ELSEVIER

European Journal of Operational Research 121 (2000) 232–246

EUROPEAN JOURNAL OF OPERATIONAL RESEARCH

www.elsevier.com/locate/orms

On finding dissimilar paths

Vedat Akgün [a], Erhan Erkut [b,*], Rajan Batta [a]

[a] *Department of Industrial Engineering, State University of New York at Buffalo, Buffalo, NY 14260-2050, USA*
[b] *Department of Finance and Management Science, Faculty of Business, University of Alberta, Edmonton, Alb., Canada T6G 2R6*

Abstract

Given a transportation network, this paper considers the problem of finding a number of spatially dissimilar paths between an origin and a destination. A number of dissimilar paths can be useful in solving capacitated flow problems or in selecting routes for hazardous materials. A critical discussion of three existing methods for the generation of spatially dissimilar paths is offered, and computational experience using these methods is reported. As an alternative method, the generation of a large set of candidate paths, and the selection of a subset using a dispersion model which maximizes the minimum dissimilarity in the selected subset is proposed. Computational results with this method are encouraging.

Keywords: Transportation planning; Path selection; Dangerous goods; Heuristics; Risk management

1. Introduction

In many single objective route-planning problems, it may be adequate to select a single "best" path from an origin to a destination. In contrast, in multi-objective routing problems, it is usually necessary to generate a set of paths and evaluate them under the relevant criteria. In addition to the multi-objective scenario, there may be instances where a decision maker is interested in developing backup routes for a daily shipment in case the best route becomes infeasible due to road construction or a snowstorm. To do this, one may wish to generate all paths with lengths that are within 10% of the length of the shortest path. It is possible to accomplish this via a k-shortest path algorithm (e.g., see Yen, 1971) by selecting a sufficiently large k. However, many of the k-shortest paths generated by such algorithms have the property that they are spatially very similar to one another. In some instances, this similarity between the generated paths may be acceptable. However, there are other instances where excessive similarity between the generated paths may be undesirable, and it may be preferable to generate spatially dissimilar paths. This is the central focus of our paper; we seek dissimilar (but not necessarily disjoint) paths from an origin to a destination on a transportation network.

Several specific examples to motivate the generation of spatially dissimilar paths are now presented. The first example provided the primary

* Corresponding author. Tel.: 001-780-492-3068; fax: 001-780-492-3325.
E-mail address: erhan.erkut@ualberta.ca (E. Erkut).

0377-2217/00/$ - see front matter
PII: S0377-2217(99)00214-3

motivation for Kuby et al. (1997). When working with a large, capacitated, multi-commodity network flow model, one may not be able to use arc variables (due to the sheer size of the model), and may be forced to use path variables instead. To reformulate the problem using path variables, the requirement is to select a small number of alternative paths between each origin–destination pair. If two of these alternative paths use the same bottleneck link, then one of them is useless for many solutions. The smaller the number of common links between the paths, the higher the total potential capacity between the origin and destination. As well, paths of excessive length are undesirable. Hence, the task is to generate a small number of paths (say 10) with acceptable lengths which have as few common links as possible.

Lombard and Church (1993) suggest a corridor location application for the same problem. When planning a pipeline between an origin and destination, it is desirable for the pipeline to be short, but there are many other concerns, such as topography and proximity to population centers. Suppose that the shortest path is infeasible, or undesirable due to these other concerns. A path that is only marginally different from the shortest path may suffer from the same infeasibility or undesirability. Hence, it is desirable to generate a number of paths that are not very similar topologically.

The third example is the primary reason we started studying this problem: routing for hazardous materials. There are at least two reasons why dissimilar paths may be desirable in hazardous materials route planning. The first reason is in recognition of the fact that accident probabilities can increase considerably in the case of adverse weather conditions, in which case it is desirable to have several dissimilar paths to increase the probability of being able to select a path that is not impacted by a weather system. A set of dissimilar paths is also needed to ensure spatial risk equity for multiple shipments of hazardous materials. One way to "spread" the risk is by using more than one path for the shipments, and ensuring that the alternate paths are quite dissimilar. As an example, Fig. 1 displays two routes between Pennsuco, FL and the proposed high-level nuclear waste repository in Yucca Mountain, NV. These routes are within 2.4% of one another in terms of length, yet they are quite dissimilar.

