Optimised Heuristics for a Geodiverse Routing Protocol

Yufei Cheng*, M. Todd Gardner‡, Junyan Li*, Rebecca May‡, Deep Medhi‡, and James P.G. Sterbenz*†
*Information and Telecommunication Technology Center
EECS, The University of Kansas, Lawrence, KS, 66045, USA
{yfcheng|balee|jpgs}@ittc.ku.edu
†SCC and InfoLab21, Lancaster LA1 4WA, UK
jpgs@comp.lancs.ac.uk
www.ittc.ku.edu/resilinets
‡University of Missouri-Kansas City, MO, USA
todd.gardner@faa.gov rm5x8@mail.umkc.edu DMedhi@umkc.edu

***Abstract*—We propose two heuristics for solving the path geodiverse problem (PGD), in which the calculation of a number of geographically separated paths is required. The geodiverse paths can be used to circumvent physical challenges such as large-scale disasters in telecommunication networks. The heuristics we propose for solving PGD have significantly less complexity compared to the optimal algorithm we previously used while still performing well by returning multiple geodiverse paths for each node pair. The geodiverse paths contribute to providing resilience against regional challenges. We present the GeoDivRP routing protocol with two new routing heuristics implemented, which provide the end nodes with multiple geographically diverse paths and demonstrates better performance compared to OSPF when the network is subject to area-based challenges.**

***Index Terms*—path geodiversitiy; physical topology; survivable routing heuristics; network resilience; diversity routing; multi-path routing;**

I. Introduction and Motivation

Telecommunication networks rely heavily on physical infrastructure such as optical fibers, amplifiers, routers, and switches to maintain normal operation, and their resilience to various faults and challenges is important to be analysed [1]. The geolocation of network components and their relative distance between each other affect the network survivability since a significant number of challenges affect a wide range of nodes and links. Most previous work considers only random link and non-correlated failures [2], [3]. In contrast, we are modelling correlated failures and attacks [4]. It has also been observed that a large number of failures in a geographical region can cause catastrophic damage to the network communications [5]. We study the geodiversity characteristics of the network graph to understand how regional challenges affect connectivity of the network and how to mitigate its impact. Many area-based challenges can be modelled as a circular area with a certain challenge radius. For example, an earthquake or hurricane that has a challenged radius of 0 to 500 miles impact zone can cause failed links and nodes with substantial impact on network communications [6].

We have proposed a two-step optimal algorithm for solving the path geodiverse problem (PGD). The algorithm begins with the Suurballe's algorithm [7], [8] in which a shortest-path algorithm (SPA) is iteratively applied. After each iteration of the SPA, the weight of the edges from the constructed path is adjusted by adding a penalty factor. Once the algorithm has identified k paths, it selects the path with distance d separation (in which any two nodes on disjoint paths are separated by greater than d distance) by iteratively comparing the distance between each and every node pair from all the candidate paths. Based on our algorithm, we have designed a geodiverse routing protocol (GeoDivRP) that uses the geographic location of nodes and routes traffic around the challenges by exploiting the diversity in the underlying physical topologies [5]. This mechanism reaches optimality in choosing the best d-separation paths assuming a large number of candidate paths. However, as SPA is applied k times for generating the candidate paths before selecting the qualified ones, its time complexity is large and the computation is slow. To reduce the complexity of the optimal algorithm, we propose two heuristics, iterative WayPoint Shortest Path heuristic (iWPSP) and Modified Link Weight heuristic (MLW). We approach these two heuristics from different perspectives: iWPSP selects one waypoint node and performs SPA twice between the waypoint node and source neighbor node, and then the waypoint node and destination neighbor node, respectively. On the other hand, our second heuristic MLW linearly or exponentially increases the links' weight within distance d from the straight line connecting source and destination node. SPA is then used to calculate the paths based on the modified link weights. We use Dijkstra's shortest path algorithm [9] for both heuristics.

Our GeoDivRP fits in the protocol stack as shown in Figure 1. *Knobs* $\mathbb{K}$ are used by higher layers to influence lower layer operation while *dials* $\mathbb{D}$ are the mechanisms for lower layers to provide feedback to higher layers [10]. The application layer passes a service specification and threat model down to our resilient transport layer protocol ResTP. Upon receiving these parameters, ResTP determines the type of transport service needed (including error control and multipath characteristics) and requests that GeoDivRP calculate geodiverse paths that meet the requirement tuple $(k, d, [h, t])$,

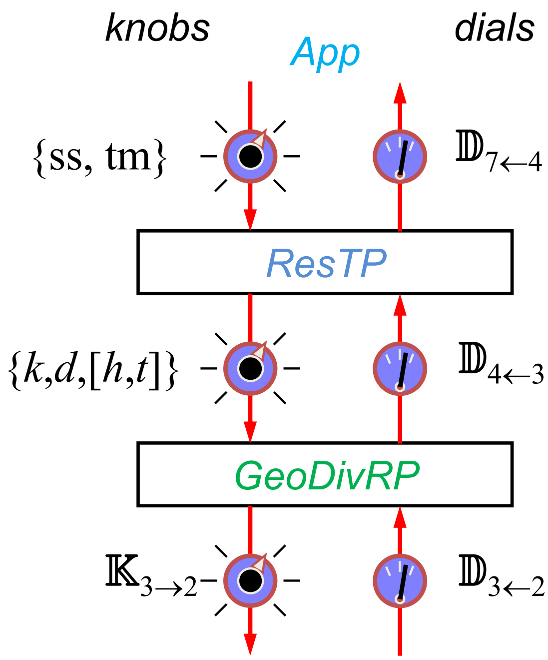

Fig. 1. Layered block diagram of GeDivRP and ResTP

where k is the total number of geodiverse paths requested, d is the distance separation criteria, $[h, t]$ are the desired constraints on path stretch h and temporal skew across paths t. ResTP then establishes a multiflow with error control needed for meet the service spec, including the per-subflow control (ARQ, hybrid ARQ, FEC, or none) and flow bundle (e.g. 2-of-3 erasure code for real-time critical service, or 1+1 redundany with a hot-standby for delay and loss tolerant service) taking advantage of k d-geodiverse paths $[P]$ provided by GeoDivRP. We apply the multipath algorithm in the context of several real-world service provider networks to analyse the diversity gain and packet delivery ratio when routing protocol is considered. We further extend this routing mechanism to our geo-diverse routing protocol GeoDivRP.

