MODRA: Multi-Objective Distributed Routing Algorithm

Maxim Prihodko 1, Abdurasul Rakhimov 1, Biswarup Das 1, Artem Krantsevich 1, Pingan Yang 1, Jian Jin 1, Zhibo Hu 1, and Longfei Dai 1

1Affiliation not available

December 7, 2023

Abstract

This paper develops a systematic strategy to construct a model of an IP Network with multiple weight links and proposes a multi-objective distributed routing algorithm (MODRA) for the One-to-All and All-to-All Multiobjective Shortest Path (MOSP) Problems. The formal proof of a loop-free routing in a distributed mode is given, as well as extensive experiments are performed in simulated networks to show the algorithm performance. The proposed algorithm is based on the single dimension path conversion principle and constructs a Shortest Path Tree w.r.t. the given single dimension path conversion metric. In this work, the proposed MODRA is tested on four network topologies with two-weight links and different configurations, and the performance is evaluated w.r.t. multiple upperbound path constraints. The algorithm is used to compute a Routing Information Base (RIB) table in each node. Then the distributed hop-by-hop packet routing is simulated and the actual path traversed by a packet is compared to the initially computed one. Our approach supports arbitrary topology, number of additive link weights and shows good performance in terms of computing feasible paths satisfying the given multiple upper-bound constraints. The proposed algorithm is implemented and tested in a simulated environment, and the framework of this work could be adopted to design other routing algorithms for multi-objective distributed routing in IP Networks. The algorithm performance evaluation shows its ability to compute efficient paths w.r.t. multiple upper-bound constraints, guarantee loop-free distributed routing and satisfy strict execution time requirements. The algorithm is fully compatible with the current router architecture and can be easily implemented in a router.
MODRA: Multi-Objective Distributed Routing Algorithm

Maxim Prihodko, Member, IEEE, Abdurasul Rakhimov, Biswarup Das, Member, IEEE, Artem Krantsevich, Pingan Yang, Jian Jin, Zhibo Hu, Longfei Dai

Abstract—This paper develops a systematic strategy to construct a model of an IP network with multiple weight links and proposes a multi-objective distributed routing algorithm (MODRA) for the One-to-All and All-to-All Multiobjective Shortest Path (MOSP) Problems. The formal proof of a loop-free routing in a distributed mode is given, as well as extensive experiments are performed in simulated networks to show the algorithm performance. The proposed algorithm is based on the single dimension path conversion principle and constructs a "Shortest” Path Tree w.r.t. the given single dimension path conversion metric. In this work, the proposed MODRA is tested on four network topologies with two-weight links and different configurations, and the performance is evaluated w.r.t. multiple upper-bound path constraints. The algorithm is used to compute a Routing Information Base (RIB) table in each node. Then the distributed hop-by-hop packet routing is simulated and the actual path traversed by a packet is compared to the initially computed one. Our approach supports arbitrary topology, number of additive link weights and shows good performance in terms of computing feasible paths satisfying the given multiple upper-bound constraints. The proposed algorithm is implemented and tested in a simulated environment, and the framework of this work could be adopted to design other routing algorithms for multi-objective distributed routing in IP Networks. The algorithm performance evaluation shows its ability to compute efficient paths w.r.t. multiple upper-bound constraints, guarantee loop-free distributed routing and satisfy strict execution time requirements. The algorithm is fully compatible with the current router architecture and can be easily implemented in a router.

Index Terms—routing, distributed routing, multi-objective routing, multi-objective optimization, loop-free

I. INTRODUCTION

The problem of identifying routes through a network is called routing. For example, in the communication networks the most widely used routing protocols are Interior Gateway Protocol (IGP) and Border Gateway Protocol (BGP), and both of them consider the routing problem as identifying the shortest routes in a network subject to a single constraint or cost. There are well known state-of-the-art algorithms for this case, namely Shortest Path First Algorithms by Dijkstra and Bellman-Ford, which use a single cost, single constraint approach and efficiently compute the shortest routes in the network.

Recently, as a consequence of the communication technology development a wide variety of new use cases emerged that require considering multiple constraints at the same time for identifying efficient routes, for example in Internet of Things (IoT) [1], 6G [2], Real-time Vehicle Routing Problem (VRP) [3], Terrain [4] and Urban Transit [5] routing.

The problem of routing while considering multiple constraints has been attracting academic and industry attention for more than 20 years and has been addressed in a number of ways [6], [7], [8], [9], [10], [11]. It is related to Multi-Objective Shortest Path Problem (MOSPP) and is known to be NP-hard. In [12] Hansen proved that, in the worst case, the number of non-dominated paths may increase exponentially with the number of nodes, making this problem intractable in any network of a reasonable scale. Despite tens of years of research, there is still no state-of-the-art algorithm for such problem as Dijkstra for the single constraint case.