The remainder of this paper is organized as follows. In Section 2, we review the directly related literature. In Section 3, we offer a critical discussion of the three existing methods. In Section 4, we design a computational test to compare the efficiency and effectiveness of the existing methods and summarize our computational results. We describe a solution technique, derived by posing this problem as a p-dispersion problem, in Section 5. The p-dispersion problem, which is related to the clique and independent set problems in graph theory, is a facility location model designed to maximize the separation (minimum distance) between selected facilities. Finally, in Section 6, we provide our concluding remarks.

2. Related literature

The problem of finding the shortest path between an origin and a destination is one of the best known applications of operations research to transport planning. There exist a number of algorithms to solve this problem (Ahuja et al., 1993). As an extension of the shortest path problem, a number of researchers have studied the problem of finding the second-shortest path, the third-shortest path, and so on. These efforts gave rise to a number of k-shortest path algorithms (Yen, 1971; Shier, 1979; Katoh et al., 1982; Skiscim and Golden, 1987; Miaou and Chin, 1991).

Obviously, the k-shortest path algorithms are capable of generating a large number of alternative paths, which can be useful in a number of transport planning instances. However, many of these alternative paths are likely to share a large number of edges. If one can define a measure of dissimilarity between these paths (similar to a distance metric), then a subset of these paths can be selected so that the minimum dissimilarity is maximized. In fact, this is an application of the "discrete p-dispersion" model, a facility location model which considers the problem of selecting a dispersed subset of a given set of points in some space