The remaining sections of the paper are organised as follows: Section II presents the background and related work. Section III introduces our two routing heuristics and the evaluation methodology. Section IV presents the simulation results and demonstrates the performance gain from our routing heuristics in real-world physical networks. Section V concludes the paper and suggests future work.

II. Background

The transportation network community has been studying geographically diverse path algorithms [11]–[13] to reduce danger to populated areas when transporting hazardous materials. However, similar to our optimal algorithm [5], their work requires the calculation of k shortest paths. One exception is the gateway shortest path algorithm [12]. It selects one gateway node for a path gateway when calculating shortest path, but has a couple of drawbacks. For example, it can easily form loops, and when calculating more than two geodiverse paths, the later calculated paths are less separated from the previous established path. This is because the gateway node is always chosen based on the distance to the absolute shortest path. Another work has presented a performance comparison of the existing geodiverse algorithms in transportation networks [14].

The telecommunication network community has long been studying edge/vertex-disjoint paths and many previous works have presented survivable network routing design using disjoint paths [7], [8], [15]–[17]. However, those works are generally concerned with ensuring that components and links do not share the same location, rather than considering the d distance between them needed for a specified separation for path geodiversity (PGD). As area-based challenges are important to analyse, an efficient algorithm is required to solve the PGD problem. We have proposed one path diversity mechanism for qualifying the reliability of network flows to characterise the network resilience [18], [19]. We have extended this work to the analysis of geodiversity in physical network topologies and proposed cTGGD (compensated geographical graph diversity) to characterise the geodiversity of different network topologies [5]. We have also proposed a routing protocol that takes advantage of the geodiversity in the network topology and achieve resilience by providing multiple geodiverse paths to different application scenarios.

III. Model Description

In this section, we define *path geodiversity* and propose two heuristics for efficiently calculating geographically diverse paths. Specifically, we consider the path geodiverse problem (PGD), which involves obtaining a set of paths that are d distance separated from each with every vertex in different disjoint paths (d-separation). The proposed heuristics return a path tuple of (S, D) paths from the graph $G = (V, E, w)$, where V is the vertex set, E is the edge set, and w is the link weight set. Dijkstra(G, n) is the standard Dijkstra algorithm we use to provide the shortest path.

A. Path Geodiversity

We define the geodiversity as how far two paths are separated from each other in terms of geographic distance. We present some necessary definitions as follows.

Path is defined as a vector that contains all the links L and intermediate nodes N from source S to destination D

$$P = L \cup N \tag{1}$$

GeoPath diversity $D(P_a)$ is defined as the distance between any node member of the vector P_a and that of the shortest path. As shown in Figure 2, node 0 is the source and node 2 is the destination. The shortest path consists node 0-1-2. The green dotted line shows the path P_1 and the diversity $D(P_1)$ equals d, which is the shortest distance between any node pairs on the disjoint paths (except for the source and destination). The blue dotted line shows path P_2 and its geodiversity $D(P_2)$ is zero since P_2 shares node 1 with the shortest path.

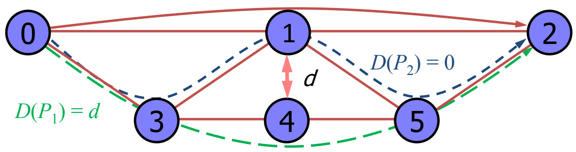

Fig. 2. Geographic diversity: distance d

Path stretch is defined as the hop counts of a given path P_A divided by the hop counts of the shortest path P_s

$$S = L_{P_A}/L_{P_s} \tag{2}$$

where we use the same definition from [16].

In addition to these metrics and definitions, we list the graph notations used in the paper as follows:

- $G(V, E, w)$: input graph $\|G\|$ with a set of vertices V, a set of edges E and weight of edges w
- S: source node
- D: destination node
- S_n: neighbor node chosen by source node
- D_n: neighbor node chosen by destination node
- k: number of geodiverse path requested
- d: distance separation between each and every node in different disjoint paths
- δ: delta distance when selecting waypoint node

B. Heuristics Introduction

In consideration of decreasing the complexity of geodiverse path calculation, we propose two heuristics: iterative WayPoint Shortest Path (iWPSP) and Modified Link Weight (MLW). As shown in Figure 3 for the case when $k = 3$, iWPSP first selects neighbor nodes S_{k1} and D_{k2} that are d distance separated from source node S and destination node D, respectively (for simplicity in this presentation we assume that such nodes exist; otherwise the nodes with the greatest distance will be chosen, iterating until nodes d apart are located). Assuming the straight link connecting S and D is L, iWPSP selects waypoint nodes m' and m'' in the opposite direction that are distance $d + \delta$ apart from the middle node m in the shortest path, where the segment $m'mm''$ interleaves with the shortest path. Dijkstra's algorithm is performed for the two branches $S_{km'}$ and $D_{km'}$. By connecting the shortest path returned from the two branches, the heuristic obtains the first geodiverse path P_1. The same mechanism repeats for waypoint node m'' for the second geodiverse path. The variable d is a user-chosen parameter based on a threat model for a challenge of distance d, and δ is experimentally chosen for different network topologies to increase the probability of the heuristic successfully returning a d-separation path. The δ parameter is also introduced to prevent the edges in each of the two paths from interleaving and creating routing loops. By tweaking the value of δ, this heuristic can select a nearby waypoint node if the previous one fails running Dijkstra's algorithm. The code of iWPSP is shown in Algorithm 1.

