Load.

constant network load (10 Erlangs). As seen in Fig. 6, the multi-path routing algorithms have lower blocking probabilities under a constant network load, compared to the single path routing algorithm. The heuristic solution has a marginally higher blocking as compared to the ILP-based algorithm within reasonable requested capacity (Maximum 14Gb/s in this case).

We then evaluate the blocking probability of all algorithms under network load varying between 10-20 Erlangs. As seen in Fig. 7, the multipath routing algorithms have significantly lower blocking probabilities as compared to the single path routing algorithm. The heuristic solution has a marginally higher blocking as compared to the ILPs. Significant decrease in blocking, especially at high network loads suggests that the multi-path routing algorithms might be beneficial conditions with high network load where the single path routing algorithm fails for find a possible route.

Given that the heuristic algorithm has comparable performance with the ILP-based algorithm with respect to blocking, we then comprehensively study the blocking probability of the proposed heuristic algorithm with different requested bandwidth ranges under different network loads (between 5-15 Erlangs). Link utilization is used as a reference for the performance evaluation of algorithms. It is defined as the ratio of used link capacity to the total link capacity. High link utilization may make the link more vulnerable to blocking. Therefore, it is reasonable to be considered as an important factor for any routing algorithm.

As shown in Fig. 8, blocking probability of multipath routing is always lower than single path routing with any bandwidth requirement in any network load. The average link utilization of single path routing and heuristic multipath routing algorithms with four incoming connections ranges, i.e., $(4, 6)$,$(6, 10)$,$(8, 12)$ and $(10, 14)$ is shown in Fig. 9. The link utilization of the heuristic is only slightly higher than that of the single path routing algorithm, suggesting that the heuristic algorithm can be applied to current networks without adverse effect on the link utilization in the network. The comparable link utilization and lower blocking together indicate the ability of the multi-path heuristic to distribute load among the different paths available to avoid heavily-loaded paths.

6. Conclusion

In this paper, we investigated benefits and challenges of multipath routing for distributed data-intensive applications. Two multipath routing schemes, i.e., *Segmental Multipath Routing* and *End-to-end Multipath Routing* were discussed. We suggested the End-to-end Multipath Routing

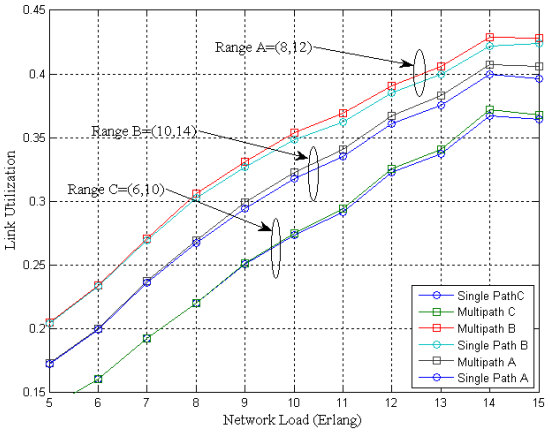

Figure 9. Average Link Utilization of different Algorithms.

scheme as a viable solution for the distributed data intensive applications and proposed an extended aggregate topology for multipath routing. We presented an ILP-based algorithm and a heuristic algorithm for traffic splitting and path selection. Memory size in sink node was considered as a constraint to minimize the differential delay problem caused by multipath routing. Benefits of multipath routing for distributed data-intensive applications with high bandwidth requirements have been shown by significantly lower blocking probability comparing with inter-domain single path routing, especially at high network load. Algorithms proposed in this paper can therefore be used as an ideal fallback mechanism to serve data-intensive applications with multiple domain reach, effectively increasing the performance of the network.

Acknowledgment

This work has been supported by Deutsche Forschungsgemeinschaft (DFG) under support code JU2757/-1/1; and FAPESP under support code 2007/54867-4.

References

[1] Xpress-MP Suite,2008. http://www.dashoptimization.com.

[2] S. Ahuja, T. Korkmaz, and M. Krunz. Minimizing the differential delay for virtually concatenated Ethernet over SONET systems. In *Computer Communications and Networks, 2004. ICCCN 2004. Proceedings. 13th International Conference on*, pages 205–210, 2004.

[3] R. Banner and A. Orda. Multipath routing algorithms for congestion minimization. *IEEE/ACM Transactions on Networking (TON)*, 15(2):413–424, 2007.

[4] I. Cidon, R. Rom, and Y. Shavitt. Analysis of multi-path routing. *Networking, IEEE/ACM Transactions on*, 7(6):885–896, 1999.

[5] W. Guo. Optimal Traffic Splitting Scheme in Multiple Path for Next-generation Optical Network. 2005.

[6] IETF. A Backward Recursive PCE-based Computation (BRPC) Procedure To Compute Shortest Constrained Inter-domain Traffic Engineering Label Switched Paths. April 2008.

[7] M. MacGregor and W. Grover. Optimized k-shortest-paths Algorithm for Facility Restoration. *Software - Practice and Experience*, 24(9):823–834, 1994.

[8] S. Rai, O. Deshpande, C. Ou, C. U. Martel, and B. Mukherjee. Reliable multipath provisioning for high-capacity backbone mesh networks. *IEEE/ACM Trans. Netw.*, 15(4):803–812, 2007.

[9] A. Reid, P. Willis, I. Hawkins, and C. Bilton. Carrier ethernet. *Communications Magazine, IEEE*, 46(9):96–103, 2008.

[10] RFC4655. A Path Computation Element (PCE)-Based Architecture.

[11] RFC4920. Crankback Signaling Extensions for MPLS and GMPLS RSVP-TE.

[12] A. Sprintson, M. Yannuzzi, A. Orda, and X. Masip-Bruin. Reliable Routing with QoS Guarantees for Multi-Domain IP/MPLS Networks. In *Proceedings of INFOCOM 2007*, pages 1820–1828, 2007.

[13] N. Taesombut, F. Uyeda, A. Chien, L. Smarr, T. DeFanti, P. Papadopoulos, J. Leigh, M. Ellisman, and J. Orcutt. The OptIPuter: High-Performance, QoS-Guaranteed Network Service for Emerging E-Science Applications. *IEEE Communications*, 44(5):38–45, 2006.

[14] S. Uludag, K.-S. Lui, K. Nahrstedt, and G. Brewster. Analysis of topology aggregation techniques for qos routing. *ACM Comput. Surv.*, 39(3), 2007.