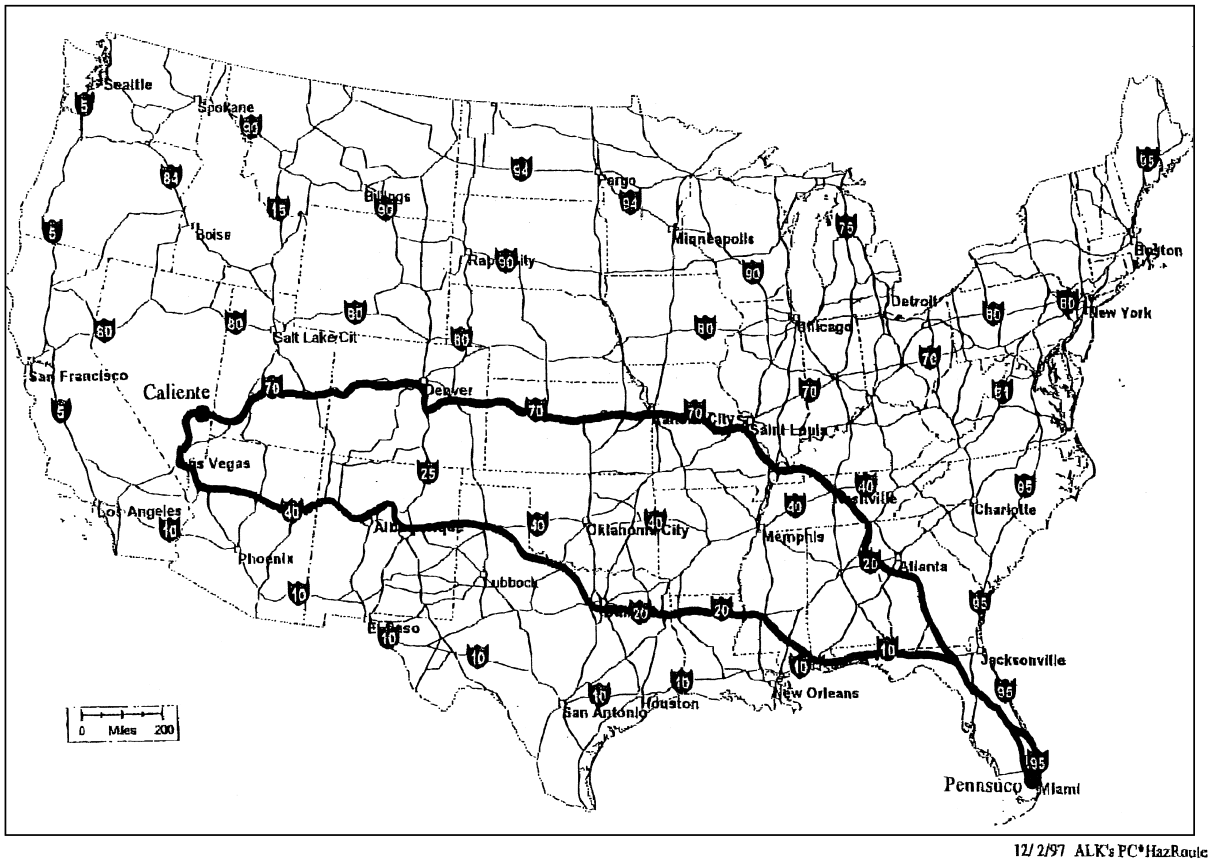

Fig. 1. Two alternate paths between Pennsuco, FL and high-level nuclear waste repository in Yucca Mountain.

so that the minimum distance between pairs of selected points is maximized.

While there is a considerable literature on different versions of the p-dispersion problem, Kuby (1987) was the first one to pose it on a general network. Erkut (1990) described two branch-and-bound methods and a heuristic for the discrete p-dispersion problem. More recently, Erkut et al. (1994) reported an empirical comparison of ten heuristics to solve this problem. A set of p dissimilar paths can be found by applying a p-dispersion algorithm to the set of alternate paths generated by a k-shortest path algorithm ($p < k$). In Section 5 a dissimilarity measure to make this strategy functional is presented, and computational experience with this method is reported. The next section contains details of the three methods from the literature that have been designed with the specific goal of generating a set of dissimilar paths.

3. Discussion of existing methods

In this section, a detailed analysis of existing methods is given, focusing on the advantages and disadvantages (or drawbacks) of these methods. Computational experience with these methods is presented in the next section.

3.1. Iterative penalty method (IPM)

The Iterative Penalty Method (IPM) is based on a repetitive application of an appropriate shortest path algorithm. After each application of the algorithm, a (cumulative) penalty on the

impedance of all links in the resulting shortest path is imposed. Hence, repeated selection of the same set of links is discouraged, and dissimilar paths are generated as a result. This method was suggested in the context of hazardous materials routing by Johnson et al. (1992), and used by Ruphail et al. (1995), in a decision support system to generate economically different paths over a network characterized by time-dependent link travel times. There are several dimensions in the implementation of the penalty mechanism:

- *Penalized units*: Penalties can be applied to the links, or nodes, or both.
- *Penalty structure*: An additive penalty (i.e. adding a fixed positive amount to the impedance), or a multiplicative penalty structure (i.e. multiplying the current impedance by a factor greater than one) can be used. If one uses a multiplicative penalty formula, the new impedance can be based on the current impedance (which may have been penalized before), or on the original impedance.
- *Penalty magnitude*: If a relatively large penalty is chosen, then links that appear in generated paths are discouraged more heavily. Smaller penalties, on the other hand allow for more frequent appearances of links in generated paths.
- *Penalized paths*: Penalties can be applied to the most-recently found path only, or to all paths found so far.

Clearly, the primary advantage of this method is its simplicity. To implement the method, all that is needed is a shortest path subroutine and a penalty mechanism. However, the ad hoc nature of this method is a drawback. The method has no way of evaluating the quality of the set of the paths it generates in terms of the spatial differences and the lengths of the paths. Additionally, to operationalize this method, it is necessary to make an (arbitrary) decision about every one of the four dimensions discussed above, and the results will depend on these decisions. For example, a small penalty may not achieve the goal of dissimilarity, while a large penalty may eliminate a great many viable paths from consideration. While it is possible to experiment with different penalty mechanisms and magnitudes, this search procedure would be somewhat arbitrary without developing an evaluation scheme for the generated set. Furthermore, the selected (fine-tuned) settings for the penalty strategy would be strictly problem-dependent (and not portable). Finally, there is some risk of numerical instability (e.g., division by zero), particularly if one uses a multiplicative penalty with a large penalty factor and penalizes based on the current impedance.

