Finding K dissimilar paths: single-commodity and discretized flow formulations*

Ali Moghanni† Marta Pascoal‡§ Maria Teresa Godinho¶

January 28, 2025

Abstract

While finding a path between two nodes is the basis for several applications, the need for alternative paths also may have various motivations. For instance, it can be of interest to ensure reliability in a telecommunications network, to reduce the consequences of possible accidents in the transportation of hazardous materials or to decrease the risk of robberies in money distribution. All these applications involve the search for a set of paths that are as dissimilar as possible, with respect to the arcs in them. In this work, we present linear integer programming formulations that intend to find K dissimilar paths. The new models derive from a straightforward single-commodity flow formulation of the K paths problem, to which extra constraints are added to account for the concept of dissimilarity. In particular, in each formulation a different measure of the similarity between the K paths is minimized: the number of repeated arcs, the number of arc reuses and the number of pairwise arc overlaps. The third measure originates a non-linear model that is linearized through a discretized model reformulation technique. These formulations are complemented by a polynomial algorithm that allows to extract K loopless paths from the solutions output by each of them.

The formulations are tested for random and grid networks with respect to the solutions' average and minimum dissimilarity and the run time. Two of them stand out in terms of both those measures, when compared to an iterative approach from the literature. The best one regarding the average dissimilarity finds 20 paths with average dissimilarity of 99.5% in about 168 seconds for random networks of 5 000 nodes and 100 000 arcs. The same approach finds 20 paths for grid instances of size 100×100 in less than 18 seconds. The other formulation requires about 11 seconds for the same random instances, with a 0.7% deviation from the best known average dissimilarity; while it requires less than 7 seconds to compute 20 paths in 100×100 grids with a 0.6% deviation to the optimal dissimilarity.

*Published as: A. Moghanni, E. Pascoal, M. T. Godinho,"Finding K dissimilar paths: single-commodity and discretized flow formulations", *Computers & Operations Research*, 147:105939, 2022; DOI: 10.1016/j.cor.-2022.105939

© 2022. This version is made available under the CC-BY-NC-ND 4.0 license https://creativecommons.org/licenses/by-nc-nd/4.0/

†University of Coimbra, CMUC, Department of Mathematics, 3001-501 Coimbra, Portugal, (a.moghanni@hotmail.com), WWW home page: https://sites.google.com/view/alimoghanni/

‡University of Coimbra, CMUC, Department of Mathematics, 3001-501 Coimbra, Institute for Systems Engineering and Computers – Coimbra, rua Sílvio Lima, Pólo II, 3030-290 Coimbra, Portugal, Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano, Piazza Leonardo da Vinci, 32, Milan, 20133, Italy (marta@mat.uc.pt)

§Corresponding author

¶Department of Mathematics and Physical Sciences, Polytechnic Institute of Beja, Campus do Instituto Politécnico de Beja, rua Pedro Soares, 7800-295 Beja, Portugal, and CMAFcIO, Faculdade de Ciências da Universidade de Lisboa, Campo Grande, 1749-016 Lisboa (mtgodinho@ipbeja.pt)

Keywords: K alternative paths, Dissimilarity, Integer linear programming formulations, Discretized formulations

1 Introduction

Finding a path in a network between a source node and a terminal node is a classical network optimization problem and the basis for several applications in operations research. The determination of alternative dissimilar paths between a pair of nodes, on the other hand, is a less studied but interesting extension of the former problem. Figure 1 illustrates the concept of dissimilarity between paths. Here, the circles represent the network nodes, with s and t as the source and the terminal nodes, respectively, and each connection between two circles represents an arc linking them. Suppose that a set of $K \geq 2$ alternative paths between s and t needs to be found in this network. For $K = 2$, the pair $\{p_3, p_4\}$ is the best solution, given that the two paths are disjoint. However, if there are no other connections in the network, no sets of $K = 3$ disjoint paths from s to t exist. So, although disjointness is ideal when looking for dissimilarity, it may be impossible to achieve in a particular network and we may have to resort to dissimilarity as its relaxation. For example, the paths $\{p_1, p_3, p_4\}$, are clearly more diverse than the paths $\{p_1, p_2, p_4\}$, even if none of these sets is disjoint.

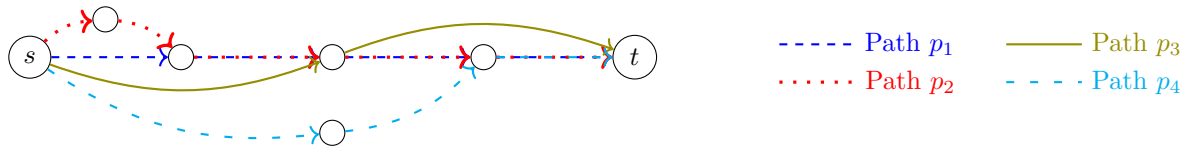


Figure 1: Graph example

The characteristic of path diversity is relevant for security, reliability or resilience reasons and has been used in a number of applications that range from hazardous materials routing to money collection or telecommunications (Constantino et al., 2017; Dell'Olmo et al., 2005; Martí et al., 2009; Mohri et al., 2022), to name a few, as will be described in more detail in Section 2. However, the problem of finding a set of dissimilar paths has been handled mostly in concrete applications under bi-objective frameworks with the goals of optimizing both the paths length/cost and the dissimilarity of the set of paths, seldom being studied by itself. In many cases, these approaches search for alternative paths in a sequential manner, often leading to solutions that disregard the interdependence of the different paths in the solution. Others rank a pre-established number of paths with the purpose of later selecting K alternative paths in a way intrinsically linked to the application, which makes the search depend on an estimated number of paths to compute and makes the paths output unfitted to other purposes. The methodological research on dissimilarity itself is, to our knowledge, scarce. Thus, the main contribution of this work is to study dissimilarity from an Integer Linear Programming point of view, which allows to look for solutions that privilege the dissimilarity of the K paths, taking into account the interaction between them. Three Integer Linear Programming models are proposed, which are simple to implement and provide sets of K dissimilar paths. The new models are studied with

respect to the consistency of their performance in two types of networks and to their resilience to changes in the size of the network and in the value of K.

Many variations on the concept of dissimilarity are found in the literature. Nevertheless, in general, this concept is associated with avoiding superpositions among the paths. We focus on arc superposition and test the adequacy of minimizing:

1. the number of repeated arcs;
2. the number of arc repetitions, that is, the number of times that an arc is reused;
3. the number of arc overlaps for all pairs of paths.

The third approach originates from a non-linear metric, which is linearized using a discretized model reformulation technique proposed in Gouveia (1995). The formulations are complemented by a polynomial algorithm that allows to extract K loopless paths from the solutions output by each of them. The dissimilarities of the sets of paths obtained by each of these approaches are compared by means of a non-linear metric proposed by (Erkut and Verter, 1998). Computational tests were performed in order to compare the quality of the formulations, both in terms of the values of this dissimilarity measure applied to the set of K paths produced and of the time required for solving each model. The results show that the third approach produces the best solutions, but that it is rather time-consuming when compared to the others. On the other hand, the second approach seems to produce fairly good results in reasonable run times.

The remainder of this work is structured as follows. Section 2 gives a literature overview of problems related to finding alternative paths. Section 3 serves as a preliminary section to the next three. Here the general problem of finding K dissimilar paths between two nodes is presented, a method for extracting K loopless paths from the solution for that problem is introduced, and the test setting to be used in the rest of the text is described. In the following three sections, ILP formulations are introduced to provide solutions based on the approaches described above. A brief computational analysis is presented for each formulation at the end of each section. Finally, in Section 7 a deeper computational comparison of the two best formulations is presented, together with a comparison with a classical approach from the alternative paths' literature, the iterative penalty method (IPM) introduced by Johnson et al. (1992). The formulations are compared in terms of the run time and of the dissimilarity of the output solutions. Conclusions and future directions for research are outlined in Section 8.

2 Related literature

Several network optimization problems search for a path between two nodes according to a certain criterion. In many cases, it is of interest to extend this problem by searching not only for the best solution, but also for the second best, the third best, and so on, that is, to rank paths by order of the objective function values. In practice, this is useful, for instance, when the paths must satisfy additional constraints, which can be verified as new solutions are found. Perhaps the best known of this family of problems is the K shortest paths problem, first proposed by

Hoffman and Pavley (1959). The K shortest paths problem has been solved efficiently by several authors. See for example (Eppstein, 1998; Martins, 1984; Martins et al., 1999) or (Katoh et al., 1982; Martins and Pascoal, 2003; Yen, 1971).

Being guided by the objective function values, the ranked paths often have similar compositions. Instead, in the K disjoint paths problem, the cost of K paths is minimized, while the overlaps between them are forbidden. The disjointness of the paths is often a requirement in telecommunications, in order to ensure the reliability of communications. In practice, this is managed by the computation of a pair of paths connecting two nodes, a primary (or active) path to use as a first option and a protection (or backup) path to replace the first one if there is a failure along its arcs or nodes. The determination of K disjoint simple paths has been studied by (Bhandari, 1998; Suurballe and Tarjan, 1984). Their approaches consist of formulating the problem as a minimum cost flow problem and applying a labeling algorithm, alternated with the update of the arc flows. The arc disjoint version of the problem has been studied in (Frank, 1985; Guruswami et al., 2003). A review on this topic can be found in (Iqbal and Kuipers, 2015).

A limitation of the K disjoint paths problem is that the disjointness constraint may lead to unfeasible instances for which no solution exists (as illustrated in Section 1). In this case, it would be desirable to find paths that are as disjoint as possible. The dissimilar paths problem searches for diverse paths and has been studied in the context of hazardous materials transportation, where the various alternatives should share few arcs and be relatively short. Different methods have been proposed for approaching this problem. The IPM (Johnson et al., 1992) is one of the most intuitive of these methods, based on the iterative application of a shortest path algorithm. At each iteration, a cost penalty is associated with each selected arc to discourage them from appearing in the forthcoming iteration; hence, generating dissimilar paths. Another proposed method is based on the Gateway Shortest Path problem, (Lombard and Church, 1993). In this case, the generated shortest paths should go through a given set of nodes called a "gateway" or corridor. It is intended that these corridors have different spacial locations. These methods are simple to implement and perform a sequential selection of paths. So, in general, they are very fast, but the selection of each path is affected by the previous choices.