In order to solve this problem, one way is to invent an efficient algorithm which could work in distributed mode, similar to the Dijkstra algorithm, but taking multiple metrics into account for each link simultaneously. The attempts to modify Dijkstra algorithm and apply it to multicriteria shortest path problem have been made since 1980-s, when Ernesto Queiros Vieira Martins [13] proposed two algorithms for this problem, one of which was an immediate generalization of the multiple labelling scheme algorithm of Hansen for the bicriteria case [12]. Recently more improvements to the classic Dijkstra algorithm have been proposed to adapt it to the multiple constraints case. In [14] the authors introduced a new label-setting algorithm for the MOSPP that computes a minimum complete set of efficient paths for a given instance and suggested to call the algorithm Multiobjective Dijkstra Algorithm (MDA). Based on [14], one more algorithm, namely Targeted Multiobjective Dijkstra Algorithm (T-MDA), was proposed in [15] by the same authors. The T-MDA is based on MDA and equips it with A*-like techniques. A version tuned for the biobjective case, the...
proves 2 important MODRA properties, in section V we give the theoretical proof of the loop-free routing in distributed mode, section VI provides the simulation results for our solution, section VII considers the impact of path metric on MODRA performance, and in section VIII we analyze MODRA robustness in distributed mode. Section IX concludes the paper.

II. MULTI-OBJECTIVE DISTRIBUTED ROUTING PROBLEM

A. Network Model

We consider the network as a graph $G = (V, E)$, where $V = \{n_i\}, i = 1, \ldots, N$ is the set of $N$ vertices, and $E = \{e_{ij}\}, i, j \in \{1, \ldots, N\}$ is the set of $M$ edges, connecting these vertices. We assume here for the simplicity that all the nodes in the graph are enumerated.

The edges connecting the nodes represent bi-directional and full-duplex links, so the graph is directed. All edges in the graph are assigned with multiple weights $d_{ij}, l_{ij}, \ldots$, and these weights in general are different in different directions for the same edge, i.e. $d_{ij} \neq d_{ji}$ for example.

Without loss of generality, in the following part of the paper we consider the case of $2$ additive metrics, while all the results can be easily generalized for an arbitrary number of additive metrics. It is worth of mentioning, that a multiplicative metric can also be reduced to an additive one by applying the logarithm function.

B. Network constraints

The edges $e_{ij}$ connecting the nodes $i$ and $j$ are assigned with delay and loss values $d_{ij}$ and $l_{ij}$ correspondently. To evaluate later the performance of the algorithm in a real-world scenario appropriate for the case of additive link metrics, we consider the constraints which are set for the Paths to be satisfied for every possible Origin $(O)$ – Destination $(Dest)$ pair, namely network-wide delay bound $D$ and loss bound $L$.

Path delay is defined as follows:

$$\text{Delay}_{ODest} = \sum_{e \in \text{Path}_{ODest}} \text{delay}_e, \tag{1}$$

where $\text{delay}_e$ describes the delay for the edge $e$ along the currently established Path from the Origin to the Destination.

A similar expression can be written for Path loss:

$$\text{Loss}_{ODest} = \sum_{e \in \text{Path}_{ODest}} \text{loss}_e. \quad \tag{2}$$

We say that a computed path is feasible (or efficient), if:

$$\text{Delay}_{ODest} \leq D, \quad \text{Loss}_{ODest} \leq L. \quad \tag{3}$$
Then the feasible paths area described by (3) forms a rectangle of the size $D \times L$ in a 2-dimensional plane with $\text{Delay}$ and $\text{Loss}$ dimensions correspondently (see Fig. 1).

It is easy to see, that by normalizing each edge weights by the constraints $D$ and $L$ the feasible paths area can be turned into a unit square ($\text{normalized feasible paths area}$). And for a real network it is beneficial for a multi-objective distributed routing algorithm to compute as many efficient paths within the normalized feasible paths area as possible.

From this point we assume, that the edge weights are normalized by the given constraints and the $\text{normalized}$ feasible paths area, or a unit square, is considered, even if the word $\text{normalized}$ is omitted.

C. Multi-Objective Distributed Routing Problem (MODRP)

Not only finding the feasible paths in a multi-objective scenario is difficult. The biggest challenge is to find an agreed set of paths for each node, such that loop-free routing can be guaranteed in distributed mode.

Following this, the main task of distributed routing is, for a given Origin node, to assign to each Destination node a specific Next Hop, such that taking independently a single Next Hop for packet forwarding at each node you can guarantee the packet delivery to the Destination node without creating loops. For example, when Dijkstra algorithm is used, it computes the trees of shortest paths, which guarantee that in any intermediate node the same segment of the original shortest path will be computed to a Destination, thus satisfying loop-free and fully-distributed properties.