3.2. Gateway shortest paths (GSPs)

The second method, GSP, is based on a constrained shortest path problem; this was proposed by Lombard and Church (1993). For a given origin and a destination, a "gateway shortest path" is the shortest path between the origin and the destination that is constrained to go through a specified node, called a "gateway". It is possible to generate a large number of such paths by forcing them through different gateways. To evaluate the similarity between two paths, a concept of "area under a path" is used. It is assumed that the network is embedded on a plane, and there exists a coordinate system on the plane. The "area" under a given path is the physical area between the path and the x-axis of the coordinate system. The similarity between two paths is the absolute difference between their "area-under-the-path" figures.

The greatest appeal of this method is the generation of a large number of alternative paths by using a shortest path algorithm twice. It also yields an efficient set of paths in terms of their path lengths and their dissimilarity from the shortest path. However, there are some critical shortcomings of this method which we summarize in point form. These drawbacks are demonstrated via numerical examples in Akgün et al. (1997).

- Some of the gateway paths may contain loops, and in most routing applications looping paths are undesirable.
- It may be impossible to identify some (rather desirable) dissimilar paths as GSPs.
- A desirable GSP may be identified as a dominated path and eliminated in the final phase of GSP.
- GSP can generate two paths that are very similar to one another because it focuses on the

similarity of the candidate paths to the shortest path only, and not to the other GSPs.
- The area difference between a path and the shortest path, is computed recursively (as part of the shortest path algorithm), and particular attention must be paid to prevent double-counting.

Given these drawbacks, it is suggested that GSP be used with caution, if at all, to generate a set of (mutually) dissimilar paths between an origin and a destination.

3.3. Minimax method

This third method has been proposed recently by Kuby et al. (1997). It aims to generate a set of "differentiated" paths by selecting a subset of a large set of paths. Attention is paid to the similarity between the selected paths and to their lengths so that the final set is mutually dissimilar and includes relatively short paths. The algorithm starts by generating k-shortest paths between the origin and the destination using a k-shortest path algorithm. These k paths are then processed and a dissimilar subset (DS) is constructed iteratively. An index is defined to measure the desirability of a candidate path for inclusion in DS.

The first path included is the shortest path. Suppose P_1 is the shortest path, and $d(P_1)$ is its length. Also suppose that we wish to find a second path according to two criteria: minimize length, minimize similarity with P_1. Consider an arbitrary path, say P_j. Denote the length of the shared portion by P_1 and P_j by $d^s(P_j, P_1)$, and the length of P_j not shared by P_1 by $d^n(P_j, P_1)$. Hence, the length of P_j is $d^s(P_j, P_1) + d^n(P_j, P_1)$. The minimization of this is the first objective. The similarity between P_1 and P_j can be minimized by minimizing $d^s(P_j, P_1)$. These two objectives can be combined using the weighting method (thereby establishing a linear tradeoff). Using a weight of β for the second objective, and scaling the weighted objective by $d(P_1)$ (which is a constant), we have:

$$\frac{(1+\beta)d^s(P_j,P_1) + d^n(P_j,P_1)}{d(P_1)}, \quad \beta \geqslant 0. \tag{1}$$

Minimization of this index with respect to j yields a relatively short path that is also dissimilar to the shortest path. Of course, the selected path will depend on the value of the weight β. The second objective can be emphasized by using a large β, whereas the first one can be emphasized by using a small β. Kuby et al. (1997) use $\beta = 1$ in their numerical examples. The third path is selected to minimize the maximum (hence the "minimax" name) of the indices between the candidates and the first two paths. The procedure continues in this fashion until the desired number of paths have been selected. By defining the index

$$M(P_j,P_i) = \frac{(1+\beta)d^s(P_j,P_i) + d^n(P_j,P_i)}{d(P_1)} \tag{2}$$

the model is equivalent to

$$\operatorname*{Min}_{j \notin DS}\left[\operatorname*{Max}_{i \in DS}\{M(P_j,P_i)\}\right].$$