The minimax method, by Kuby et al. (1997), selects paths using dissimilarity indices starting from K assigned paths. Akgün et al. (2000) suggest that when computing the similarity between two paths, the area of the intersection of a "buffer zone" defined for each path should be considered and makes use of a p-dispersion location model. Dell'Olmo et al. (2005)'s approach combines these two ideas. They start with a set of Pareto-optimal paths. Then, for each one, a "buffer zone" approximating the impact area of a material being released after an accident is constructed and a dissimilarity index is derived for every pair of paths, to find the most spatially different routes. These methods work in two phases: first, a number of paths are generated; second, a set of K paths is selected out of the obtained list considering the dissimilarity values. Because the calculation phase is done a priori, this type of approaches can also be efficient in practice. However, the quality of the final set of paths in terms of dissimilarity depends on the number of paths generated initially. If this number is too small, the dissimilarity may be compromised, if it is too big, the first phase may be very demanding.

Martí et al. (2009) considers the simultaneous minimization of the paths cost and maximization of their dissimilarity, while following a different spatial point of view regarding the dissimilarity index. They propose a GRASP metaheuristic to solve that problem. More recently, Zajac (2018) works on a related problem but considering vehicle routing instead of K paths. The work considers two objectives and routes for capacitated vehicles while exploring two dissimilarity indices: the "grid metric", modeling spatial dissimilarity, and the "edge metric", based on the arcs shared by different routes. Jabbarzadeh et al. (2020) extended the studies on hazardous materials shipment to the case of railway shipments in the presence of random disruption. In particular, it proposes a bi-objective two-stage stochastic program to study the impact of random disruption on rail hazmat shipments as well as a risk measure. Bi-objective ϵ-constraint algorithms for ILP formulations to optimize the cost and the dissimilarity of K paths are proposed in (Moghanni et al., 2021). This work is related to the present manuscript, but considers the optimization of two objective functions and uses formulations that are more demanding to solve than the single commodity flow formulations now introduced. Both Erkut et al. (2007); Batta and Kwon (2013), as well as the very recent work Mohri et al. (2022), present surveys on this topic.

For completeness, we mention also other application problems that involve the dissimilarity between paths. Another class of applications for this problem is related to the unpredictability of the routes, in order to prevent robberies, when transporting cash, or to ensure efficient street patrols. In this case, besides spatial dissimilarity, time dissimilarity is also relevant. Calvo and Cordone (2003) addresses a real-case submitted by a security company operating in Milan, Italy. The response to alerts must be secured using as few guards as possible in a way that each guard desirably performs a different route every night. The authors propose an ILP model as well as a heuristic to approach the problem. Constantino et al. (2017) study a dissimilar arc routing problem proposed by a company that needs to collect safes from parking meters located along the streets, minimizing the total collecting time and avoiding similar tours, in terms of space and time. Also in this case the problem is addressed through an ILP model and a heuristic, the total collecting time being minimized and subject to constraints to avoid similar tours. Talarico et al. (2015) study a K dissimilar vehicle routing problem, with the aim of generating a set of K alternative solutions of a single vehicle routing problem instance, in such a way that each alternative solution differs from the others by at least a given threshold. Tikani et al. (2021) extend such applications with a time-dependent dimension, while still taking into account the risk of robberies and the effects of traffic congestion. They propose an evolutionary algorithm, which includes a caching mechanism and fuzzy logic to dynamically adjust the rates of operators during the search process.

As mentioned earlier, in some problems in telecommunications, pairs of paths are sought to have an alternative backup path in case a failure occurs along the active path (Hu, 2003; Gomes et al., 2020). A more flexible approach to the problem imposes only that the overlap between them does not exceed a certain value. Another telecommunications problem that limits the resources shared by paths consists in minimizing the number of shared risk link groups in pairs of paths. In this case, each arc in the network is associated with a given set of risk groups

and the goal is to find two paths that have as few groups in common as possible (Gomes et al., 2016; Pascoal and Clímaco, 2020). Related problems are surveyed in the recent work (Clímaco and Craveirinha, 2019).

3 The K dissimilar paths problem

3.1 Problem definition

Let (N, A) be a directed graph with $|N| = n$ nodes and $|A| = m$ arcs, where s denotes a given source node and t denotes a given target node, $s, t \in N$. Let P denote the set of paths from node s to node t in (N, A) and $K \geq 2$ be a given integer. The problem of finding K paths in P can be modelled using the usual single commodity flow variables, such that f_{ij} accounts for the amount of flow (or the number of paths) traversing arc $(i, j) \in A$, subject to the set of constraints:

$$\sum_{j \in N:(i,j) \in A} f_{ij} - \sum_{j \in N:(j,i) \in A} f_{ji} = \begin{cases} K, & i = s \\ 0, & i \neq s, t \\ -K, & i = t \end{cases} \tag{1a}$$

$$f_{ij} \in \mathbb{Z}_0^+, \quad (i, j) \in A \tag{1b}$$

The first set of constraints, (1a), ensures the determination of K paths from s to t, whereas the second, (1b), defines the domain of the variables f.

In the K dissimilar paths problem from s to t, it is intended that the the K paths are spread in the network. For instance, in Figure 1 paths p_1 and p_3 are more dissimilar than paths p_1 and p_2. The limit situations are that the paths fully coincide, corresponding to the minimum dissimilarity, or that the paths are disjoint, corresponding to the maximum dissimilarity. In other words, the K paths should have as few superpositions between them as possible. This additional requirement can be modeled by considering a metric, M, which measures the superpositions between the paths, and minimizing its value:

$$\begin{aligned} \min \quad & M & (2a) \\ \text{subject to} \quad & (1a), (1b) & \\ & M = \varphi(f) & (2b) \end{aligned}$$

where φ stands for a function depending on the vector of variables f.

Several works have proposed dissimilarity indices as a way to evaluate paths diversity. Erkut and Verter (1998) described four similarity indices and showed that there is a strong correlation between them. We will use the following:

$$S(p, q) = \frac{1}{2} \left(\frac{L(p \cap q)}{L(p)} + \frac{L(p \cap q)}{L(q)} \right) \tag{3}$$

of those indices, where p and q are two paths and $L(p)$ denotes the number of arcs in the sequence p. The dissimilarity between p and q is then given by $D(p, q) = 1 - S(p, q)$. The dissimilarity values vary between 0 and 1, the first when the paths coincide and the latter when they are

arc disjoint. The dissimilarity of a set of $K \in \mathbb{N}$ paths, denoted by D_K, is the average of the pairwise dissimilarities of the paths in the set.

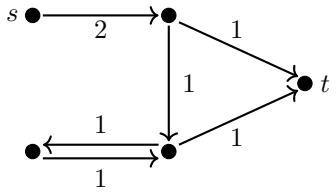
The similarity index S, associated to D, is difficult to handle in an ILP context. To overcome this barrier, we devise a strategy that consists in considering three metrics (M_1, M_2 and M_3) that emulate S with growing proximity. This framework will allow us to understand to what extent it is possible to obtain solutions with good dissimilarities, whilst obviating the complexity of S, and quantifying the computational effort required to achieve that goal. This will be done in the following sections, where three formulations exploring different ways of expressing the metric (2b) are introduced. In each section, a preliminary empirical study of the behaviour of the new formulation in terms of the dissimilarity of its solutions, run times and integer programming gaps is included. It is worth noting that knowledge of the individual paths is required to calculate S and that this is not provided directly by the variables f. A method for retrieving an optimal loopless solution from any optimal solution of problem (2) is described next. Afterwards, the instances used to carry out the computational tests are also described.

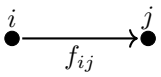
3.2 Algorithm for retrieving K loopless paths

The variables f in formulation (1) do not distinguish the arcs that belong to each path. Additionally, the formulation may admit optimal solutions that contain subtours, as long as they do not include any arc that is common to several paths. A set of K loopless paths can be retrieved by the algorithm outlined below.

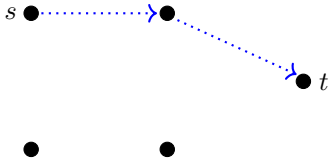
Consider the network in Figure 2a, depicting the arcs of a feasible solution of formulation (2) when finding $K = 2$ paths between nodes s and t. One of the arcs appears simultaneously in the two paths (the one emerging from node s). Besides, this solution contains subtours. A loopless solution can be retrieved from this one by alternating the shortest paths computation with respect to the length and updating the network to reflect the removal of the path that was found. Figure 2b shows the first path to consider within the given solution, p_1. The network in Figure 2c results from limiting the utilization of the arcs of that path to reflect that they have been used once. This is done by reducing by 1 the value associated with the arcs of that path. Afterwards, these two steps are repeated to find the second path in the solution, p_2, as shown in Figures 2d and 2e. Exactly K repetitions of these steps are required to fully retrieve a loopless optimal solution out of vector f, and there is no path that links nodes s and t in the network after these K iterations. The method described to obtain K loopless paths when given any optimal solution f for formulation (2) is outlined in Algorithm 1.

In lines 1 and 2 of Algorithm 1 the network corresponding to the arcs in the given solution f is built, (N', A'). Each arc $(i, j) \in A'$ is associated to the value of f_{ij}. This value works like the arc (i, j)'s capacity. Then, in each iteration of the loop in lines 3 to 7, one path is determined and the capacities of its arcs are updated. The outer loop runs exactly K times, so that K paths are found. Moreover, the paths determined on line 4 are always loopless, given that each of them minimizes the number of arcs. These paths can be found by means of breadth-first search, and therefore the algorithm runs in $O(Km)$ time. Also, the empirical tests reported later showed that its run time is very small when compared to solving any of the formulations here presented.

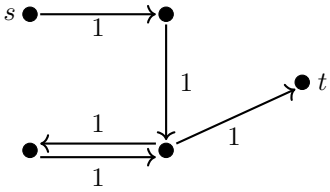




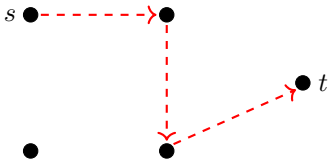
(a) Network (N', A')



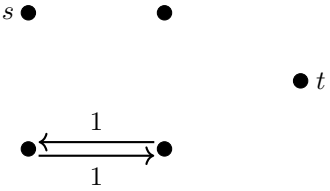
(b) Path from s to t of minimum length in (N', A'), p_1



(c) Network (N', A') after decreasing f_{ij} along p_1 and removing the arcs with $f_{ij} = 0$.