Unfortunately it is easy to show, that in general finding 100% feasible paths in a multi-objective scenario and ensuring loop-free distributed routing satisfying the constraints are two contradictory goals. The reason is that in the current IGP protocols paradigm the routing is Destination-based and Source-agnostic. While in some cases Source-dependent decisions are required to guarantee the constraints satisfaction.

In Fig. 2 an example of a simple network is shown. The edge weights are show in the figure, and the constraints are set as $D = 1$ and $L = 1$.

It easy to see, that both for source node $S_1$ and source node $S_2$ there’s a path to destination node $D$, satisfying the constraints: $S_1 \rightarrow N$, $N \rightarrow_a D$ and $S_2 \rightarrow N$, $N \rightarrow_b D$. Though, from intermediate node $N$ to $D$ these paths have to use different edges — $a$ for $S_1$ and $b$ for $S_2$. At the same time, in the current distributed routing paradigm only one Next Hop node (link to be used) can be assigned to destination $D$ in node $N$. That is why, no matter which edge is used, at least one of the nodes $S_1$ and $S_2$ will violate the constraints in the process of distributed hop-by-hop routing.

This is the reason why in our algorithm design we prioritize the loop-free guarantees in distributed routing mode over the ability to compute 100% of the theoretically feasible paths, thus providing an efficient, but best-effort solution in the second aspect.

III. MULTI-OBJECTIVE DISTRIBUTED ROUTING ALGORITHM (MODRA)

A. MODRP from Feasible Paths Area Perspective

Though the formulated above MODRP routing problem is related to classic multi-objective problems like MOSPP and Multi-Constrained Path (MCP) problem [17], [18], significant difference is present.

Similar to MOSPP and MCP problem multiple edge weights and path objectives are considered in MODRP, but in contrast to them One-to-All routing problem is being solved. Moreover, not only multiple paths need to be found to all the possible destinations in the network, while satisfying the given multiple constraints, but these paths can not be absolutely independent and need to align into a tree structure.

At the same time, similar to MCP problem, in MODRP for each possible destination in the network it is enough to find a single feasible path, satisfying the given multiple constraints, while maximizing the number of such feasible paths in the overall routing tree. This means, that similar to MCP problem, where we need to find a path from a feasible paths area (3) (or a normalized feasible paths area), in MODRP we need to find the paths, such that the routing tree fits in the feasible paths area as much as possible (see Fig. 3).
Fig. 3. A routing tree fitting in the normalized feasible paths area.

The paths dependency on each other, caused by the tree structure, introduces important specificity — if a path to some node violates any constraint, i.e. the node itself lies outside the feasible paths area, then the paths to all its’ children nodes in the tree will also violate the constraint, being infeasible.

This finally leads to the main motivation behind the algorithm design. Prioritizing and minimizing any specific metric will deform the routing tree, causing it to leave the feasible paths area (see Fig. 4), and deviating from the optimal routing tree expansion direction, which is the diagonal of the feasible paths area (see Fig. 3). The reason is, in the direction of the diagonal all the path metrics can be used to their fullest, until reaching the upper bounds (3).

That is why the key idea of the algorithm is to guide the routing tree expansion in the direction of the diagonal of the normalized feasible paths area. This is done by the introduction of a geometric measure for a path length estimation and building the routing tree to minimize the geometric length of a newly discovered path (distance to a newly attached to the tree node). Basically, this means that the final tree is actually a Shortest Path Tree, though a different meaning of “shortest” is used.

B. Multi-Objective Distributed Routing Algorithm

We extend Dijkstra’s Shortest Path First (SPF) algorithm to follow the logic described in Subsection III-A.

First, we introduce the separate accumulated link weights along that path: $\text{distD}[v]$ for Delay and $\text{distL}[v]$ for Loss. In general case, the number of additional separately accumulated link weights is equal to the number of additive metrics used to describe a link.

Second, we introduce a new metric function to measure a path length $DVEC$ (see Algorithm 1). The most intuitive way is to use a well-known geometric distance metric, i.e. $l^2$-norm:

$$|x| = \sqrt{\sum_{i=1}^{n} x_i^2}. \quad (4)$$

Thus, in our case for a path $P$ with accumulated metrics $\text{distanceD}$ and $\text{distanceL}$ the length is computed as follows:

$$|P| = \sqrt{\text{distD}^2 + \text{distL}^2}. \quad (5)$$

Finally, we choose the next node $v$ to be attached to the tree based on a newly introduced path length: the next node is chosen such that the global $\text{distance}[v]$ to this node, computed using $l^2$-norm, is minimized. Literally speaking, we embed the shortest path tree in a circle of the minimum possible radius with the center in Cartesian plane origin of coordinates, thus maximizing the part of the tree fitting into the feasible paths area. And the Source node is placed in the center of this circle, i.e. is represented by the point with $(0, 0)$ coordinates (see Fig. 5).
When we consider the Shortest Path Tree construction as an embedding process, it is possible to show why the resulting tree tends to grow in the direction of the diagonal of the normalized feasible paths area. Though, in this paper we are not pretending to give a formal proof of this statement. Nor we are stating that this is true in any case.