Among the existing three methods, this is the only one that explicitly compares pairs of paths for similarity. While this is a clear advantage, this method has several drawbacks. The first drawback is algorithmic. If we define

$$M'(P_j,P_i) = K - M(P_j,P_i),$$

where K is a sufficiently large number, then the maximization of the minimum of $M'(P_j,P_i)$ values is equivalent to the minimization of the maximum of the $M(P_j,P_i)$ values. Hence, the underlying problem is, in fact, a p-dispersion problem, and the algorithm proposed by Kuby et al. (1997) amounts to a simple greedy construction heuristic for the p-dispersion problem. However, there are exact algorithms and efficient heuristics available for the p-dispersion problem, and the Kuby et al. (1997) algorithm may not be the best way to solve this problem.

The second drawback is associated with the definition of the "distance not shared" between two paths. This value, $d^n(P_j,P_i)$, is defined as the length of P_j not common to P_i. This definition implies that if $P_i \neq P_j$, then $d^n(P_j,P_i) \neq d^n(P_i,P_j)$ unless $L(P_i) = L(P_j)$. Therefore, this measure is not symmetrical, and the final DS set may depend on

the order of selection of its members as demonstrated in the following numerical example.

Assume that we want to choose 3 dissimilar paths from a set of 4 paths. Table 1 summarizes all relevant data.

Let Q and C be the sets for the selected paths and candidate paths, respectively, and let the value of β be 1. First, the shortest path, P_1, is selected. The second path to be selected will be the one that minimizes $M(P_j, P_1)$ for $j = 2, 3, 4$. There is a tie between P_2 and P_4 for the minimum. If P_2 is selected, the third path will be found by

$$\underset{j=3,4}{\text{Min}}\left[\underset{i=1,2}{\text{Max}}[M(P_j, P_i)]\right],$$

which is P_3. Therefore the solution is $Q = \{P_1, P_2, P_3\}$. However, the solution would be $Q = \{P_1, P_2, P_4\}$ by

$$\underset{i=2,3}{\text{Min}}\left[\underset{i=1,4}{\text{Max}}[M(P_j, P_i)]\right]$$

if P_4 was selected as the second path. Hence, the tie-breaking decision plays a critical role in the generation of the final solution due to the asymmetry of the measure.

Finally, the definition of the index $M(P_j, P_i)$ offers a complication to potential users of this method. As detailed above, β serves as the "weight" for the dissimilarity criterion. While this definition allows the flexibility to emphasize one of the two objectives, one may have to experiment with different values of β before deciding on its value. For example, with $\beta = 1$ (as in Kuby et al., 1997) in the above numerical example, P_2 is selected in the second iteration although it is quite similar to P_1. To generate a dissimilar set, one has to use a larger value of β.

Table 1
Shared (not-shared) distance matrix and lengths of 4 paths

Path	P_1	P_2	P_3	P_4	Length
P_1	N/A	N/A	N/A	N/A	5
P_2	5 (1)	N/A	4 (2)	5 (1)	6
P_3	5 (2)	4 (3)	N/A	3 (4)	7
P_4	3 (5)	5 (3)	3 (5)	N/A	8

4. Computational experience

In our computational experiments, we compared the performance of the three algorithms mentioned above. As an example network, we used the provincial highway road network of Alberta, Canada, which has 305 nodes and 854 arcs. Eleven representative origin–destination pairs on this network were chosen for consideration. The shortest distances between the OD pairs range from 230 to 566 km. Our experience with the 11 pairs is summarized, but we focus on one typical OD pair for detailed examples due to space limitations. The methods were coded in C+ and the programs executed on a Pentium 200 MMX machine.