(d) Path from s to t of minimum length in (N', A'), p_2



(e) Network (N', A') after decreasing f_{ij} along p_2 and removing the arcs with $f_{ij} = 0$.

Figure 2: Application of Algorithm 1 to a feasible solution of formulation (2) for $K = 2$

A proof of the correction of Algorithm 1 can be found in Appendix A.

3.3 Tests setting

The formulations presented in the following sections are analyzed empirically with the goal of studying their behavior in terms of the dissimilarity of the solutions that each one produces, their run times and integer programming gaps. The setting for those experiments is now described.

The experiments were carried out on a 64-bit PC with an Intel®Core™ i9-10900K Quad core at 3.7GHz with 128GB of RAM.

Two types of directed networks were considered when computing $K = 3, 4, \ldots, 10$ paths:

- Random networks, denoted by $R_{n,m}$, with $n = 100, 300, 500$ nodes, and $m = dn$ arcs, for average degrees $d = 5, 10$. A Hamiltonian cycle is created for all the nodes in the network, and afterwards the remaining arcs are generated randomly. This Hamiltonian cycle is directed, therefore only the weak connectivity of the graph is ensured and some unfeasible instances may occur. There are no parallel arcs between pairs of nodes. For each size of these networks, 30 instances were generated for different seeds. The reported results are based only on the 22, out of the 30, which originated feasible problems.

Algorithm 1: Algorithm for retrieving K loopless paths from a given solution f

$N' \leftarrow \{i \in N : f_{ij} \geq 1 \text{ for some } j \in N\} \cup \{t\}$
$A' \leftarrow \{(i,j) \in A : f_{ij} \geq 1\}$
for $k = 1, \ldots, K$ **do**
    $p_k \leftarrow$ shortest path from s to t in terms of the number of arcs in network (N', A')
    for $(i,j) \in p_k$ **do**
        $f_{ij} \leftarrow f_{ij} - 1$
        if $f_{ij} = 0$ **then** Delete arc (i,j) from A'

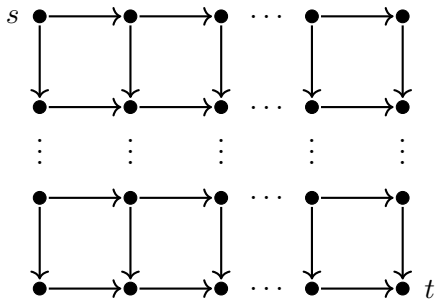


Figure 3: Grid network

- Grid networks, denoted by $G_{h,w}$, with h rows and w columns, arranged in a planar grid as shown in Figure 3. Vertices are numbered consecutively from left to right and from top to bottom, and any pair of adjacent nodes is connected by an arc. The characteristics of this set of tests are summarized in Table 1. Because these grids have a fixed structure, only one instance was considered for each size.

In both cases the source and the target nodes are $s = 1$ and $t = n$, respectively, with no loss of generality.

Table 1: Grid instances characteristics

$G_{h,w}$	$G_{3,12}$	$G_{6,6}$	$G_{4,36}$	$G_{12,12}$
n	36	36	144	144
m	57	60	248	264

We begin by observing that in a non-negligible number of the random instances, K disjoint paths can be found, specially when K is small. This happens for 274 of the 1056 instances. Unless otherwise stated, hereafter only the instances with no disjoint solutions are considered, the reason being that the main goal of this work is to find good methods to obtain dissimilar paths in networks where finding those paths is not easy. The drawback of this decision is that each set $R_{n,m}$ becomes smaller. Table 2 indicates the final number for each type of instances. It should be remarked that there are no disjoint solutions in the grid instances, given that $K \geq 3$.

Table 2: Final number of random network instances

	K							
$R_{n,m}$	3	4	5	6	7	8	9	10
$R_{100,500}$	10	20	21	22	22	22	22	22
$R_{100,1000}$	–	4	8	10	15	18	20	21
$R_{300,1500}$	8	15	21	22	22	22	22	22
$R_{300,3000}$	1	4	9	13	15	18	20	21
$R_{500,2500}$	8	18	19	21	21	22	22	22
$R_{500,5000}$	5	8	10	14	17	19	22	22

4 Minimization of the number of repeated arcs

4.1 Formulation

In the following, a modeling approach to problem (2) is introduced, exploring the idea that dissimilar paths have few arcs that are shared, and thus minimizing the number of repeated arcs that appear in those paths (designated `MRA`). A given arc $(i, j) \in A$ is said to be repeated in a set of K paths if it belongs to more than one of them.

Let y_{ij} be a variable equal to 1 if and only if the arc (i, j) is repeated, or 0 otherwise, for any $(i, j) \in A$. We use the y variables to define the first metric, M_1, indicating the number of repeated arcs in the K paths, as follows:

$$M_1 = \sum_{(i,j)\in A} y_{ij}. \tag{4}$$

The next formulation fits the schematic model presented in the previous section. Its purpose is to find a set of K paths which minimize M_1:

$$\begin{aligned}
\min \quad & M_1 && \text{(5a)}\\
\text{subject to} \quad & (1a), (1b), (4) &&\\
& y_{ij} \le f_{ij},\ (i,j) \in A && \text{(5b)}\\
& f_{ij} - 1 \le (K-1) y_{ij},\ (i,j) \in A && \text{(5c)}\\
& y_{ij} \in \{0,1\},\ (i,j) \in A && \text{(5d)}
\end{aligned}$$

This formulation has $O(m)$ variables and $O(n + m)$ constraints. The constraints (5b) and (5c) relate the f and the y variables, and, together with (5a), ensure that y_{ij} is 1 if and only if the arc (i, j) is used in more than one path, that is, if arc (i, j) is repeated, and 0 otherwise. In fact, given the arc $(i, j) \in A$:

1. If $f_{ij} = 0$, then by (5b) we have $y_{ij} = 0$.
2. If $f_{ij} = 1$, then neither (5b) nor (5c) have implications on the value of y_{ij}, that is, the value of y_{ij} can be arbitrary. However, (5a) minimizes the sum of all these variables and this minimization is achieved if the arbitrary y_{ij}'s are equal to 0, as intended.

3. If $f_{ij} > 1$, then by (5c) we have $y_{ij} = 1$.

It is worth noting that the reasoning used in point 2., also holds to show that the set of constraints (5b) can be dropped from formulation (5).

4.2 Preliminary computational experiments

In the following, formulation (5) is analyzed empirically. The corresponding code (`MRA`) was implemented in C. This code uses IBM ILOG CPLEX version 20.1 as the ILP solver. Algorithm 1, also coded in C, was applied to the results of this implementation, in order to remove possible loops from the obtained solutions. The testing conditions were those described in Section 3.3.

The considered grid networks are acyclic, therefore Algorithm 1 was not applied for such instances. Moreover, the run times of Algorithm 1 in random networks were small (at most 90 milliseconds), therefore they are not included in the results reported in the following.

Besides formulation (5), also its variant obtained by removing the constraints (5b) was implemented. Both models are valid formulations of the problem. The differences in the run times obtained with both variants were small and the original version presented better dissimilarity values, therefore, only the original one is considered below.

An upper bound of 300 seconds was set for the elapsed time when running the solver. The results presented in the following for random networks are average values for solving the instances with the same characteristics except for the seed, and with no disjoint solutions. The results shown for grid networks are based on a single instance, again with no disjoint solutions.

Table 3: Average run times of `MRA` (seconds)

	K								
$R_{n,m}$	3	4	5	6	7	8	9	10	
$R_{100,500}$	0.017	0.016	0.031	0.054	0.073	0.080	0.094	0.111	
$R_{100,1000}$	–	0.031	0.053	0.095	0.062	0.085	0.099	0.125	
$R_{300,1500}$	0.047	0.059	0.056	0.080	0.118	0.130	0.184	0.225	
$R_{300,3000}$	0.108	0.106	0.113	0.122	0.132	0.142	0.146	0.153	
$R_{500,2500}$	0.081	0.090	0.094	0.097	0.103	0.118	0.141	0.162	
$R_{500,5000}$	0.156	0.160	0.163	0.169	0.176	0.174	0.192	0.200	
Average	0.082	0.077	0.085	0.103	0.111	0.122	0.143	0.163	0.110

	K								
$G_{h,w}$	3	4	5	6	7	8	9	10	
$G_{3,12}$	0.025	0.217	0.225	0.268	0.571	0.525	0.168	0.122	
$G_{4,36}$	0.020	0.074	1.149	1.624	5.383	10.742	11.120	12.534	
$G_{6,6}$	0.516	0.217	0.110	0.462	0.728	0.562	0.676	0.153	
$G_{12,12}$	0.018	0.061	0.509	0.411	2.495	8.903	22.049	300	
Average	0.144	0.142	0.498	0.691	2.294	5.183	8.503	78.202	11.957

Unlike for the remaining instances, the formulation `MAR` reached the time limit of 300 seconds

when finding the best set of $K = 10$ paths in the 12×12 node grid. This corresponds to solving 96.9 of the initial grid instances and all the random instances. Table 3 presents the run times required to solve the problem using `MAR`. The code `MRA` is fairly fast, especially for the random networks, for which all the instances were solved in average times that never exceed 0.3 seconds.

Next, we focus on the dissimilarity based on the metric D of the best integer solution produced by solving `MRA` within the time limit. We first calculate the dissimilarity obtained for each set of K paths, D_K. Afterwards, the averages of these values, $\overline{D}_K$, are calculated for each set of instances. The results are summarized in Table 4.