In Fig. 6 a simple example is shown. Each time a new node is attached to the shortest path tree, in general, multiple tree expansion options (candidate nodes) are considered. If we keep only Pareto-optimal, or non-dominated candidate nodes, they will form a so-called Pareto-optimal front. A part of this front, namely nodes A, B and C, is shown in Fig. 6.

When an algorithm prioritizes Delay, node A will be chosen, approaching us to the upper bound of the feasible paths area and quickly leading to Loss constraint violation. When an algorithm prioritizes Loss, node C will be chosen, this time approaching us to the right border of the feasible paths area and quickly leading to Delay constraint violation. This behavior can be clearly observed in Fig. 4. At the same time, the closest to the Source in terms of geometric distance node B, while tending to the diagonal of the normalized feasible paths area, leaves enough space for further tree expansion both in the direction of Delay and Loss, leading to higher overall number of feasible paths in the routing tree.

Finally, Algorithm 1 presents MODRA pseudo-code in detail. In the following sections we provide the formal proof that it is loop-free in a distributed mode, consider alternative metric functions $l^p$, $p \geq 3$ and $l^\infty$ and their impact on the algorithm performance and analyze the algorithm performance using intensive simulations.

Algorithm 1 Multi-Objective Distributed Routing Algorithm

1: function DVEC(distD, distL, delay, loss)
2:     dist ← (distD + delay)$^2$
3:     dist ← dist + (distL + loss)$^2$
4:     return $\sqrt{dist}$
5: end function
6: procedure MODRA
7:     $G \leftarrow$ Graph.
8:     source $\leftarrow$ root_node.
9:     $Q \leftarrow$ vertex set.
10:     for each vertex $v$ in $G$ do
11:         distance[$v$] $\leftarrow$ INFINITY
12:         distD[$v$] $\leftarrow$ INFINITY
13:         distL[$v$] $\leftarrow$ INFINITY
14:         parent[$v$] $\leftarrow$ NULL
15:     add $v$ to $Q$
16:     end for
17:     distance[source] $\leftarrow$ 0
18:     distD[source] $\leftarrow$ 0
19:     distL[source] $\leftarrow$ 0
20:     while $Q$ is not empty do
21:         $u \leftarrow$ vertex in $Q$ with min distance[$u$].
22:         remove $u$ from $Q$.
23:         for each neighbor $v$ of $u$ do
24:             alt $\leftarrow$ DVEC(distD[$u$], distL[$u$],
25:                 $(u,v).D, (u,v).L$)
26:             if alt $<$ distance[$v$] then
27:                 distance[$v$] $\leftarrow$ alt.
28:                 distD[$v$] $\leftarrow$ distD[$u$] + $(u,v).D$
29:                 distL[$v$] $\leftarrow$ distL[$u$] + $(u,v).L$
30:                 parent[$v$] $\leftarrow$ $u$.
31:             end if
32:         end for
33:     end while
34:     return distance[], parent[]
35: end procedure

IV. MODRA THEORETICAL ANALYSIS

In this section we prove 2 important MODRA properties:

- for a given Source node $S$ in a graph with non-negative edge weights and at least one strictly positive weight, MODRA computes the shortest of all possible paths from $S$ to every Destination $D$ in terms of $l^p$-norm (Correctness);
- MODRA computes non-dominated, Pareto-optimal paths (Pareto-optimality).

A. Correctness

Theorem 1. For a given Source node $S$ in a graph with non-negative edge weights and at least one strictly positive weight, MODRA computes the shortest of all possible paths from $S$ to every Destination $D$ in terms of $l^p$-norm, $p \in \mathbb{N}$. 
Proof. To prove the correctness of MODRA, in our proof we use the same logic as is used to prove Dijkstra algorithm correctness. The nodes of a graph are divided into 2 sets — unvisited (set Q in MODRA) and visited nodes. And the proof is constructed by induction on the number of visited nodes.

Initially, similar to Dijkstra, all nodes of the graph are marked as unvisited (Q is equal to the full vertex set of the graph). Then, step by step, a single node is selected, processed and marked as visited.