Three important measures to judge the quality of a set of dissimilar paths were focused on: the average path length, the average dissimilarity between pairs of paths, and the minimum dissimilarity in the set. To measure the spatial (dis)similarity between any pair of paths, a similarity index defined in Erkut and Verter (forthcoming) was used, namely

$$S(P_i, P_j) = [L(P_i \cap P_j)/L(P_i) + L(P_i \cap P_j)/L(P_j)]/2, \tag{3}$$

where $L(\cdot)$ denotes the length of a path. A corresponding dissimilarity index is defined as

$$D(P_i, P_j) = 1 - S(P_i, P_j). \tag{4}$$

For convenience, the indexes are multiplied by 100 and we refer to them as percentages.

4.1. Gateway shortest path (GSP) method

The GSP method was applied to each OD pair to find all gateway paths, and the Tchebycheff metric was used to determine the efficient set with respect to length and dissimilarity. The area differences were computed either in relation to the horizontal or the vertical axis, depending on the alignment of the OD pair. (If the horizontal distance between the origin and the destination is greater (less) than the vertical distance between them, we used the horizontal (vertical) axis as the base.) Table 2 shows the number of possible

Table 2
Number of candidate gateway paths and number of paths in the efficient set

	Origin–destination pair										
	1	2	3	4	5	6	7	8	9	10	11
Gateway paths	292	290	271	287	293	292	279	294	286	287	283
Loopless paths	116	130	142	119	119	141	144	85	137	85	144
Distinct paths	72	79	70	65	60	77	78	54	66	52	68
Efficient paths	10	16	18	15	18	13	24	16	23	15	25

gateway paths, number of loopless paths, number of disjoint paths and the number of paths in the efficient set.

Note that, for each OD pair, there are many paths with loops and a large number of the loopless paths are identical. Also, the number of paths in the efficient set is small when compared to the number of possible gateway paths.

GSP does not do well on the three measures of interest; it is prone to finding long and similar paths. When generating the efficient set, many long gateway paths and some similar paths are excluded. Nevertheless, the average length of the efficient paths is considerably longer than the length of the shortest path. As well, there are many similar paths in the efficient set since GSP compares each path only with the shortest path for similarity. For example, the minimum dissimilarity for OD Pair 7 is only 0.3%, implying that at least two of the efficient paths are almost identical.

It is important to note that GSP was designed to solve a corridor location problem in which the area under study was divided into grid cells and the center of each grid cell was considered as a node. This network does not have the characteristics of a general road network. As well, the original goal of GSP is not to generate a set of mutually dissimilar paths, but to generate alternatives to the shortest path. Our experience with GSP on a road network showed that GSP is not a suitable method for our problem. It failed to generate a large number of alternate paths, eliminated some good candidates, and selected some spatially similar paths.

It is possible to improve GSP's performance by increasing the number of loopless gateway paths generated. If a gateway path has a loop, then a certain number of k-shortest paths can be found from the origin to the gateway node and from the gateway node to the destination. It may be possible to construct a number of loopless OD paths through that gateway node by combining different pairs of these subpaths. In fact, this procedure can also be used for gateway paths without loops in order to generate more than one path passing through a gateway node. Once a large number of alternate paths are found, a p-dispersion algorithm can be used to select a dissimilar set of paths as discussed in Section 5.

4.2. Iterative penalty method (IPM)

IPM was applied to find r alternative paths for each OD pair, where $r = 3, \ldots, 25$. As mentioned in Section 3, there are several dimensions in the implementation of the penalty mechanism. In our experiment, we assigned penalties to the links (as opposed to the nodes). A multiplicative penalty mechanism was used and penalties were computed in two different ways: (i) based on the current impedance (CI) (which may include past penalties), (ii) based on the original impedance (OI). Both small and large penalty magnitudes were considered. The IPM method was tested with penalty factors from 1% to 10% of the impedance with 1% increments, and from 10% to 100% of the impedance with 10% increments. At each iteration, penalties were assigned only to the links on the most-recently found path.