Table 4: $\overline{D}_K$ values of the solutions produced solving `MRA` to optimality

	K								
$R_{n,m}$	3	4	5	6	7	8	9	10	
$R_{100,500}$	0.896	0.910	0.900	0.881	0.856	0.831	0.804	0.780	
$R_{100,1000}$	–	0.964	0.964	0.952	0.940	0.923	0.927	0.861	
$R_{300,1500}$	0.954	0.956	0.949	0.937	0.929	0.923	0.906	0.897	
$R_{300,3000}$	0.960	0.973	0.965	0.887	0.888	0.891	0.894	0.874	
$R_{500,2500}$	0.966	0.968	0.952	0.953	0.943	0.940	0.919	0.895	
$R_{500,5000}$	0.919	0.937	0.940	0.947	0.951	0.949	0.943	0.931	
Average	0.939	0.951	0.945	0.926	0.918	0.910	0.899	0.873	0.920
	K								
$G_{h,w}$	3	4	5	6	7	8	9	10	
$G_{3,12}$	0.949	0.718	0.708	0.000	0.000	0.000	0.000	0.000	
$G_{4,36}$	0.982	0.982	0.771	0.277	0.687	0.000	0.000	0.000	
$G_{6,6}$	0.933	0.933	0.900	0.893	0.000	0.000	0.000	0.000	
$G_{12,12}$	0.970	0.970	0.955	0.952	0.939	0.935	0.924	0.919	
Average	0.958	0.900	0.833	0.530	0.406	0.233	0.231	0.229	0.540

An absolute assessment of the dissimilarity results would require the optimal value of D_K to be known for each instance, which is not. Therefore, the analysis will be based only on a relative overall comparison with other approaches presented in the following, recalling that the values of D_K vary between 0 and 1.

The $\overline{D}_K$ of the solutions produced by the model `MRA` in the random instances are far better than those produced for the grid instances. In addition, several of these $\overline{D}_K$ values are 0 when $K \geq 6$, depending on the size of the grid. This means that all paths in the solution coincide. Thus, although fast, this does not seem to be a very good option to consider in order to obtain K dissimilar paths because if an arc is repeated, the present model does not take into account its number of repetitions. The approach introduced in the next section tackles that situation.

Finally, Tables 5 and 6 present the average of the best known objective function values and the average integer programming gaps determined by the lower bound produced by the linear relaxation of formulation (5). These gaps are computed as $100(M_1^* - M_{LR_1}^*)/|M_1^*|$ %, where M_1^* denotes the optimum value of (5) and $M_{LR_1}^*$ denotes the optimum value of its linear

Table 5: Average best know integer objective function values of MRA (M_1)

	K							
$R_{n,m}$	3	4	5	6	7	8	9	10
$R_{100,500}$	1.1	1.2	1.2	1.5	1.6	2.0	2.3	2.5
$R_{100,1000}$	–	1.0	1.0	1.0	1.0	1.0	1.0	1.0
$R_{300,1500}$	1.0	1.1	1.1	1.1	1.3	1.4	1.8	2.0
$R_{300,3000}$	1.0	1.0	1.0	1.0	1.1	1.1	1.1	1.1
$R_{500,2500}$	1.1	1.1	1.1	1.1	1.2	1.4	1.7	1.9
$R_{500,5000}$	1.0	1.0	1.0	1.1	1.1	1.1	1.1	1.1

	K							
$G_{h,w}$	3	4	5	6	7	8	9	10
$G_{3,12}$	2.0	11.0	12.0	13.0	13.0	13.0	13.0	13.0
$G_{4,36}$	2.0	4.0	35.0	36.0	37.0	38.0	38.0	38.0
$G_{6,6}$	2.0	4.0	6.0	8.0	10.0	10.0	10.0	10.0
$G_{12,12}$	2.0	4.0	6.0	8.0	10.0	12.0	14.0	16.0

programming relaxation. Whenever the value M_1^* is unknown, the best known integer value of the objective function is used to compute these gaps. In the random instances, the number of repeated arcs is rather small and, in most cases, only one arc is repeated in the solutions, even when several paths are found. The same values are bigger for the grid instances, due to their topology, and this may be one of the reasons for the difficulty in solving this set of instances. As for the gap values, in the random instances, they are at most 66%, and are even bigger for the grid networks, between 32% and 72%.

Table 6: Average integer programming gaps of MRA (%)

	K							
$R_{n,m}$	3	4	5	6	7	8	9	10
$R_{100,500}$	22	44	38	38	35	41	43	43
$R_{100,1000}$	–	66	62	56	58	56	53	51
$R_{300,1500}$	37	49	47	41	39	40	46	47
$R_{300,3000}$	50	58	61	58	56	54	53	49
$R_{500,2500}$	47	51	40	38	37	41	45	45
$R_{500,5000}$	30	37	37	47	49	47	47	42

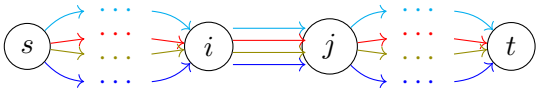
	K							
$G_{h,w}$	3	4	5	6	7	8	9	10
$G_{3,12}$	50	60	50	46	41	37	34	32
$G_{4,36}$	50	66	72	58	50	45	41	37
$G_{6,6}$	50	66	66	70	70	65	60	55
$G_{12,12}$	50	66	66	70	70	71	71	72

5 Minimization of the number of arc reuses

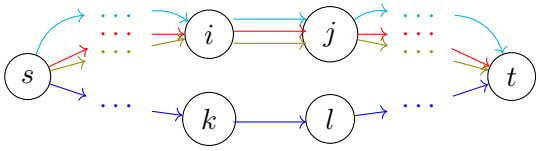
5.1 Formulation

The goal of formulation (5) is to minimize the number of arcs that are repeated in the solution. The undesired effect of this objective may be that few arcs appear in many different paths. An example is depicted by the two sets of $K = 4$ paths in Figure 4. Both the depicted solutions have a single repeated arc, the arc (i, j). However, the paths in the solution in Figure 4b are more dissimilar than the paths in Figure 4a, the reason being that in the first case the repeated arc, (i, j), appears in three paths, while it appears in all four paths in the latter case.

The previous example suggests that the way how paths spread in a network is affected not only by the number of repeated arcs but also by the number of times that each repeated arc is



(a) Solution with one repeated arc, shared by several paths



(b) Solution with one repeated arc, shared by fewer paths

Figure 4: Different sets of $K = 4$ paths

reused. This is taken into account in the following, when defining the second metric, M_2, which results in a formulation aiming at minimizing the number of arc reuses (designated `MAR`).

Let the number of reuses of a given arc $(i, j) \in A$ in a set of K paths be the number of paths it belongs to, excluding one of them. Now, consider variables u_{ij}, corresponding to the number of times that arc $(i, j) \in A$ is reused in different paths and binary variables w_{ij}, such that w_{ij} is equal to 1 if and only if the arc (i, j) appears in at least one path, or 0 otherwise, for any $(i, j) \in A$. Then, for each given arc $(i, j) \in A$, the new sets of variables are related to the variables f_{ij} as follows:

$$f_{ij} \leq K w_{ij}, \quad (i, j) \in A \tag{6a}$$

$$w_{ij} \leq f_{ij}, \quad (i, j) \in A \tag{6b}$$

$$u_{ij} = f_{ij} - w_{ij}, \quad (i, j) \in A \tag{6c}$$

Observe that the constraints (6a) and (6b) ensure that w_{ij} is equal to 1 if and only if (i, j) is used in some path, that is, if and only if $f_{ij} > 0$, for any $(i, j) \in A$. In fact:

- If $f_{ij} = 0$, then (6b) implies $w_{ij} = 0$.
- If $f_{ij} \geq 1$, then (6a) implies $w_{ij} = 1$.

In turn, constraints (6c) ensure that variables u_{ij} correspond to the number of times that arc $(i, j) \in A$ is reused in different paths:

- If $f_{ij} = 0$ and $w_{ij} = 0$, then (6c) implies $u_{ij} = 0$.
- If $f_{ij} = 1$ and $w_{ij} = 1$, then (6c) implies $u_{ij} = 0$.
- If $f_{ij} > 1$ and $w_{ij} = 1$, then (6c) implies $u_{ij} = f_{ij} - 1$.

Hence, the metric M_2, indicating the total number of reuses of the arcs in the K paths, can be defined as follows:

$$M_2 = \sum_{(i,j) \in A} u_{ij} \tag{7}$$

Thus, considering the constraints (6a) to (6c), together with (7) and

$$w_{ij} \in \{0, 1\}, \; u_{ij} \in \mathbb{Z}_0^+, \quad (i, j) \in A \tag{8}$$

we have a valid model for (2b). It is now possible to write an ILP model of the problem of minimizing the number of arc reuses:

$$\begin{aligned} \min \quad & M_2 && \text{(9a)} \\ \text{subject to} \quad & (1a), (1b), (6a) - (6c), (7), (8) \end{aligned}$$

This formulation has $O(m)$ variables and $O(m+n)$ constraints. Because both f_{ij} and w_{ij} are integer variables, the variables u_{ij} are implicitly defined as integers, for any $(i,j) \in A$. Observe also that constraints (6c), together with the non-negativity constraints of the variables u_{ij}, imply (6b). Therefore, the constraints (6b) can be skipped from the formulation.

The application of Algorithm 1 to any optimal solution of (9) still outputs K loopless and optimal paths with respect to the number of arc reuses. Computational results for an implementation of this formulation are shown next.

5.2 Preliminary computational experiments

The setup for the tests presented in the following for assessing formulation (9) is similar to what was described in Section 3.3. While a number of relaxations of this formulation were tested, the model obtained by skipping the constraints (6b) originated slightly more dissimilar solutions than the others. Thus, only the results for this model are presented. For simplicity, we refer to this new model as formulation (9) (or code `MAR`).

The code `MAR` was able to solve to optimality all the instances within the 300 seconds time limit, and was faster than the previous in almost all cases. Table 7 summarizes its average run times. The major differences are found in the results for the grid networks, where run times fall, on average, by 99.6%, and in the subset of the random networks previously identified as the harder to solve, which has reductions of 48% for the $R_{100,500}$ instances, 41% for the $R_{300,1500}$ instances and 25% for the $R_{300,1500}$ instances. Still, it is important to point out that `MAR` solved all instances, in less than 0.3 seconds.

Table 8 shows the values of $\overline{D}_K$ for the solutions produced by solving `MAR` to optimality. As expected, in general `MAR` produced solutions with higher dissimilarities, when compared to `MRA`. There are a few exceptions on random networks when finding 3 or 4 paths, where the number of arc reuses was particularly small, as shown in Table 9.

Table 9 presents the optimal objective values for formulation (9). As expected, these are always greater than or equal to those obtained with the model `MRA`, which confirms that in many cases the repeated arcs in the solutions of `MRA` appear more that once. Taking also into account the $\overline{D}_K$ values, we can conclude that the paths output by `MAR` are either more spread or are longer than those output by `MRA`. However, because all the solutions in grid networks have the same length, at least in those cases, the first conclusion must hold, thus showing the superiority of the current approach over the previous.

Finally, the value of the integer programming gap produced by the linear relaxation of (9) was 0 for all instances.