The invariant hypothesis of the induction is as follows: For each visited node \( v \) and path \( P_v \) from Source to \( v \) constructed by MODRA, \( \text{distance}[v] = dvec(P_v) \) is the shortest distance from Source to \( v \) in terms of \( l^p \)-norm. And for each unvisited node \( u \), \( \text{distance}[u] \) is the shortest distance from Source to \( u \) in terms of \( l^p \)-norm when traveling via visited nodes only, or infinity, if no such path exists.

The base case is when there is just one visited node, namely the initial Source node. It is obvious, that in this case the hypothesis is true.

Let us assume now, that the hypothesis is true for \( k-1 \) visited nodes. And let us choose \( u \) as the next visited node according to MODRA. By contradiction, we prove that \( \text{distance}[u] = dvec(P_u) \) is the shortest distance in terms of \( l^p \)-norm from Source to \( u \).

If there is a shorter path \( \tilde{P}_u \), there are two cases:

- \( \tilde{P}_u \) contains another unvisited node;
- \( \tilde{P}_u \) contains only visited nodes.

In the first case, let \( w \) be the first unvisited node in \( \tilde{P}_u \). First of all, \( dvec(P_u) = \text{distance}[u] < \text{distance}[w] \), as MODRA selected \( u \) instead of \( w \) as the next visited node.

Secondly, by the induction hypothesis, the shortest path lengths in terms of \( l^p \)-norm from Source to \( w \) through visited nodes only are \( \text{distance}[u] \) and \( \text{distance}[w] \) respectively. In case of non-negative metrics with at least one strictly positive metric, this means that the cost of going from Source to \( u \) through \( w \) — \( dvec(\tilde{P}_u) \) is greater than \( \text{distance}[w] \).

This leads to a contradiction:

\[
\text{distance}[w] > \text{distance}[u] = dvec(P_u) \quad \text{and} \quad dvec(P_u) > dvec(\tilde{P}_u) > \text{distance}[w].
\]

In the second case, let \( w \) be the last but one node in \( \tilde{P}_u \). This means that \( dvec(P_u + \{w, u\}) < dvec(P_u) \), which is a contradiction, because by the time \( w \) is set as visited, \( \text{distance}[u] = dvec(P_u) \) is set to at most \( dvec(P_u + \{w, u\}) \).

For all other visited nodes \( v \), the induction hypothesis stays true, as MODRA does not change the paths to them. And after processing \( u \) it is still true, that for each unvisited node \( w \), \( \text{distance}[w] \) is the shortest distance in terms of \( l^p \)-norm from Source to \( w \) using visited nodes only.

Finally, when all nodes are visited, the shortest path in terms of \( l^p \)-norm from Source to any node \( v \) consists of only visited nodes, and \( \text{distance}[v] = dvec(P_v) \) is the shortest distance in terms of \( l^p \)-norm. The statement stays true for an arbitrary \( l^p \)-norm, \( p \in \mathbb{N} \).

\[ \square \]

B. Pareto-optimality

**Theorem 2.** MODRA computes non-dominated, Pareto-optimal paths.

**Proof.** Using previously proven Correctness property of MODRA, it is easy to show by contradiction, that MODRA computes Pareto-optimal paths. Indeed, let \( P_v \) be any shortest path with path metrics \( m^v \) computed by MODRA, which is dominated by some other path \( P_w \) with path metrics \( m^w \). By the definition of dominance, this means that for all path metrics \( m^v \leq m^w \), and for at least one of them the strict inequality holds: \( \exists k : m^v_k < m^w_k \). But at the same time this means, that \( dvec(P_w) < dvec(P_v) \), which contradicts the fact, that \( P_v \) is the shortest path in terms of \( l^p \)-norm, \( p \in \mathbb{N} \).

\[ \square \]
1) Conclusion: MODRA can be considered as an optimal Pareto front sampling method, selecting the single non-dominated shortest path (in terms of $l^p$-norm) for each Source-Destination pair in the graph.

V. MODRA LOOP-FREE PROPERTY IN DISTRIBUTED MODE

In this section we provide the formal proof that MODRA is loop-free in a distributed mode. We do it by contradiction — similar to the proof of Dijkstra algorithm loop-free property.

Let us consider distributed, i.e., hop-by-hop routing from Source node to Destination. At the first step Source node, using MODRA, computes path $d_S$ to Destination, such that $H_1$ is the first Next Hop. The rest of the path, computed by Source, can be represented as an aggregated vector $\delta_{H_1}$, (see Fig. 7). It is obvious, that $|\delta_{H_1}| < |d_S|$, due to triangle inequality $l^2$-norm satisfies and the fact, that the edge weights (link metrics) are, first, additive, and, second, non negative (with at least one strictly positive).