Functions:
Calculate k paths from S to D separated by distance d
Input:
G_i:= input graph
S:= source node
D:= destination node
k:= number of requested geodiverse path
d:= separation distance between the paths
δ:= delta distance when selecting waypoint node
Output:
k number of geographically d distance separated paths
begin
  segment L connecting S and D, with its middle point m;
  choose neighbor node S_k, D_k that is at least d distance from S_{k-1}, D_{k-1}, respectively;
  if *k is odd number* **then**
    choose two nodes m_1 and m_2 that are separated by $d + \delta$ on each direction of L, where m_1mm_2 is perpendicular bisector of L;
    $P_1 = \text{SourceTree}_{DS} \leftarrow \text{Dijkstra}(D, S)$;
    $k- = 3$;
  else
    choose two nodes m_1 and m_2 that are separated by $d/2 + \delta$ on each direction of L, where m_1mm_2 is perpendicular bisector of L;
    $k- = 2$;
  end
  $p_{m_1S_1} = \text{SourceTree}_{S_1m_1} \leftarrow \text{Dijkstra}(m_1, S_1)$;
  $p_{m_2S_2} = \text{SourceTree}_{S_2m_2} \leftarrow \text{Dijkstra}(m_2, S_2)$;
  $p_{m_1D_1} = \text{SourceTree}_{D_1m_1} \leftarrow \text{Dijkstra}(m_1, D_1)$;
  $p_{m_2D_2} = \text{SourceTree}_{D_2m_2} \leftarrow \text{Dijkstra}(m_2, D_2)$;
  while $k > 0$ **do**
    segment L = newest established path;
    choose one node m_k that is separated by distance $d + \delta$ from L on the farther direction from the absolute shortest path;
    $p_{m_kS_k} = \text{SourceTree}_{m_kS_k} \leftarrow \text{Dijkstra}(m_k, S_k)$;
    $p_{m_kD_k} = \text{SourceTree}_{m_kD_k} \leftarrow \text{Dijkstra}(m_k, D_k)$;
    $k- = 1$;
  end
  if *k is odd number* **then**
    $P_2 = p_{m_1S_1} + p_{m_1D_1}$;
    $P_3 = p_{m_2S_2} + p_{m_2D_2}$;
    ...
    $P_k = p_{m_{k-1}S_{k-1}} + p_{m_{k-1}D_{k-1}}$;
  else
    $P_1 = p_{m_1S_1} + p_{m_1D_1}$;
    $P_2 = p_{m_2S_2} + p_{m_2D_2}$;
    ...
    $P_k = p_{m_kS_k} + p_{m_kD_k}$;
  end
  return $(P_1, P_2, ..., P_k)$
end

Algorithm 1: Iterative waypoint shortest path heuristic

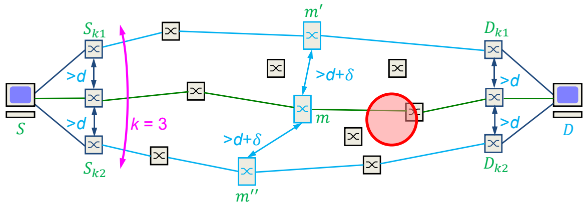

Fig. 3. Iterative waypoint shortest path heuristic

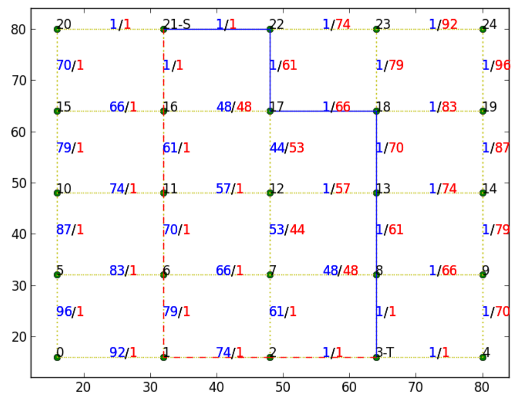

Fig. 4. Geodiverse paths by MLW heuristic in Grid network

Our second heuristic MLW statistically modifies the link weights and performs Dijkstra's algorithm to calculate the geodiverse path with the modified link weights in the network. The heuristic begins by increasing, linearly or squarely, the weight in one direction based on the perpendicular distance to the line L connecting source node S and destination node D. The weight incremental ratio is inversely proportional to the distance from L. Dijkstra's algorithm is applied on the graph with modified link weights. The heuristic repeats the process for the other perpendicular direction to L. This way the heuristic can generate two paths that are geographically separated. If more diverse paths are required, the heuristic selects one of the geodiverse paths established as the starting line for modifying link weights and iteratively generates k geodiverse paths.

We use a 5×5 grid network to demonstrate the d-separation paths calculated by MLW. As shown in Figure 4, MLW calculates two paths that are separated by distance d by statistically modifying link weights. Node 21 is the source and node 3 is the destination. The d value is set as twice length of edges in the grid. iWPSP heuristic generates same results when using the same setup and the mechanism is shown in Figure 3. The weight shown in different color is used for calculating paths in its representative colors. For example, when MLW is calculating the path shown in blue solid links, the link weight is statistically modified by decreasing towards the top right corner of the Grid network. The detailed heuristic is presented in Algorithm 2.