Table 7: Average run times of MAR (seconds)

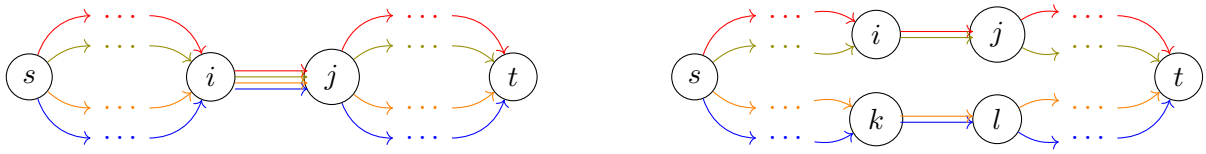
	K								
$R_{n,m}$	3	4	5	6	7	8	9	10	
$R_{100,500}$	0.020	0.020	0.016	0.018	0.018	0.019	0.018	0.017	
$R_{100,1000}$	–	0.029	0.029	0.030	0.032	0.030	0.031	0.030	
$R_{300,1500}$	0.050	0.058	0.062	0.063	0.063	0.064	0.067	0.067	
$R_{300,3000}$	0.094	0.096	0.111	0.100	0.104	0.107	0.108	0.114	
$R_{500,2500}$	0.090	0.114	0.105	0.117	0.128	0.108	0.112	0.117	
$R_{500,5000}$	0.167	0.205	0.211	0.219	0.214	0.221	0.233	0.232	
Average	0.084	0.087	0.089	0.091	0.093	0.092	0.095	0.096	0.090

	K								
$G_{h,w}$	3	4	5	6	7	8	9	10	
$G_{3,12}$	0.007	0.005	0.001	0.006	0.001	0.000	0.007	0.004	
$G_{4,36}$	0.012	0.014	0.017	0.009	0.016	0.013	0.017	0.016	
$G_{6,6}$	0.004	0.006	0.004	0.006	0.007	0.006	0.007	0.004	
$G_{12,12}$	0.012	0.017	0.018	0.044	0.018	0.022	0.011	0.017	
Average	0.009	0.010	0.010	0.016	0.010	0.010	0.010	0.010	0.010

6 Minimization of the number of arc overlaps for each pair of paths

6.1 Formulation

In this section, we explore a third approach to the K dissimilar paths problem, by defining a metric M_3 that accounts for the total number of arc overlaps between pairs of paths. The corresponding formulation is designated by MAO.



(a) Solution with one repeated arc, shared by several paths

(b) Solution with many repeated arcs, shared by few paths

Figure 5: Different sets of $K = 4$ paths

Given two paths $p, q \in P$, it is said that there is an overlap between them if there is an arc $(i, j) \in A$ that belongs to both p and q. Thus, the number of overlaps between those paths is the number of arcs that appear in both, that is, $OL(p, q) = |\{(i, j) \in A : (i, j) \in p \wedge (i, j) \in q\}|$, which coincides with $L(p \cap q)$, a common term to the several similarity indices and in particular to the index S in (3). In this way, the number of overlaps in a set of K paths is the total number

Table 8: $\overline{D}_K$ values of the solutions produced solving MAR to optimality

	K								
$R_{n,m}$	3	4	5	6	7	8	9	10	
$R_{100,500}$	0.895	0.910	0.901	0.894	0.877	0.868	0.847	0.848	
$R_{100,1000}$	–	0.967	0.973	0.958	0.967	0.966	0.953	0.954	
$R_{300,1500}$	0.946	0.951	0.951	0.941	0.938	0.932	0.926	0.923	
$R_{300,3000}$	0.974	0.978	0.967	0.966	0.971	0.958	0.956	0.953	
$R_{500,2500}$	0.954	0.971	0.962	0.959	0.959	0.952	0.949	0.942	
$R_{500,5000}$	0.947	0.957	0.956	0.962	0.964	0.963	0.962	0.962	
Average	0.943	0.956	0.952	0.947	0.946	0.940	0.932	0.930	0.943

	K								
$G_{h,w}$	3	4	5	6	7	8	9	10	
$G_{3,12}$	0.949	0.808	0.754	0.764	0.700	0.695	0.684	0.631	
$G_{4,36}$	0.982	0.982	0.892	0.809	0.811	0.755	0.707	0.679	
$G_{6,6}$	0.933	0.933	0.900	0.893	0.867	0.857	0.833	0.822	
$G_{12,12}$	0.970	0.970	0.955	0.952	0.939	0.935	0.924	0.919	
Average	0.958	0.923	0.875	0.854	0.829	0.810	0.787	0.763	0.849

of overlaps for all pairs of paths, that is,

$$OL(p_1, p_2, \ldots, p_K) = \sum_{i=1}^{K-1} \sum_{j=i+1}^{K} OL(p_i, p_j) = \sum_{i=1}^{K-1} \sum_{j=i+1}^{K} L(p_i \cap p_j). \tag{10}$$

This expression defines a metric that is closer to S than both M_1 and M_2.

Figure 5 exhibits two sets of $K = 4$ paths which illustrate this concept. In Figure 5a, the four paths share arc (i, j), therefore, there are 6 overlaps in the depicted solution. Instead, in Figure 5b, both the arcs (i, j) and (k, l) appear in two paths. Thus, there are 2 overlaps.

Now, let OL_{ij} be the number of overlaps of all the pairs of the K paths that occur at a given repeated arc (i, j). Recall that the variables f_{ij} account for the number of paths that traverse

Table 9: Average optimal values of MAR (M_2^*)

	K							
$R_{n,m}$	3	4	5	6	7	8	9	10
$R_{100,500}$	1.7	2.0	3.1	4.5	6.0	8.0	10.0	12.0
$R_{100,1000}$	–	1.0	1.5	2.2	2.5	3.1	3.8	4.6
$R_{300,1500}$	1.2	1.8	2.4	3.4	4.6	5.8	7.3	9.0
$R_{300,3000}$	1.0	1.2	1.6	2.1	2.8	3.4	4.1	5.0
$R_{500,2500}$	1.2	1.6	2.6	3.4	4.5	5.5	6.8	8.3
$R_{500,5000}$	1.4	1.9	2.5	2.9	3.4	4.1	4.6	5.7

	K							
$G_{h,w}$	3	4	5	6	7	8	9	10
$G_{3,12}$	2.0	13.0	24.0	35.0	46.0	57.0	68.0	79.0
$G_{4,36}$	2.0	4.0	39.0	74.0	109.0	144.0	179.0	214.0
$G_{6,6}$	2.0	4.0	8.0	12.0	18.0	24.0	32.0	40.0
$G_{12,12}$	2.0	4.0	8.0	12.0	18.0	24.0	32.0	40.0

arc (i,j), then $OL_{ij} = \binom{f_{ij}}{2}$ and we can write

$$OL(p_1, p_2, \ldots, p_K) = \sum_{(i,j)\in A} OL_{ij} = \sum_{(i,j)\in A} \binom{f_{ij}}{2}. \tag{11}$$

This expression has m terms, given that each arc is considered only once. Contrarily, computing the number of overlaps using (10) involves a higher number of terms, $\binom{K}{2}m$, which likely makes the use of (11) less demanding than the use of (10). Moreover, using this reasoning, a third (non-linear) model can be obtained from the generic formulation, (2), if modelling (2b) as

$$M_3 = \sum_{(i,j)\in A} \binom{f_{ij}}{2} \tag{12}$$

and minimizing this expression.

As mentioned, expression (12) is not linear, however, a discretized model can be applied in this case. As pointed by Gouveia (1995), discretized models have been used to derive strong formulations for several network problems (see, for instance (Gouveia, 1998; Gouveia and Moura, 2012)), but have also been applied to tackle non-linear costs, as in the present case (Gouveia and Saldanha-da Gama, 2010; Correia et al., 2008, 2010). Therefore, one way of overcoming the difficulty of modeling the number of overlaps is to use the variables proposed in Gouveia (1995) in a different context.

Consider binary variables, g^a_{ij}, equal to 1 if arc (i,j) appears in exactly a paths, or 0 otherwise, for any $(i,j) \in A$ and $a = 1, \ldots, K$. The f_{ij} and the new g^a_{ij} variables are related as follows:

$$f_{ij} = \sum_{a=1}^{K} a\, g^a_{ij}, \quad (i,j) \in A. \tag{13}$$

By using this expression we obtain a discretized reformulation of the flow conservation constraints of the generic formulation, (1a):

$$\sum_{j\in N:(i,j)\in A} \sum_{a=1}^{K} a\, g^a_{ij} - \sum_{j\in N:(j,i)\in A} \sum_{a=1}^{K} a\, g^a_{ji} = \begin{cases} K & i = s \\ 0 & i \neq s,t \\ -K & i = t \end{cases} \tag{14}$$

The advantage of this reformulation is that M_3 can be easily expressed using the variables g. In fact, for a given arc $(i,j) \in A$ traversed by a paths, the number of overlaps is given by $\binom{a}{2} g^a_{ij}$. Thus,

$$M_3 = \sum_{a=1}^{K} \sum_{(i,j)\in A} \binom{a}{2} g^a_{ij}. \tag{15}$$

Using the previous relations, we obtain a discretized reformulation of the problem of finding a set of K paths that minimizes M_3, the total number of overlaps between pairs of paths:

$$\begin{array}{ll} \min & M_3 \\ \text{subject to} & (14), (15) \\ & \displaystyle\sum_{a=1}^{K} g^a_{ij} \leq 1, \quad (i,j) \in A \\ & g^a_{ij} \in \{0,1\}, \quad (i,j) \in A, \quad a = 1, \ldots, K \end{array} \tag{16}$$

This formulation has $O(Km)$ variables and $O(m+n)$ constraints.

The K paths in the solution can be retrieved by applying Algorithm 1. To run this algorithm, it is necessary to associate the value $f_{ij} = \sum_{a=1}^{K} a\, g_{ij}^{a}$ to each arc of the network. Like before, this procedure removes possible subtours. The next section presents preliminary empirical results involving the discretized formulation for minimizing the number of path overlaps, (16).

6.2 Preliminary computational experiments

An empirical assessment of formulation (16) was performed under the conditions established for the previous tests. The corresponding code was implemented similarly to the previous, and is designated by `MAO` hereafter. First, we notice that `MAO` was able to find an optimal solution for all the instances described in Section 3.3. Additionally, as shown in Table 10, the average run times obtained by this formulation do not exceed 1.4 seconds for the random instances, and are always shorter than 0.07 seconds for the grid instances. Nevertheless, they are worse than those of the code `MAR` described in Section 5.