In our implementation of IPM, we rejected repeated paths. If a repeat shortest path is found at an iteration, penalties are applied to the links on this path although the path is rejected. Fig. 2 shows the number of rejected paths generated when attempting to find 20 distinct paths with OI

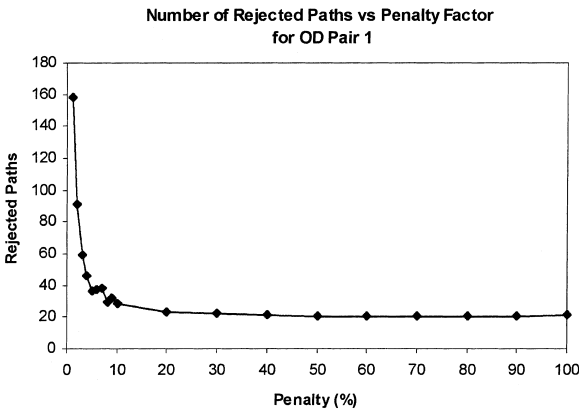

Fig. 2. Number of rejected paths as a function of the penalty factor.

for OD Pair 1. As one would expect, higher penalty factors result is smaller rejected paths since they alter the relative impedances more severely than lower penalty factors. For penalty factors above 20%, the number of rejected paths is quite low, and seems independent of the exact value of the penalty factor. It was observed that with penalty factors above 10%, the number of rejected paths was approximately the same for OI and CI, while for smaller factors, OI resulted in a higher number of rejects than CI. These general patterns were observed for all OD pairs.

As expected, the average path length increases gradually as the number of paths generated increases. The rate of increase in the average path length is almost constant when the penalty factor is varied between 1% and 10%. However, the average path length increases faster when the penalty factor is increased from 10% to 100%.

The values of the average path length (AvLe), the average dissimilarity between pairs of paths (AvDi) and the minimum dissimilarity (MiDi) for OD Pair 1 are summarized in Table 3 for the OI alternative.

Generally speaking, AvLe increases with the penalty factor and with the number of paths required (as one would expect). However, there are certain exceptions. For example, with a 1% penalty factor, AvLe for 11 paths is lower than the AvLe for 10 paths for OD Pair 1. As another example, when generating 4 paths for OD pair 1, a penalty factor of 70% results in a smaller AvLe than a factor of 40%. However, as the number of paths increase, the occurrence of these counter-intuitive cases becomes rare.

High penalty factors (40% and higher) result in higher MiDi values than low penalty factors. For example, a factor of 100% generates the 10 maximally dissimilar paths for OD Pair 1. However, for generating 20 dissimilar paths a factor of 70% works better than 100%. Hence, there are no general rules for selecting an "ideal" penalty factor for maximal dissimilarity.

Comparing briefly the results produced by CI and OI, in general CI results in higher MiDi values than OI does. Although the difference in AvDi values between OI and CI is insignificant, with high penalty factors, AvLe for CI can be considerably higher than for OI.

Although we used only one network in our experiment, we believe that the performance of a certain set of algorithm parameters will depend on the underlying network. For example, a low penalty factor may succeed in generating more dissimilar paths on a dense network than on a sparse network.

We conclude that IPM is a suitable algorithm to generate a set of alternative paths efficiently, but the selected paths may not be very desirable with respect to one of the two relevant criteria. However, a large number of alternative paths can be generated easily using IPM, and a subset of this set selected in a second phase (see Section 5.2).

It may be possible to refine IPM via a selective penalty mechanism using different penalty factors for different links on the path. For example, in a hazmat transport context, links passing through densely populated areas may be penalized more heavily than links that pass through areas with sparse populations.