Functions:
cost(L):= cost function
Input:
G_i:= input graph
W_i:= link weights
S:= source node
D:= destination node
k:= number of diverse paths requested
buffer := distance buffer to increase link weight
Output:
k number of paths that are geographically separated by distance d
begin
  straight line l connecting source S and destination D
  if k *is odd number* **then**
    $P_1 = \text{SourceTree}_{DS} \leftarrow \text{Dijkstra}(D, S)$;
    modify link weight linearly or squarely on one direction perpendicular to line l until distance d;
    $P_2 = \text{SourceTree}_{DS} \leftarrow \text{Dijkstra}(D, S)$;
    repeat the process for the other direction;
    buffer = d;
    $k- = 3$;
  else
    modify link weight linearly or squarely on one direction perpendicular to line l until distance $d/2$;
    $P_1 = \text{SourceTree}_{DS} \leftarrow \text{Dijkstra}(D, S)$;
    repeat the process for the other direction;
    buffer = $d/2$;
    $k- = 2$;
  end
  while $k > 0$ **do**
    buffer += d;
    modify link weight linearly decreasing on one direction perpendicular to line l until buffer;
    links beyond distance buffer, link weight = 1;
    $P_{k-1} = \text{SourceTree}_{DS} \leftarrow \text{Dijkstra}(D, S)$;
    repeat the process in the other direction;
    $P_k = \text{SourceTree}_{DS} \leftarrow \text{Dijkstra}(D, S)$;
    $k- = 1$;
  end
  return $(P_1, P_2, ..., P_k)$
end

Algorithm 2: Modified link weight shortest path heuristic

We have implemented both of the heuristics in our GeoDivRP routing protocol first introduced in [5]. The two heuristics are implemented as different options and take a user-provided switch to control which heuristic the routing protocol uses. The implementation is done using ns-3 [20], a popular network simulator to analyse network protocols and challenges through simulation. We base this protocol on link state routing methodology. At the beginning of the simulation, by obtaining node locations from the link state update messages, we calculate the geodiverse paths and store them in the path cache server. When the simulation begins, our protocol sends data traffic using the paths from the cache. When a challenge occurs in the network, our protocol responds to the challenge faster than OSPF (Open Shortest Path First) [21] and calculates the paths according to the challenge estimation [5]. The distance value d is a user-provided value, and when challenges occur, users can modify d according to the different threat models to ensure the traffic circumvents the challenged area.

Both of the heuristics have incorporated improvement mechanisms. When the calculated paths obtained fail to qualify the d-separation criteria, iWPSP will choose another waypoint that has a slightly larger d distance, while MLW will increase the link weight around the avoidance line. Then the heuristics initialise another iteration of Dijkstra's algorithm. The heuristics fall back to the optimal algorithm if the result still does not qualify, which ensures that both of the heuristics have a better chance acquiring the geodiverse path while not generating their worst case complexity. Another major component of both heuristics is loop detection. For example, the iWPSP algorithm can create routing loops when calculating paths for corner nodes in the topology. We use a loop detection algorithm so that if a previous node from one path is identified, the algorithm deletes that part.

IV. Real Network Results

In this section, we evaluate the proposed heuristics and compare their performance with the two-step optimal algorithm [5]. We present the geodiverse paths calculated by our heuristics using the Nobel-EU (Pan-European Reference Network) with 28 nodes and 40 links [22]. We assume a challenge along the line from Amsterdam to Rome with a radius of 50 km. Node Strasbourg and Frankfurt are in the challenge circle. The result of iWPSP is shown in Figure 5. The challenge area is shown in the red circle. The result of MLW is shown in Figure 6 for its two paths. We only show the two paths from Amsterdam to Rome. The first path shown in red dashed link is Amsterdam-Hamburg-Berlin-Munich-Vienna-Zagreb-Rome, and the second path shown in blue solid link is Amsterdam-Brussels-Paris-Lyon-Rome. We shown a large radius challenge in Figure 7.

A. Complexity Analysis and Evaluation

We analyse the complexity of the two heuristics compared to the optimal algorithm. For simplicity, we examine the complexity for obtaining two d-separation paths and assume the Fibonacci heap for Dijkstra's algorithm. The optimal algorithm starts by calculating k edge-disjoint paths using Suurballe's algorithm, which requires k iterations of Dijkstra's algorithm. Dijkstra's algorithm can be performed in time $O(m + n \log n)$ on a graph with n vertices and m edges. Therefore, the same time complexity applies to each path for the Suurballe's algorithm, which makes its complexity $O(km + kn \log n)$. After generating k disjoint paths, the optimal algorithm demands choice of paths that qualify the distance separation criteria. This process requires n^2 time, which means the total complexity for the optimal algorithm is

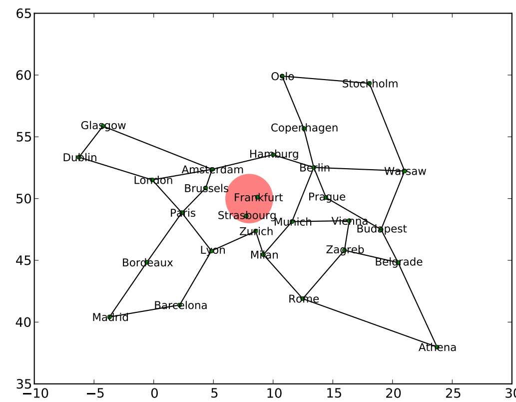

Fig. 5. iWPSP heuristic in Nobel-EU network

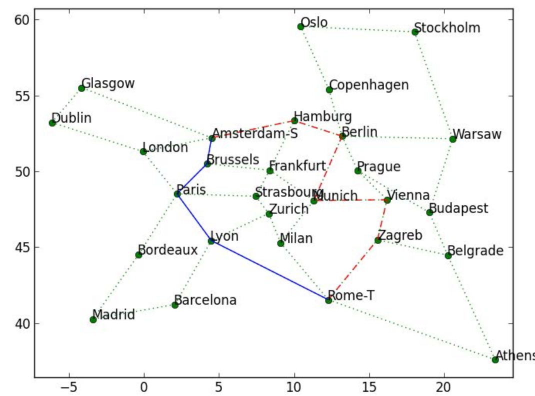

Fig. 6. MLW heuristic in Nobel-EU network

$m + n \log n + kn^2$, or $O(kn^2)$. The number of edge-disjoint paths k is usually large to establish the algorithm's optimality. For most application scenarios, k is chosen to be 1000 [12]. Therefore, for a network with vertices less than 1000, the complexity of the optimal algorithm goes up to $O(n^3)$.

iWPSP has a complexity of $2c^2n^2 \log n$, where c is the average number of neighbors for vertices, the complexity for choosing the waypoint node is $O(n)$, where n represents the number of nodes, and $2n \log n$ is for Dijkstra's algorithm to calculate the two shortest paths. Therefore, the worst case scenario is $O(n^2 \log n)$ while the best case scenario is $O(n \log n)$. Most of the physical topologies have an average degree between two and three [23]. This means that c in our complexity analysis is a small constant. This reduces the best case time complexity of iWPSP to $O(n \log n)$. The complexity of MLW is $O(2n \log n)$, which is the complexity for invoking Dijkstra's algorithm twice. The complexity for both of our heuristics is much better than that of the optimal algorithm, which is $O(n^3)$.