Table 10: Average run times of `MAO` (seconds)

	K								
$R_{n,m}$	3	4	5	6	7	8	9	10	
$R_{100,500}$	0.029	0.046	0.059	0.070	0.079	0.083	0.092	0.101	
$R_{100,1000}$	–	0.099	0.118	0.135	0.151	0.162	0.190	0.210	
$R_{300,1500}$	0.104	0.147	0.187	0.216	0.235	0.258	0.289	0.323	
$R_{300,3000}$	0.159	0.264	0.345	0.392	0.449	0.501	0.587	0.666	
$R_{500,2500}$	0.200	0.227	0.295	0.360	0.377	0.414	0.479	0.529	
$R_{500,5000}$	0.312	0.443	0.599	0.682	0.882	1.027	1.152	1.304	
Average	0.161	0.204	0.267	0.309	0.362	0.408	0.465	0.522	0.337
	K								
$G_{h,w}$	3	4	5	6	7	8	9	10	
$G_{3,12}$	0.000	0.012	0.013	0.007	0.006	0.013	0.015	0.017	
$G_{4,36}$	0.029	0.021	0.050	0.045	0.056	0.052	0.064	0.090	
$G_{6,6}$	0.006	0.002	0.012	0.011	0.010	0.004	0.020	0.011	
$G_{12,12}$	0.022	0.027	0.035	0.036	0.046	0.045	0.055	0.062	
Average	0.014	0.016	0.028	0.025	0.030	0.028	0.038	0.045	0.028

The $\overline{D}_K$ values of the solutions obtained by the same formulation are summarized in Table 11. The average of the $\overline{D}_K$ values was 0.944. Therefore, the previous model produces sets of K paths with good dissimilarities and better than those obtained when applying `MRA` or `MAR`. For the grid instances, the new code `MAO` is able of finding the optimal value of D_K, given that in this case all the $s-t$ paths in the grid $G_{h,w}$ have length $h+w-2$, thus an optimal solution for D_K is an optimal solution for (16).

The optimal objective values obtained by code `MAO` are reported in Table 12. The objective

Table 11: $\overline{D}_K$ values of the solutions produced solving MAO to optimality

	K								
$R_{n,m}$	3	4	5	6	7	8	9	10	
$R_{100,500}$	0.849	0.905	0.910	0.903	0.902	0.891	0.891	0.885	
$R_{100,1000}$	–	0.978	0.958	0.967	0.971	0.971	0.972	0.970	
$R_{300,1500}$	0.925	0.952	0.960	0.955	0.947	0.944	0.941	0.937	
$R_{300,3000}$	0.972	0.960	0.971	0.975	0.974	0.975	0.974	0.974	
$R_{500,2500}$	0.935	0.961	0.961	0.963	0.958	0.957	0.953	0.951	
$R_{500,5000}$	0.868	0.929	0.937	0.960	0.956	0.960	0.965	0.962	
Average	0.910	0.948	0.950	0.954	0.951	0.950	0.949	0.946	0.944

	K								
$G_{h,w}$	3	4	5	6	7	8	9	10	
$G_{3,12}$	0.949	0.833	0.800	0.790	0.762	0.745	0.741	0.730	
$G_{4,36}$	0.982	0.982	0.892	0.860	0.848	0.848	0.826	0.815	
$G_{6,6}$	0.933	0.933	0.900	0.893	0.867	0.857	0.844	0.840	
$G_{12,12}$	0.970	0.970	0.955	0.952	0.939	0.935	0.929	0.927	
Average	0.958	0.930	0.887	0.874	0.854	0.846	0.835	0.828	0.876

function values represent the number of overlaps, which are bigger in the grid than in the random instances and tend to increase with K. Finally, the values of the integer programming gaps produced by the linear relaxation of (14) were 0 for all instances.

Table 12: Average optimal values of MAO (M_3^*)

	K							
$R_{n,m}$	3	4	5	6	7	8	9	10
$R_{100,500}$	2.3	3.0	5.2	8.0	12.0	16.7	22.3	28.7
$R_{100,1000}$	–	1.0	1.5	2.2	2.7	3.5	4.6	5.9
$R_{300,1500}$	1.5	2.3	3.4	5.3	8.0	11.0	14.9	19.4
$R_{300,3000}$	1.0	1.2	1.7	2.3	3.3	4.1	5.3	6.9
$R_{500,2500}$	1.4	1.8	3.4	4.9	7.5	10.0	13.6	17.7
$R_{500,5000}$	1.8	2.8	4.1	4.9	6.2	7.8	9.1	11.8

	K							
$G_{h,w}$	3	4	5	6	7	8	9	10
$G_{3,12}$	2.0	13.0	26.0	41.0	65.0	93.0	121.0	158.0
$G_{4,36}$	2.0	4.0	41.0	80.0	121.0	162.0	238.0	316.0
$G_{6,6}$	2.0	4.0	10.0	16.0	28.0	40.0	56.0	72.0
$G_{12,12}$	2.0	4.0	10.0	16.0	28.0	40.0	56.0	72.0

The results show that the formulation (16) outputs solutions with very good dissimilarity scores, indicating that minimizing the number of path overlaps might be a good approach to the K dissimilar paths problem.

7 Application to finding K dissimilar paths

In the previous sections, ILP formulations were presented for the problems of finding K paths between a given pair of nodes, such that:

- the number of repeated arcs is minimized (formulation `MRA`);
- the number of arc repetitions is minimized (formulation `MAR`);
- the number of arc overlaps for each pair of paths is minimized (formulation `MAO`).

Preliminary sets of experiments led to a few conclusions. While modeling exactly the dissimilarity if the path length is constant, the formulation `MAO` outperforms the remaining models in the case of the grid instances. Also, the formulation `MRA` showed some limitations when compared to `MAR`. Even if the corresponding code was generally fast, `MRA` did not solve all the instances to optimality and the $\overline{D}_K$ values it obtained were worse than with `MAR`. The purpose of the current section is to perform a deeper discussion of the behavior of the presented approaches from the perspective of the dissimilarity metric D. The discussion focuses only the best formulations, `MAR` and `MAO`, presented in Sections 5 and 6, respectively. Bearing that objective in mind, a comparison to a well known method extracted from the literature, the IPM method, proposed in Johnson et al. (1992) is introduced and tests using bigger, and thus harder to solve, instances (when compared to those used in the preliminary tests) were carried out. As mentioned in Section 2, the IPM method solves K shortest path problems and penalizes the cost of the selected arcs every time one of those paths is computed, in order to prevent their overlap as much as possible. This approach has often been used for comparisons in the literature due to its flexibility to incorporate features of various problems as well as the simplicity of its implementation. The `IPM` code implements this method, it is coded in C and it calls the CPLEX solver for solving the shortest path problems. `IPM` works with unitary arc costs and a penalization factor of 1 is added to the cost of the arcs of the most recently found path. Its time has polynomial complexity of $O(Km + Kn \log n)$, so the code is expected to run fast. This empirical analysis was run using the same machine and CPLEX solver used before. As mentioned, new sets of instances were considered. Their characteristics are as follows:

- Random graphs with $n = 1000, 3000, 5000$ nodes, obtained by generating randomly $m = dn$ arcs, with average degrees $d = 5, 10, 15, 20$. The results presented below are averages for finding sets of $K = 20$ paths over 30 instances generated for each dimension of these data sets, after the instances with disjoint solutions or with no feasible solution have been excluded.
- Grid graphs with 2500 nodes of the following sizes: $h \times w = 5 \times 500, 10 \times 250, 50 \times 50$, and with 10 000 nodes of the following sizes: $20 \times 500, 50 \times 200, 100 \times 100$. In this case the number of computed paths was $K = 5, 10, 15, 20$.

Besides the average of the dissimilarities of the sets of the K paths, $\overline{D}_K$, also the minimum pairwise dissimilarity of those paths, $mD_K = \min\{D(p_i, p_j) : i, j = 1, \ldots, K \wedge i \neq j\}$, is calculated for each instance. Then, the average of mD_K, denoted by $\overline{mD}_K$, for each set of instances is calculated.

Table 13 presents the results of each model for the random networks. The highest values of $\overline{D}_K$ are always associated with the code `MAO`. The percentage difference between these values

Table 13: Average results for $K = 20$ in random networks

	MAO				MAR				IPM			
	$\overline{D}_K$	$\overline{mD}_K$	OV	Time (s)	$\overline{D}_K$	$\overline{mD}_K$	OV	Time (s)	$\overline{D}_K$	$\overline{mD}_K$	OV	Time (s)
$R_{1000,5000}$	0.917	0.517	152.828	3.300	0.901	0.538	38.862	0.276	0.863	0.134	276.966	1.248
$R_{1000,10\,000}$	0.969	0.719	44.167	8.177	0.961	0.726	16.200	0.504	0.943	0.452	132.267	2.134
$R_{1000,15\,000}$	0.988	0.760	18.967	13.858	0.968	0.748	12.000	0.796	0.970	0.475	94.500	2.990
$R_{1000,20\,000}$	0.991	0.758	12.567	20.615	0.980	0.745	10.100	1.706	0.979	0.438	83.233	4.166
$R_{3000,15\,000}$	0.934	0.678	145.464	13.735	0.918	0.603	34.071	0.882	0.875	0.257	276.286	4.100
$R_{3000,30\,000}$	0.982	0.795	35.667	32.625	0.967	0.787	14.933	1.893	0.957	0.577	129.733	7.031
$R_{3000,45\,000}$	0.989	0.810	23.033	56.528	0.977	0.804	12.700	3.603	0.969	0.664	104.700	10.266
$R_{3000,60\,000}$	0.992	0.799	13.967	86.600	0.976	0.775	10.167	6.090	0.979	0.538	89.267	15.930
$R_{5000,25\,000}$	0.959	0.710	102.643	25.938	0.953	0.740	29.250	1.839	0.914	0.414	235.536	6.863
$R_{5000,50\,000}$	0.983	0.816	36.267	60.007	0.973	0.805	14.967	4.050	0.957	0.606	131.133	12.516
$R_{5000,75\,000}$	0.991	0.827	19.000	109.344	0.981	0.817	11.633	8.077	0.975	0.656	101.833	20.683
$R_{5000,100\,000}$	0.995	0.820	10.400	168.028	0.988	0.819	8.933	10.589	0.984	0.614	87.233	28.803

OV: Optimal value

and those obtained by the remaining two codes is listed in Table 14. The results for MAO are followed closely by MAR's (the difference between the values associated with the two models does not exceed 2.024%), and then by IPM's (with a difference of at most 6.317%). IPM has the worst performance of all three codes in this regard, except for the instances with $n = 1000$ and $m = 15\,000$, or with $n = 3000$ and $m = 60\,000$, for which it outperforms MAR. Moreover, according to Table 14, the $\overline{D}_K$ values obtained by IPM tend to be closer to the values found by MAO for the random instances when d grows, whereas the $\overline{D}_K$ values found by MAR have a more unstable behavior for the same instances.