Let us consider now some intermediate node $H_i$ along the path computed in a distributed mode. In general, this node can be different from the $i$-th node of the originally computed path $d_S$. Node $H_i$ runs MODRA and computes its’ own shortest path $d_{H_i}$ and Next Hop $H_{i+1}$. As MODRA minimizes $l^2$-norm, or distance to Destination, the following is true: $|d_{H_i}| < |\delta_{H_i}|$ — the newly computed path to Destination is no longer than the rest of the path to Destination from the node $H_i$, computed by the previous node (see Fig. 8).

At the same time:

$$|d_{H_i}| < |\delta_{H_i}| < |d_{H_{i-1}}|, \quad |\delta_{H_{i+1}}| < |d_{H_i}|,$$

thus,

$$|\delta_{H_{i+1}}| < |\delta_{H_i}|.$$

Finally, a path from Source to Destination, computed in a distributed mode using MODRA, can be described as a sequence of next hops $H_1, H_2, \ldots, H_i, \ldots, H_k, \ldots$.

We’ve just shown that the distance to Destination from these next hops only decreases:

$$|d_{H_1}| > |d_{H_2}| > \cdots > |d_{H_i}| > \cdots > |d_{H_k}| > \cdots$$

Imagine we have a loop, then some nodes in the sequence of the next hops are the same, for example $H_i$ and $H_k$. This means, that for the same node we’ve found a shorter path on the plane to the destination which is a contradiction, as the algorithm finds the shortest path. This proves the loop-free property, which stays true for an arbitrary $l^p$-norm, $p \in \mathbb{N}$.

VI. MODRA SIMULATION

The evaluation of MODRA is performed via simulations in virtual environments. This section describes the setup of the virtual environments. Then we analyze the results.

A. Simulation Environment

To comprehensively evaluate the proposed algorithms, virtual networks with different topologies and scales are built. The topologies under consideration are: Grid network, Full-Mesh network, Mouth-Like and Dual-Homed networks.

1) Grid network: A Grid network model is shown in Fig. 9.

We consider square Grid networks of the following size: $45 \times 45$. Each node is connected to its’ direct neighbors in the Grid. There are no diagonal connections.

The edges $e_{ij}$ connecting the nodes $i$ and $j$ are assigned with delay and loss values $d_{ij}$ and $l_{ij}$ correspondently. Delays are assigned randomly using normal distribution $N(7.5,1.25)$. The losses are assigned randomly from the set $\{0.001, 0.002, 0.003, 0.004, 0.005\}$ using discrete uniform distribution.

2) Full-Mesh network: A Full-Mesh network model is shown in Fig. 10. The network topology is a fully-connected graph. For example, it can represent an overlay network structure.

We consider Full-Mesh Networks with 200 nodes, such that the number of edges — $200 \times 199$ — is similar to the number of edges in a $45 \times 45$ Grid network. Link weights are assigned similar to Grid network case.

3) Dual-Homed and Mouth-like network: A general Carrier-like network model is shown in Fig. 11. The network consists of 2 parts: a Core and a set of $k$ Groups connected to the Core.

In our simulations we use a Grid network as the Core. $k$ Groups are connected to the Core — each Group is connected to a single pair of boundary nodes in the Core.
There are Groups of 2 kinds: Mouth-Like groups and Dual-Homed groups.

**Dual-Homed** group consists of $2m$ nodes each one connected to both nodes in the Core that are assigned to the Group.

**Mouth-like** group consists of $2m$ nodes divided into $m$ pairs. The nodes in each pair are connected to each other. The first node of a pair is connected to the first out of two nodes in the Core that are assigned to the Group. The second node of a pair is connected to the second out of two nodes in the Core that are assigned to the Group.

We consider both Mouth-like (see Fig. 12) and Dual-Homed (see Fig. 13) networks with 1016 nodes — 36 in the Core ($6 \times 6$ Grid network with 20 boundary nodes) connected with 10 groups each with 98 nodes ($m = 49$). Link weights are assigned similar to Grid network case.

**B. Simulation Scenario**

For the simulations we generate 100 networks of each kind and scale and set the constraints as maximum delay (packet loss) value along the shortest paths calculated with Floyd-Warshall algorithm and multiplied by a constraint rate from the set \{0.95, 0.9, 0.85, 0.8, 0.75, 0.7, 0.65, 0.6, 0.55, 0.5\}. The performance of the algorithm is evaluated using the coverage rate metric $CR = OD_{\text{found}} / OD_{\text{feasible}}$, where $OD_{\text{found}}$ is the number of found OD-pairs that satisfy all the necessary Path constraints, $OD_{\text{feasible}}$ the total number of all theoretically feasible OD-pairs in the network.

The coverage rate is measured for OD-pairs formed by boundary nodes, i.e. nodes on the boundary of the Grid network or nodes inside Dual-Homed and Mouth-like groups. For a Full-Mesh network all nodes are considered as boundary.