We present the execution time of the heuristics to demonstrate their effectiveness compared to optimal algorithm in the

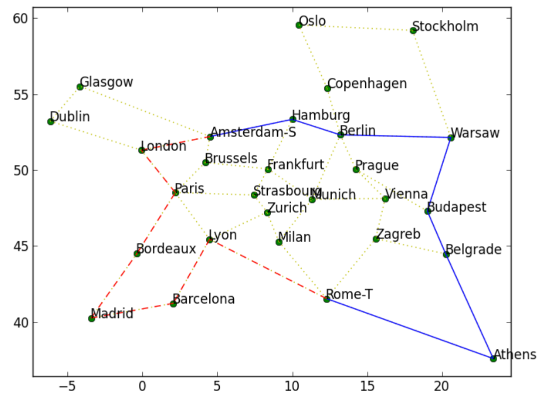

Fig. 7. MLW heuristic in Nobel-EU network with large radius

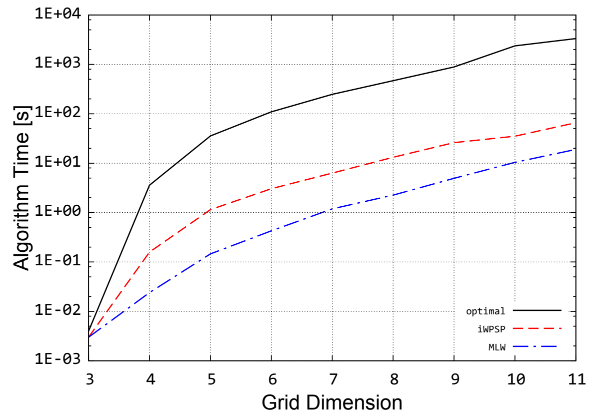

Fig. 8. Complexity analysis and comparison

case of calculating two d-separation paths. The evaluation is on a Linux machine with 3.16GHz Core 2 Duo CPU with 4GB memory. We use different dimensions of grid networks to analyse the time complexity. The grid dimension ranges from 3×3 to 11×11, which means the number of nodes varies from 9 to 121. We show the time to calculate all the node pairs in the topology. When calculating only one path pair that happens more often in real-world scenarios, the time is exponentially less. As shown in Figure 8, the x-axis is grid dimension and the y-axis is the log-level algorithm execution time in seconds. Both MLW and iWPSP algorithms show better execution time compared to the optimal algorithm. For calculating all the paths in 11×11 grid, MLW takes 20 s, iWPSP takes 65 s, while the optimal algorithm takes greater than 3000 s. We can observe that iWPSP has greater execution time compared to that of MLW. This is because of the extra time of Dijkstra's algorithm and selecting qualifying waypoint nodes. However, we observe that when calculating geodiverse paths in real-world topologies, iWPSP is more efficient in calculating the paths for node pair around the topology boundary. This is because by selecting waypoints based on a distance and a delta value, iWPSP has more control over the distance separated from the two paths. One better algorithm might be combining the two heuristics in calculating one topology, and this will be analysed in future work.

B. Routing Performance Comparison

We now present simulations using fiber-level topologies including Sprint [23], Level 3 [24], Internet2 [25], and Telia-Sonera [26]. We carry out the simulation once for each topology since there is no randomness in a given provider topology. We use CBR (constant bit rate) traffic, sending from each node to all others at a data rate of one packet per second. There are three area-based challenges we have simulated. From 20 to 40 s, the challenge occurs around Los Angeles, from 60 to 80 s in Kansas City, while the last challenge occurs at New York City from 100 to 120 s. The challenge locations come from the flow robustness analysis [5], and our challenge duration time is set as 20 seconds. We choose these different challenge areas so that the most vulnerable area is around Kansas City, due to its high betweenness as a major fiber exchange point for the US. The next damage level is around New York City. While it does not have many high-betweenness nodes, the network is dense and more nodes are challenged in a given radius. The least vulnerable among these big cities is around Los Angeles. The radii of the challenge areas are 300 km. By assuming the correct estimation of the challenge radius and position, we compare our protocol's performance with standard OSPF in terms of PDR (packet delivery ratio) as well as delay. Packet delivery ratio is the ratio of packets delivered divided to total packets sent, while delay is the time it takes for the data packet to travel end-to-end. We use all the same challenge areas throughout the topologies for easy comparison. The iWPSP heuristic is used in the GeoDivRP for calculating the geodiverse paths. MLW achieves the same PDR and delay result as iWPSP when the links are carefully modified to make sure paths calculated are d distance separated. Since ns-3 is an event-driven network simulator and the algorithm execution time is not included in the simulation time, the delay in ns-3 for both iWPSP and MLW is the same.