The obtained $\overline{D}_K$ values improve as the average degree increases for the networks with the same number of nodes, for all three approaches. The increase of the number of network nodes has the same effect in the solutions. The increase in $\overline{D}_K$ is closely related to the run times, even if this latter parameter is more sensitive to the number of network nodes than to the network average degree. The instances with 5000 nodes and 100 000 arcs are solved in average times of 168.028 seconds, 28.803 seconds and 10.589 seconds when running MAO, IPM and MAR, respectively. Additionally, it is worth noting that MAR always ran faster than the other two approaches for these instances and that MAO was the slowest code.

The values of $\overline{mD}_K$ measure the worst case in terms of dissimilarity. In this case, the best results are found for the MAO and the MAR models. IPM is worse than the other formulations with this regard, while the latter show very similar performances (the differences between them do not exceed 5%).

Table 15 presents the results obtained by the different approaches for the grid instances. As already mentioned, the values of $\overline{D}_K$ obtained by MAO for these instances are optimal. On the other hand, IPM is also able of finding good solutions, it has the strong advantage of running in less time, and it is even capable of finding the best solution in terms of dissimilarity for the square grids. IPM's dissimilarity results are better for grids than for random networks. This

Table 14: Relative decrease of $\overline{D}_K$ for each code compared to MAO (%)

(a) Random networks, $K = 20$

	MAR	IMP
$R_{1000,5000}$	1.745	5.889
$R_{1000,10\,000}$	0.826	2.683
$R_{1000,15\,000}$	2.024	1.822
$R_{1000,20\,000}$	1.110	1.211
$R_{3000,15\,000}$	1.713	6.317
$R_{3000,30\,000}$	1.527	2.546
$R_{3000,45\,000}$	1.213	2.022
$R_{3000,60\,000}$	1.613	1.310
$R_{5000,25\,000}$	0.626	4.692
$R_{5000,50\,000}$	1.017	2.645
$R_{5000,75\,000}$	1.009	1.615
$R_{5000,100\,000}$	0.704	1.106

(b) Grid networks

	K	MAR	IMP		K	MAR	IMP
$G_{5,500}$	5	0.000	9.820	$G_{20,500}$	5	0.000	0.000
$G_{5,500}$	10	4.505	3.716	$G_{20,500}$	10	0.000	0.201
$G_{5,500}$	15	1.869	1.636	$G_{20,500}$	15	0.100	1.807
$G_{5,500}$	20	2.021	0.951	$G_{20,500}$	20	0.201	2.211
$G_{10,250}$	5	0.000	0.100	$G_{50,200}$	5	0.000	0.000
$G_{10,250}$	10	0.101	4.225	$G_{50,200}$	10	0.101	0.000
$G_{10,250}$	15	0.844	1.371	$G_{50,200}$	15	0.302	0.101
$G_{10,250}$	20	3.715	1.699	$G_{50,200}$	20	0.404	0.202
$G_{50,50}$	5	0.000	0.000	$G_{100,100}$	5	0.000	0.000
$G_{50,50}$	10	0.203	0.000	$G_{100,100}$	10	0.101	0.000
$G_{50,50}$	15	0.613	0.000	$G_{100,100}$	15	0.303	0.000
$G_{50,50}$	20	1.127	0.000	$G_{100,100}$	20	0.607	0.000

is due to the fact that in each of the used grids, all paths have the same length. In effect, IPM naturally avoids the arcs previously used, unless the cost associated to the increase in the number of arcs exceeds the penalty for repeating arcs. When applied to the random networks, the length of the paths varies and so does the dissimilarity of the solutions found by the method. As to the $\overline{D}_K$ values associated to MAR, these are smaller than MAO's but very close to them (with a difference of at most 4.505% when computing $K = 10$ in 5×500 grids, according to Table 14). The difference between the two increases with the number of paths K for the rectangular grids. The IPM finds the optimal dissimilarity more often than MAR. However, the average distance of the $\overline{D}_K$ values to the optimal is of 0.713% and 1.168%, for MAR and IPM, with standard deviations of 1.192 and 2.206, respectively. Therefore, the $\overline{D}_K$ values found by the IPM are more disperse when compared to the optimal than those found by MAR.

As to the run times, IPM was the fastest code for most of the considered grids, followed by the MAR code except when finding $K = 5$ paths in the 5×500 and 50×2000 instances. Moreover, the codes MAO, MAO and IPM present the best $\overline{mD}_K$ results in 38%, 33% and 54% of the times, even if the average and the standard deviation values of $\overline{mD}_K$ slightly favor MAR (0.773 and 0.377 for MAO, 0.773 and 0.356 for MAR, 0.710 and 0.358 for IPM). Also worthy of note is the significant decrease of the $\overline{mD}_K$ values observed with the increase of K.

The previous discussion confirms IPM as an interesting method for solving grid network instances if the run time is a limitation. IPM produced solutions with high average dissimilarities in significantly lower run times. IPM works best in square networks and worst in narrow rectangular networks. On the other hand, the results associated to MAR are also to be considered. In average the difference in the dissimilarity towards MAO is 3%. This, together with the very reasonable run times, makes MAR also an interesting approach for this type of network.

Table 15: Average results for grid networks

		MAO				MAR				IPM			
	K	$\overline{D}_K$	$\overline{mD}_K$	OV	Time (s)	$\overline{D}_K$	$\overline{mD}_K$	OV	Time (s)	$\overline{D}_K$	$\overline{mD}_K$	OV	Time (s)
$G_{5,500}$	5	0.998	0.992	10	0.934	0.998	0.994	8	1.259	0.900	0.507	3 016	0.969
$G_{5,500}$	10	0.888	0.000	2 541	2.345	0.848	0.135	2 503	1.293	0.855	0.024	8 308	1.905
$G_{5,500}$	15	0.856	0.000	7 597	3.578	0.840	0.010	4 998	1.319	0.842	0.004	15 878	2.801
$G_{5,500}$	20	0.841	0.000	15 180	4.700	0.824	0.008	7 493	1.306	0.833	0.000	25 973	3.713
$G_{10,250}$	5	0.996	0.984	10	1.073	0.996	0.984	8	0.903	0.995	0.981	1 302	0.511
$G_{10,250}$	10	0.994	0.981	72	1.873	0.993	0.969	40	1.004	0.952	0.504	3 140	1.020
$G_{10,250}$	15	0.948	0.275	1 409	4.126	0.940	0.310	1 285	0.862	0.935	0.140	5 638	1.490
$G_{10,250}$	20	0.942	0.035	2 822	5.002	0.907	0.062	2 530	0.907	0.926	0.450	8 782	1.973
$G_{50,50}$	5	0.990	0.959	10	0.875	0.990	0.959	8	0.690	0.990	0.980	500	0.224
$G_{50,50}$	10	0.984	0.939	72	1.548	0.982	0.918	40	0.867	0.984	0.939	1 052	0.420
$G_{50,50}$	15	0.979	0.898	218	2.395	0.973	0.857	98	0.823	0.979	0.898	1 688	0.617
$G_{50,50}$	20	0.976	0.888	452	3.225	0.965	0.816	180	0.912	0.976	0.857	2 416	0.825
$G_{20,500}$	5	0.998	0.994	10	6.967	0.998	0.992	8	6.377	0.998	0.996	2 600	2.644
$G_{20,500}$	10	0.997	0.985	72	12.315	0.997	0.985	40	8.922	0.995	0.975	5 301	5.217
$G_{20,500}$	15	0.996	0.983	218	16.158	0.995	0.975	98	9.780	0.978	0.568	8 943	7.810
$G_{20,500}$	20	0.995	0.975	452	22.246	0.993	0.965	180	5.500	0.973	0.494	13 036	10.551
$G_{50,200}$	5	0.996	0.984	10	4.844	0.996	0.984	8	5.337	0.996	0.988	1 250	1.580
$G_{50,200}$	10	0.994	0.968	72	9.770	0.993	0.968	40	6.839	0.994	0.972	2 552	3.201
$G_{50,200}$	15	0.992	0.964	218	13.901	0.989	0.944	98	5.367	0.991	0.972	3 958	4.678
$G_{50,200}$	20	0.990	0.940	452	21.639	0.986	0.927	180	7.416	0.988	0.948	5 516	6.395
$G_{100,100}$	5	0.995	0.980	10	6.428	0.995	0.980	8	5.394	0.995	0.980	1 000	1.337
$G_{100,100}$	10	0.992	0.960	72	7.605	0.991	0.960	40	6.217	0.992	0.970	2 052	2.759
$G_{100,100}$	15	0.990	0.944	218	11.764	0.987	0.929	98	4.977	0.990	0.949	3 188	4.056
$G_{100,100}$	20	0.988	0.929	452	17.592	0.982	0.909	180	6.754	0.988	0.955	4 417	5.424

OV: Optimal value

8 Concluding remarks

This manuscript addressed the search for K dissimilar paths connecting a given pair of nodes in a directed graph. Three integer formulations that aim to integrate a measure of the similarity of the paths in the model were introduced. The formulations have different motivations and model different problems with the general goals of minimizing the number of arcs that appear in more than one path, `MRA`, the number of arc reuses, `MAR`, or the total number of arc overlaps, `MAO`, while searching for sets of paths with a good dissimilarity. The performance of the new formulations and of a traditional method extracted from the literature, the iterative penalty method, `IPM`, was tested over random and grid networks, assessing the required run time as well as the average dissimilarity and the minimum dissimilarity of the solutions. Three conclusions can be highlighted:

- The code `IPM` was fast, as expected, given that it simply solves K shortest path problems, and provided solutions with good dissimilarity for the grid networks. On the other hand,

with the current implementation (which can be improved if using a labeling algorithm, for instance (Dijkstra, 1959)) it is slower than `MAR` for most of the tested instances, and the solutions it produced for the random networks are poor as to the average and the minimum dissimilarities of the set of K paths.