**C. Multi-Objective Distribute Routing Algorithm Performance**

As mentioned in Subsection III-A, MODRA is supposed to construct a more balanced Shortest Path Tree, compared to ones computed by Dijkstra algorithm using just Delay or Loss as the single metric. This is the reason why we analyze MODRA performance in comparison to Delay SPT and Loss SPT as the baseline.

Network topologies considered for the evaluation are significantly different. While $45 \times 45$ Grid network diameter
is 88, for Dual-Homed and Mouth-like networks it is only 3 and 4 correspondently. And for Full-Mesh network it is just 1. This allows us to analyze MODRA performance in significantly different environments to show its' robustness and high efficiency.

1) Grid network: It is clearly seen from Fig. 14, that MODRA significantly outperforms both Delay SPT and Loss SPT. At the same time, Loss SPT outperforms Delay SPT and forms the trend which MODRA Coverage Rate follows. For example, both MODRA and Loss SPT encounter difficulties in finding feasible paths for the constraint multiplier equal to 0.7. The difference in Delay and Loss distributions can be a reason of such behavior.

Overall, MODRA Coverage Rate is noticeably higher than 90% and the minimal value is 94.58% for the constraint multiplier equal to 0.5, when Loss SPT Coverage Rate almost meets MODRA and is equal to 94.88%.

2) Full-Mesh network: In a Full-Mesh network the picture is qualitatively different (see Fig. 15), though again MODRA outperforms both Delay SPT and Loss SPT or shows exactly the same performance in terms of Coverage Rate. After noticeable MODRA superiority for high values of the constraint multiplier, near 0.75 the Coverage Rates become literally indistinguishable.

3) Dual-Homed network: Similar to Grid network, in a Dual-Homed network MODRA significantly outperforms both Delay SPT and Loss SPT (see Fig. 16). At the same time, in this case Delay SPT outperforms Loss SPT.

MODRA Coverage Rate is noticeably higher than 90% and the minimal value is 95.34% for the constraint multiplier equal to 0.5, when Loss SPT Coverage Rate almost meets MODRA and is equal to 94.88%.

4) Mouth-like network: In a Mouth-Like network MODRA again significantly outperforms both Delay SPT and Loss SPT (see Fig. 17). Similar to Dual-Homed network, Delay SPT also again outperforms Loss SPT.

MODRA Coverage Rate is noticeably higher than 90% and the minimal value is 95.58% for the constraint multiplier equal to 0.5, while for Delay SPT at the same time the Coverage Rate is more than 2% points lower and is equal to 93.08%.

VII. Metric function impact on the algorithm performance

It is an obvious next step to check alternative $l^p$-norms, $p \in \mathbb{N}$ and compare corresponding Coverage Rate to basic $l^2$-norm-based MODRA algorithm performance.

In our experiments, we additionally evaluate MODRA performance for the following metrics: $l^1$ (delay + loss), $l^3$, $l^5$, $l^{10}$ and $l^\infty$ (max(delay, loss)). Some authors, for example [19], suggest that there is an interconnection
between the metric being used and the ability of an algorithm to identify feasible routes w.r.t. multiple constraints. Based on the motivation given in [19], the authors even propose that $l_\infty$ potentially is the optimal choice. We verify this hypothesis for MODRA Coverage Rate, though it is important to say, that $l_\infty$-norm does not guarantee loop-free routing.

Our experiments show, that indeed increasing $p$ in $l^p$-norm being used to measure a path length in MODRA improves the performance of the algorithm and increases the coverage rate. At the same time, though for limited $p$ values $l^{10}$ shows the best performance out of all evaluated metrics, $l_\infty$ does not outperform $l^{10}$ and in some cases even completely lose to it. For example, figures 18 and 19 show the performance of MODRA with different metrics in Grid and Dual-Homed networks respectively.

The horizontal red line represents the baseline — MODRA with $l^2$-norm as a routing path length. The performance of MODRA with other metrics is shown as the difference from the baseline. It can be clearly seen, that $l^2$ outperforms $l^1$, while each $l^p$-norm, $p > 2$ performs better with increasing $p$. At the same time, $l_\infty$ loses to $l^{10}$, sometimes significantly.

A different picture can be observed in Full-Mesh and Mouth-Like networks (figures 20 and 21 respectively).

In Full-Mesh network MODRA with $l_\infty$-norm outperforms all other metrics, including $l^{10}$. Though, in general, all the metrics show very similar performance in terms of Coverage Rate.

And in Mouth-Like network MODRA with $l_\infty$-norm performance is almost the best, loosing to $l^{10}$ for only 2 constraint rate values 0.6 and 0.55.