The Sprint physical network contains 77 nodes and 114 links. The PDR result for the Sprint network is shown in Figure 10. We compare the performance of our GeoDivRP with standard OSPF. The second challenge at Kansas City area happens at 60 s and GeoDivRP shows substantial performance improvement compared to OSPF. The PDR of OSPF drops to 75 percent and takes 10 s to converge while the time for GeoDivRP is within one second and the PDR only drops two percent before it converges. The paths calculated by GeoDivRP to bypass the challenge is shown in Figure 9. The red circle shown in this figure is the challenges area. The last challenge occurs from 100 s to 120 s and the difference in PDR between OSPF and GeoDivRP is small, only about one percent. This is because the challenge at New York City has little effect on the connectivity of this overall topology. The PDR for OSPF drops about one percent and takes ten seconds to recover, and there is no noticeable PDR drop for

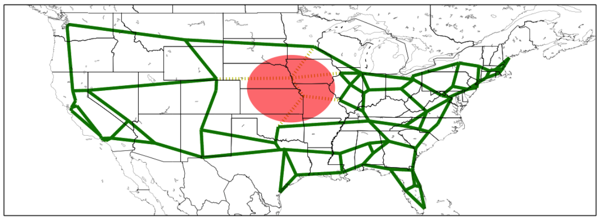

Fig. 9. Sprint topology under regional challenges

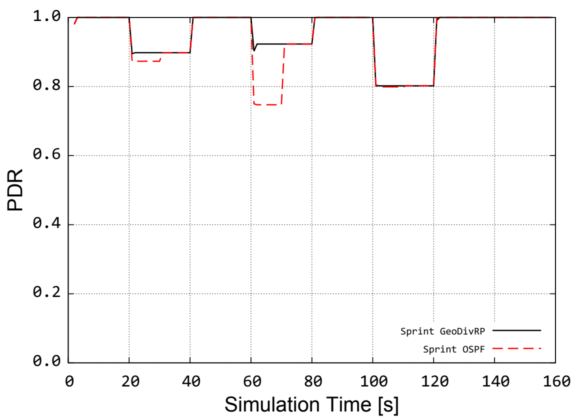

Fig. 10. Sprint PDR under regional challenges

our protocol. The first challenge happens at 20 s to 40 s and there is no noticeable PDR drop for both of the protocols. This is due to the same reason as in New York City and the damage is even less.

The delay analysis for Sprint network is shown in Figure 11. The reason that OSPF shows lower delay when the network is under challenge compared to GeoDivRP is because most of the data packets during the challenge have been dropped and the lost packets are not counted as delay; this is why there is a delay drop for OSPF before converging. Consider the first challenge in Figure 11, the delay for OSPF drops from 20 to 30 s due to the packet drops, while GeoDivRP converges and calculates geodiverse paths during that period of time and shows one second higher in delay. However, the extra delay is caused by extra path stretch due to routing packets around the challenged area. We also notice one delay bump for OSPF right after the challenge is finished. For example, in Figure 11, from 40 to 50 s, there is one increase in delay for OSPF. The same happens at 80 to 90 s, and 120 to 130 s. This is because OSPF needs to converge again after the topology has recovered from the challenge. In contrast, for our protocol, the convergence time is still one second and no noticeable delay increase is recorded.

The Level 3 physical network contains 99 nodes and 132 links. The PDR for the Level 3 network is shown in Figure 12. Since Level 3 shares geographical similarities to the Sprint network, we observe a similar PDR result. The challenge at Kansas City area reduces the PDR for OSPF significantly; it is even greater than for Sprint. This is because the Level 3

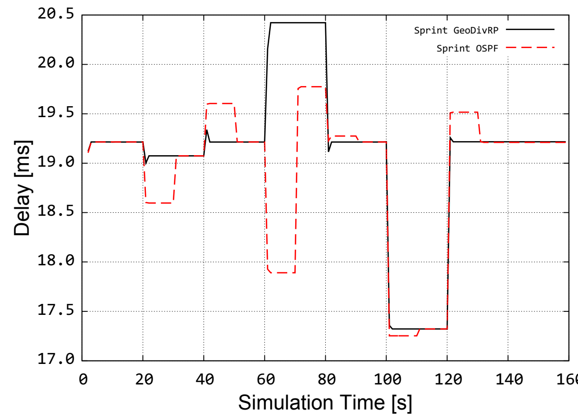

Fig. 11. Sprint delay under area-based challenges

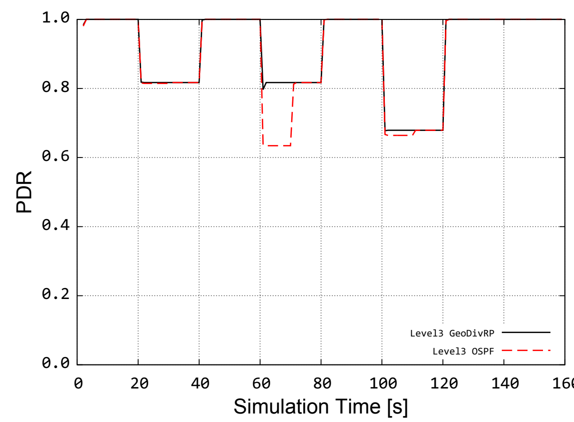

Fig. 12. Level 3 PDR under regional challenges

network lacks some of the nodes and links from Seattle to the Chicago area and the challenge around the Kansas City area causes more damage to the PDR. As shown in Figure 13 using Level 3 network. The similar challenge location as from the Sprint network has caused more nodes and links to fail. We are not showing the delay case for the Level 3 network as they are similar to those of the Sprint network.

The Internet2 physical network contains 16 nodes and 24 links. The PDR for the Internet2 network is shown in Figure 14. The challenged PDR and delay show a similar trend. The first challenge does damage to the network connectivity