- The code `MAO`, is clearly the slowest of the codes but is still capable of finding 20 dissimilar paths in random networks with 5 000 nodes and 100 000 arcs within 170 seconds time. In spite of this drawback, `MAO` always produced the solutions with the best average dissimilarity, both for the random and the grid networks.

- Finally, the code `MAR` also provides good quality solutions with regard to the average and the minimum dissimilarities. Furthermore, in general `MAR` is much faster than the remaining codes. This formulation found the solution in less than 11 seconds for random networks with 5 000 nodes and 100 000 arcs.

It is worth noting that the introduced formulations can be extended to a case where overlaps are penalized according to a given cost for the involved arcs, with no significant difference of the implementation. One such case may represent the fact that path overlaps are acceptable in a given region or for a certain part of the network, but are strongly advised against in others. For instance, in hazmat transportation it is often the case that overlaps in different regions, such as country roads, urban areas, or industrial areas, raise different levels of concerns.

In the future, it would be interesting to investigate the relation between the dissimilarity and run time results of the formulations and some influencing characteristics of the networks, such as the density, average degree or topology. This aspect would allow a better use of the new models, particularly in the context of real-world applications of the problem, for, in those cases, a wider variety of networks can arise.

Acknowledgment Authors AM and MP were partially financially supported by the Portuguese Foundation for Science and Technology (FCT) under project grants UID/MAT/00324/2020 and UID/MULTI/00308/2020. The work was also partially supported by project P2020 SAICT-PAC/0011/2015, co-financed by COMPETE 2020, Portugal 2020 – Operational Program for Competitiveness and Internationalization (POCI), European Union's European Regional Development Fund, and FCT, and by FEDER Funds and National Funds under project CENTRO-01-0145-FEDER-029312.

References

Akgün, V., Erkut, E., Batta, R., 2000. On finding dissimilar paths. European Journal of Operational Research 121, 232–246.

Batta, R., Kwon, C., 2013. Handbook of OR/MS Models in Hazardous Materials Transportation. First ed., Springer, New York.

Bhandari, R., 1998. Survivable Networks: Algorithms for Diverse Routing. Kluwer Academic Publishers, USA.

Calvo, R., Cordone, R., 2003. A heuristic approach to the overnight security service problem. Computers & Operations Research 30, 1269–1287.

Clímaco, J., Craveirinha, J., 2019. MCDA/M in telecommunication networks: Challenges and trends, in: Advanced Studies in Multi-Criteria Decision Making. Chapman and Hall/CRC, New York, NY, pp. 11–55.

Constantino, M., Mourão, M.C., Pinto, L., 2017. Dissimilar arc routing problems. Networks 70, 233–245.

Correia, I., Gouveia, L., Saldanha-da Gama, F., 2008. Solving the variable size bin packing problem with discretized formulations. Computers & Operations Research 35, 2103–2113.

Correia, I., Gouveia, L., Saldanha-da Gama, F., 2010. Discretized formulations for capacitated location problems with modular distribution costs. European Journal of Operations Research 204, 237–244.

Dell'Olmo, P., Gentili, M., Scozzari, A., 2005. On finding dissimilar Pareto-optimal paths. European Journal of Operational Research 162, 70–82.

Dijkstra, E., 1959. A note on two problems in connection with graphs. Numerical Mathematics 1, 269–271.

Eppstein, D., 1998. Finding the k shortest paths. SIAM Journal on Computing 28, 652–673.

Erkut, E., Tjandra, S., Verter, V., 2007. Hazardous materials transportation. Handbooks in Operations Research and Management Science 14, 539–621.

Erkut, E., Verter, V., 1998. Modeling of transport risk for hazardous materials. Operations Research 46, 625–642.

Frank, A., 1985. Edge-disjoint paths in planar graphs. Journal of Combinatorial Theory, Series B 39, 164–178.

Gomes, T., Jorge, L., Girão-Silva, R., Yallouz, J., Babarczi, P., Rak, J., 2020. Fundamental schemes to determine disjoint paths for multiple failure scenarios, in: Rak, J., Hutchison, D. (Eds.), Guide to Disaster-Resilient Communication Networks. Springer International Publishing, Cham, pp. 429–453.

Gomes, T., Jorge, L., Melo, P., Girão-Silva, R., 2016. Maximally node and SRLG-disjoint path pair of min-sum cost in GMPLS networks: a lexicographic approach. Photonic Network Communications 31, 11–22.

Gouveia, L., 1995. A 2n constraint formulation for the capacitated minimal spanning tree problem. Operations Research 43, 130–141.

Gouveia, L., 1998. Using variable redefinition for computing lower bounds for minimum spanning and Steiner trees with hop constraints. INFORMS Journal on Computing 10, 180–188.

Gouveia, L., Saldanha-da Gama, F., 2010. On the capacitated concentrator location problem: a reformulation by discretization. Computers & Operations Research 33, 69–82.

Gouveia, L., Moura, P., 2012. Enhancing discretized formulations: the knapsack reformulation, the star reformulation. TOP 20, 52–74.

Guruswami, V., Khanna, S., Rajaraman, R., Shepherd, B., Yannakakis, M., 2003. Near-optimal hardness results and approximation algorithms for edge-disjoint paths and related problems. Journal of Computer and System Sciences 67, 473–496.

Hoffman, W., Pavley, R., 1959. A method for the solution of the n-th best path problem. Journal of the ACM 6, 506–514.

Hu, J.Q., 2003. Diverse routing in optical mesh networks. IEEE Transactions on Communications 51, 489–494.

Iqbal, F., Kuipers, F., 2015. Wiley Encyclopedia of Electrical and Electronics Engineering. Wiley, New York. chapter Disjoint paths in networks. pp. 1–14.

Jabbarzadeh, A., Azad, N., Verma, M., 2020. An optimization approach to planning rail hazmat shipments in the presence of random disruptions. Omega 96, 102078.

Johnson, P., Joy, D., Clarke, D., 1992. Highway 3.01, an enhancement routing model: program, description, methodology and revised user's manual. Working paper, Oak Ridge National Laboratories, Washington, DC .

Katoh, N., Ibaraki, T., Mine, H., 1982. An efficient algorithm for k shortest simple paths. Networks 12, 411–427.

Kuby, M., Zhongyi, X., Xiaodong, X., 1997. A minimax method for finding the k best "differentiated" paths. Geographical Analysis 29, 298–313.

Lombard, K., Church, R., 1993. The gateway shortest path problem: Generating alternative routes for a corridor location problem. Geographical Systems 1, 25–45.

Martí, R., González-Velarde, J., Duarte, A., 2009. Heuristics for the bi-objective path dissimilarity problem. Computers & Operations Research 36, 2905–2912.

Martins, E., 1984. An algorithm for ranking paths that may contain cycles. European Journal of Operational Research 18, 123–130.

Martins, E., Pascoal, M., 2003. A new implementation of Yen's ranking loopless paths algorithm. Quarterly Journal of the Belgian, French and Italian Operations Research Societies 1, 121–133.

Martins, E., Pascoal, M., Santos, J., 1999. Deviation algorithms for ranking shortest paths. International Journal of Foundations of Computer Science 10, 247–261.

Moghanni, A., Pascoal, M., Godinho, M.T., 2021. Finding k shortest and dissimilar paths. International Transactions in Operational Research 29, 1573–1601.

Mohri, S.S., Mohammadi, M., Gendreau, M., Pirayesh, A., Ghasemaghaei, A., Salehi, V., 2022. Hazardous material transportation problems: A comprehensive overview of models and solution approaches. European Journal of Operational Research 302, 1–38.

Pascoal, M., Clímaco, J., 2020. On a relaxed maximally disjoint path pair problem: A bicriteria approach. International Transactions in Operational Research 27, 2045–2063.

Suurballe, J., Tarjan, R., 1984. A quick method for finding shortest pairs of disjoint paths. Networks 14, 325–336.

Talarico, L., Sörensen, K., Springael, J., 2015. The k-dissimilar vehicle routing problem. European Journal of Operational Research 244, 129–140.

Tikani, H., Setak, M., Demir, E., 2021. Multi-objective periodic cash transportation problem with path dissimilarity and arrival time variation. Expert Systems with Applications 164, 114015.

Yen, J., 1971. Finding the k shortest loopless paths in a network. Management Science 17, 712–716.

Zajac, S., 2018. On a two-phase solution approach for the bi-objective k-dissimilar vehicle routing problem. Journal of Heuristics 24, 515–550.

A Proofs

Proposition A.1 *Let f be a feasible solution for formulation (2) and $p_1, \ldots, p_K$ be the corresponding paths output by Algorithm 1. Then, $p_1, \ldots, p_K$ are K loopless paths from s to t.*

Proof. Every path p_k is loopless because it is the solution of a shortest path problem with unitary costs, for $k = 1, \ldots, K$. Additionally, the vector f represents K paths from s to t, given that it satisfies the constraints (1a). Therefore, also $p_1, \ldots, p_K$ are K loopless paths in P. □

Proposition A.2 *Let f be a solution for formulation (2). Let also $p_1, \ldots, p_K$ be the paths output by Algorithm 1 when applied to f. Then, $p_1, \ldots, p_K$, are a loopless optimal solution for problem (2).*

Proof. Associated with $p_1, \ldots, p_K$, let us define $\bar{f}_{ij} = |\{p_k : (i,j) \in p_k \wedge k = 1, \ldots, K\}|$, and show that $\bar{f}$ is a feasible solution for (2) with optimal objective value.

We note that $f_{ij} \geq \bar{f}_{ij}$, for any $(i,j) \in A$, and consider two aspects:

1. Because $p_1, \ldots, p_K$ are K paths, then $\bar{f}$ satisfies (1a). Therefore, $\bar{f}$ is a feasible solution of (2).

2. Because $f_{ij} \geq \bar{f}_{ij}$, all the arcs present in the solutions represented by $\bar{f}$ are also in f, therefore, the number of superpositions in the paths represented by $\bar{f}$ never exceeds that associated with f. Therefore, the new solution is optimal.

It can then be concluded that $\bar{f}$ is an optimal solution of (2). □