Thus, MODRA can be implemented using different $l^p$-norms with finite $p \in \mathbb{N}$ as a routing path length. This can increase the Coverage Rate of the algorithm, while preserving all the properties of MODRA, proven in Section IV.

**VIII. MODRA ROBUSTNESS**

As we have already shown in Section II, without Source-dependent information it is impossible to design a fully-distributed Multi-Objective Destination-based routing algorithm for an arbitrary network.

A simple example of a network, where such routing is impossible is shown in Fig. 2. We have already shown in Section VI that in realistic network topologies MODRA
identifies the majority of theoretically feasible paths subject to multiple constraints. In this section we analyze MODRA robustness — the ability to satisfy multiple constraints for an identified feasible path in the process of distributed hop-by-hop routing.

Fig. 22 shows MODRA robustness, when using $l^2$-norm. Though it is lower than the initial Coverage Rate, it is still above 95%. The picture also shows, that for some theoretically feasible paths, which are not identified by the algorithm, in the process of distributed hop-by-hop routing the constraints are satisfied. That is why actual Distributed Routing Coverage Rate is slightly higher than Robustness.

Another important observation is that increasing $k$ in $l^k$-norm being used by MODRA, Robustness does not grow as much as initial Coverage Rate. In Fig. 23 MODRA Robustness is compared for the cases when $l^2$- and $l^{10}$-norms are used as a routing path length. It is higher for $l^{10}$-norm, but the difference from $l^2$-norm is at most 0.6% point.

Finally, MODRA Robustness w.r.t. initial Coverage Rate is shown in Fig. 24. It can be seen, that in each test topology Robustness is at least 99% w.r.t. initial Coverage Rate, and for Full-Mesh network it is 100%.

IX. CONCLUSION

Identifying routes through a network subject to several constraints is an well-known NP-hard problem, even in the case when the number of constraints is limited to 2. This problem has been addressed by various researchers in the last 20 years. Some of the proposed ideas can be found in [6], [7], [8], [9], [10], [11]. Though, a variety of approaches has been suggested, from the point of view of distributed routing in IP Networks there is still no state-of-the-art algorithm, like Dijkstra for the single constraint case. The reason is that there are several requirements for such solution, like being loop-free in distributed mode, computing one-to-all paths (instead of one-to-one, as in the majority of multi-objective routing algorithms), providing guarantees to satisfy multiple QoS-constraints in the process of distributed (hop-by-hop) routing.

Several attempts to modify single-constraint Dijkstra algorithm and apply it to multicriteria shortest path problem have been made since 1980-s [13], [12], [14], [15], [16]. Still, though the proposed multi-objective versions of Dijkstra algorithm can be considered as novel and efficient, most of them (if not all) are not suitable for distributed multi-objective routing in IP Networks.

In our paper we propose a new Dijkstra-like algorithm — MODRA — for multi-objective distributed routing, which is based on the single dimension path conversion principle and constructs a Shortest Path Tree w.r.t. the given single dimension path conversion metric. Then, we theoretically prove, that MODRA is loop-free in distributed mode, computes exactly the shortest paths in terms of $l^p$-norm, $p \in \mathbb{N}$ being used as a path length, thus for each Source-Destination pair computing a single Pareto-optimal path.

MODRA evaluation in several realistic network topologies with multiple sets of constraints shows its’ ability to identify the majority of all theoretically feasible routes in the network (no less than 94% in Grid, Mouth-Like and Dual-Homed networks). This value, called Coverage Rate, grows with increasing $p$ in $l^p$-norm being used by MODRA. Finally, we show that MODRA robustness — the ability to
satisfy multiple constraints for an identified feasible path in the process of distributed hop-by-hop routing — is at least 99% in all the considered test-cases.

APPENDIX

ALL-PAIR MULTI-OBJECTIVE ROUTING ALGORITHM (APMORA)

A. All-Pair Multi-Objective Routing Algorithm

Using MODRA critical idea, an all-Pair Multi-Objective Routing Algorithm (APMORA) can be designed, based on the modification of Floyd-Warshall algorithm (see Algorithm 2).

As APMORA computes all-to-all paths, it can be used as a solution for centralized routing with guaranteed performance subject to multiple QoS-constraints.

B. APMORA Performance and Comparison to MODRA

In terms of Coverage Rate, APMORA performance is similar to MODRA. We show it as the difference between APMORA and MODRA Coverage Rates (see Fig. 25).

The difference in Coverage Rate is no higher than 0.3% point. In general, APMORA performs slightly better than MODRA, though in some topologies, for example Mouth-Like, MODRA performs better.

ACKNOWLEDGMENT

The authors would like to thank all the team members of Moscow Network Algorithm Laboratory for their constant support in this research.

REFERENCES