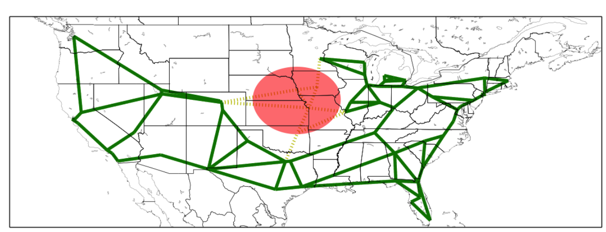

Fig. 13. Level 3 topology under regional challenges

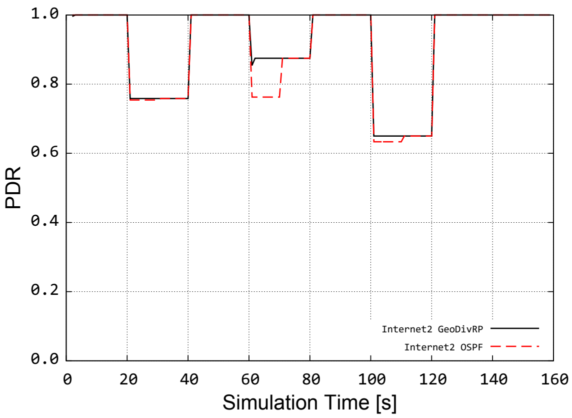

Fig. 14. Internet2 PDR under regional challenges

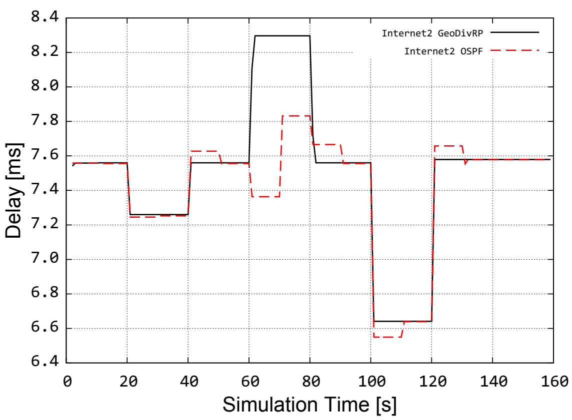

Fig. 15. Internet2 delay under regional challenges

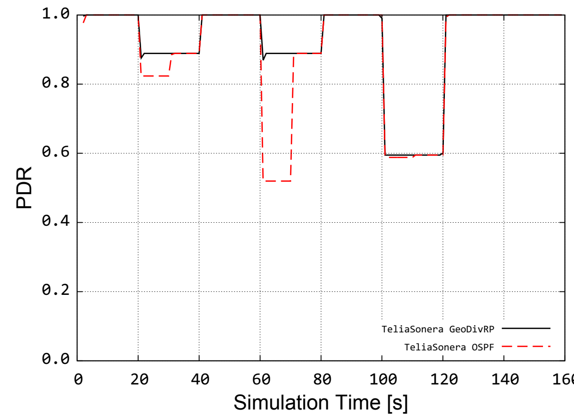

Fig. 16. TeliaSonera PDR under regional challenges

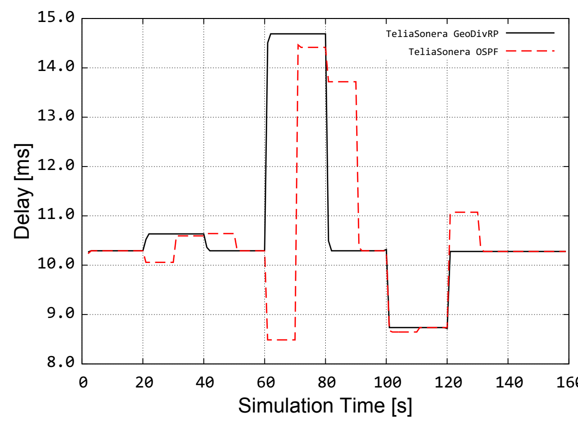

Fig. 17. TeliaSonera delay under regional challenges

and GeoDivRP converges within one second. The second challenge in Kansas City area causes OSPF to drop around ten percent in PDR and takes 10 s to converge and return the PDR to normal. The Los Angeles challenge has small impact on the network similar to the Sprint case. The delay analysis for the Internet2 network is shown in Figure 15. For the same reason, OSPF shows a lower delay compared to that of GeoDivRP during challenges from 20 to 30 s, 60 to 70 s, and 100 to 110 s.

The TeliaSonera physical network contains 18 nodes and 21 links. The PDR for TeliaSonera is shown in Figure 16. The second challenge at Kansas City area drops the PDR for OSPF to around 50 percent. This significant drop is caused by two reasons. First, the Kansas City node is connecting multiple nodes between the east and west coast. Second, the TeliaSonera network is very sparse so the damage from Kansas City node is greater than that for the other networks. However, GeoDivRP recovers from the damage in only one second and limits the PDR drop within one percent. The PDR case for both the second and third challenges are similar. At the same time, OSPF drops about one percent of total packets and recovers only after 10 s. The delay analysis is shown in Figure 17. The delay for OSPF is lower during challenges since the dropped packets are not counted in delay analysis. We notice that the delay increase after the challenge for OSPF at 80 to 90 s is larger than other challenge locations as well as the same challenge location in other topologies. This is because OSPF is using a path with more path stretch before convergence.

V. Conclusion and Future Work

We have proposed two geodiverse heuristics for efficiently solving the path geodiverse problem (PGD): iWPSP (iterative WayPoint Shortest Path) and MLW (Modified Link Weight). We have implemented both of the heuristics in ns-3 and verified their effectiveness in providing reliable paths in the face of area-based challenges. We have demonstrated the effectiveness of the heuristics in calculating and choosing different geographically diverse paths to meet the requirements from higher layers and its efficiency in routing the data traffic around the failure area. GeoDivRP shows significant improvement in both packet delivery ratio compared to OSPF and has comparable delay. The two heuristics for GeoDivRP use different mechanisms to calculate geodiverse paths. By carefully modifying the link weights, MLW is capable of providing one geodiverse path using one iteration of Dijkstra's algorithm. However, it is difficult to provide solutions when the paths required are for node pairs around the topology boundary, and the choice of link weight need to be carefully considered for different networks. On the other hand, iWPSP requires one extra Dijkstra's algorithm for each geodiverse path than MLW, therefore, it takes a bit greater time to execute and solve the PGD problem. However, by carefully selecting waypoint node and the parameters d and δ, different topologies are similar and iWPSP works better than MLW when dealing with node pairs in topology boundaries.

For future work, we will implement these two heuristics in a testbed to emulate its effectiveness in real-world routers and examine the mechanisms to incorporate our geodiverse routing protocol into the current Internet. We will extend this work to analyse how the geographic multi-path mechanism improves flow robustness. We will further extend this work to wireless networks and wired-wireless hybrid networks. We will incorporate our protocol with ResTP to test the protocol stack and analyse the protocol performance with multiple geodiverse paths. We will fully analyse the relative benefits of the two heuristics and their combination, and enable GeoDivRP to automatically choose different heuristics in different network topologies.

ACKNOWLEDGMENTS

The authors would like to thank the members of the ResiliNets group for discussions which led to this work. This research was supported in part by NSF grant CNS-1219028 and CNS-1217736 (Resilient Network Design for Massive Failures and Attacks) at KU and UMKC.

REFERENCES

[1] J. P. G. Sterbenz, D. Hutchison, E. K. Çetinkaya, A. Jabbar, J. P. Rohrer, M. Schöller, and P. Smith, "Resilience and survivability in communication networks: Strategies, principles, and survey of disciplines," *Computer Networks*, vol. 54, no. 8, pp. 1245–1265, 2010.

[2] R. Cohen, K. Erez, D. ben Avraham, and S. Havlin, "Resilience of the Internet to Random Breakdowns," *Phys. Rev. Lett.*, vol. 85, pp. 4626–4628, November 2000.

[3] D. Magoni, "Tearing down the internet," *IEEE J.Sel. A. Commun.*, vol. 21, pp. 949–960, September 2006.

[4] E. K. Çetinkaya, D. Broyles, A. Dandekar, S. Srinivasan, and J. P. G. Sterbenz, "Modelling Communication Network Challenges for Future Internet Resilience, Survivability, and Disruption Tolerance: A Simulation-Based Approach," *Telecommunication Systems*, vol. 52, no. 2, pp. 751–766, 2013.

[5] Y. Cheng, J. Li, and J. P. G. Sterbenz, "Path Geo-diversification: Design and Analysis," in *Proceedings of the 5th IEEE/IFIP International Workshop on Reliable Networks Design and Modeling (RNDM)*, (Almaty), September 2013.

[6] M. J. F. Denise M. B. Masi, Eric E. Smith, "Understanding and Mitigating Catastrophic Disruption and Attack," *Sigma Journal*, pp. 16–22, September 2010.

[7] J. W. Suurballe, "Disjoint paths in a network," *Networks*, vol. 4, no. 2, 1974.

[8] J. W. Suurballe and R. E. Tarjan, "A Quick Method for Finding Shortest Pairs of Disjoint Paths," *Networks*, vol. 14, no. 2, 1984.

[9] E. W. Dijkstra, "A Note on Two Problems in connection with Graphs," *Numerische Mathematik*, vol. 1, pp. 269–271, 1959.

[10] J. P. G. Sterbenz, D. Hutchison, E. K. Çetinkaya, A. Jabbar, J. P. Rohrer, M. Schöller, and P. Smith, "Redundancy, Diversity, and Connectivity to Achieve Multilevel Network Resilience, Survivability, and Disruption Tolerance (invited paper)," *Springer Telecommunication Systems*, 2012. (accepted April 2012).

[11] Y. Dadkar, D. Jones, and L. Nozick, "Identifying Geographically Diverse Routes for the Transportation of Hazardous Materials," *Transportation Research Part E: Logistics and Transportation Review*, vol. 44, no. 3, pp. 333–349, 2008.

[12] V. Akgün, E. Erkut, and R. Batta, "On Finding Dissimilar Paths," *European Journal of Operational Research*, vol. 121, no. 2, pp. 232–246, 2000.

[13] M. Kuby, X. Zhongyi, and X. Xiaodong, "A Minimax Method for Finding the k Best Differentiated Paths," *Geographical Analysis*, vol. 29, no. 4, pp. 298–313, 1997.

[14] R. Martí, J. Luis González Velarde, and A. Duarte, "Heuristics for the bi-objective Path Dissimilarity Problem," *Comput. Oper. Res.*, vol. 36, pp. 2905–2912, Nov. 2009.

[15] R. Bhandari, *Survivable Networks: Algorithms for Diverse Routing*. Norwell, MA, USA: Kluwer Academic Publishers, 1998.

[16] M. Motiwala, M. Elmore, N. Feamster, and S. Vempala, "Path Splicing," in *Proceedings of the ACM SIGCOMM*, (Seattle, WA), pp. 27–38, August 2008.

[17] J. Lang, Y. Rekhter, and D. Papadimitriou, "RSVP-TE Extensions in Support of End-to-End Generalized Multi-Protocol Label Switching (GMPLS) Recovery." RFC 4872 (Proposed Standard), May 2007. Updated by RFC 4873.

[18] J. P. Rohrer, A. Jabbar, and J. P. G. Sterbenz, "Path diversification: A multipath resilience mechanism," in *Proceedings of the IEEE 7th International Workshop on the Design of Reliable Communication Networks (DRCN)*, (Washington, DC), pp. 343–351, October 2009.

[19] J. P. Rohrer, A. Jabbar, and J. P. Sterbenz, "Path Diversification for Future Internet End-to-End Resilience and Survivability," *Springer Telecommunication Systems*, 2012.

[20] "The ns-3 network simulator." http://www.nsnam.org, July 2009.

[21] J. Moy, "OSPF specification." RFC 1131 (Proposed Standard), Oct. 1989. Obsoleted by RFC 1247.

[22] S. Orlowski, M. Pióro, A. Tomaszewski, and R. Wessäly, "SNDlib 1.0–Survivable Network Design Library," *Networks*, vol. 55, no. 3, pp. 276–286, 2010.

[23] E. K. Çetinkaya, M. J. F. Alenazi, Y. Cheng, A. M. Peck, and J. P. G. Sterbenz, "On the Fitness of Geographic Graph Generators for Modelling Physical Level Topologies," in *Proceedings of the 5th IEEE/IFIP International Workshop on Reliable Networks Design and Modeling (RNDM)*, (Almaty), September 2013.

[24] "Level 3 network map." http://maps.level3.com.

[25] "Internet2." http://www.internet2.edu.

[26] "TeliaSonera." http://www.teliasoneraic.com.