4.3. Minimax method

To initialize this method, between 100 and 200 k-shortest paths for each OD pair were found. Then, the minimax method of Kuby et al. (1997) was used to select anywhere from 3 to 25 paths for each OD pair. It was observed that the time and

Table 3
IPM solution by using penalty assignment with original impedance (OI) (O–D Pair 1; shortest length = 358)

No.	AvLe					AvDi					MiDi				
	1%	10%	40%	70%	100%	1%	10%	40%	70%	100%	1%	10%	40%	70%	100%
3	389.1	389.1	422.1	422.1	422.1	65.3	65.3	94.2	94.2	94.2	44.7	44.7	85.4	85.4	85.4
4	394.9	406.5	453.2	429.9	453.2	58.7	76.5	93.7	88.1	93.7	4.8	44.7	82.1	67.1	82.1
5	407.5	415.3	445.4	449.4	461.5	67.7	78.8	88.4	89.2	91.7	4.8	44.7	56.9	67.1	77.8
6	416.0	422.4	453.4	456.9	453.6	67.1	76.6	87.9	88.5	87.9	4.4	4.4	56.9	67.1	56.9
7	420.9	427.6	455.2	447.6	474.3	72.4	76.2	85.3	84.4	88.1	4.4	4.4	13.0	22.5	56.9
8	426.5	435.1	448.3	456.1	496.8	70.9	76.8	83.8	84.6	88.6	4.4	4.4	13.0	22.5	56.9
9	430.1	432.6	453.4	461.3	512.6	72.8	74.9	83.4	84.3	88.2	4.4	4.4	13.0	22.5	56.9
10	435.9	438.7	469.5	480.6	518.4	73.1	76.5	84.7	85.5	87.6	4.4	4.4	13.0	22.5	39.0
11	432.6	446.7	468.5	486.2	513.6	73.0	78.8	83.7	85.2	86.2	1.7	4.4	13.0	22.5	13.0
12	435.4	442.8	469.6	494.0	529.5	71.4	77.8	83.3	85.2	86.6	1.7	1.7	13.0	22.5	13.0
13	439.9	441.4	464.7	491.2	536.9	73.5	77.8	82.5	84.3	86.6	1.7	1.7	13.0	22.5	13.0
14	446.0	441.7	470.3	503.9	528.0	74.7	77.0	82.6	84.8	86.0	1.7	1.7	13.0	22.5	13.0
15	446.9	449.4	476.1	503.0	523.8	73.5	78.5	82.9	84.5	85.4	1.7	1.7	13.0	22.5	13.0
16	450.6	451.5	474.9	496.6	525.5	73.4	78.5	82.5	84.0	85.2	1.7	1.7	13.0	22.5	13.0
17	456.2	452.0	486.4	497.0	526.9	74.4	77.9	83.2	83.9	85.0	1.7	1.7	13.0	22.5	13.0
18	453.9	451.5	484.9	508.2	543.4	74.5	77.6	82.8	84.3	85.4	1.7	1.7	13.0	22.5	13.0
19	453.5	456.7	483.7	510.3	540.2	75.1	78.2	82.4	84.2	84.9	1.7	1.7	13.0	22.5	13.0
20	455.1	453.9	494.4	507.3	535.7	74.4	78.1	82.9	83.8	84.7	1.7	1.7	13.0	22.5	13.0
21	455.4	454.5	495.9	519.2	544.5	73.2	77.6	82.9	84.2	84.8	1.7	1.7	13.0	22.5	13.0
22	458.1	452.6	498.1	513.7	542.1	73.0	77.5	82.8	83.7	84.5	1.7	1.7	13.0	22.5	13.0
23	455.8	454.7	494.4	516.1	540.2	74.0	77.7	82.5	83.7	84.3	1.7	1.7	13.0	17.1	13.0
24	459.8	459.5	495.3	515.4	540.0	74.5	78.3	82.3	83.6	84.2	1.7	1.7	13.0	17.1	13.0
25	463.7	460.5	491.5	511.2	538.1	74.8	77.9	82.0	83.1	84.0	1.7	1.7	13.0	17.1	13.0

storage requirements of finding k-shortest paths increase quickly with k. For example, when k was increased from 100 to 200, the computational time quadrupled (increasing from 0.7 hours to 2.5 hours).

Since the k-shortest path algorithm finds many paths that are small variations of the shortest path, the lengths of the 100th (and the 200th) shortest paths are between 1.8% (2.2%) and 35.4% (38.5%) higher than the length of the shortest paths for the 11 OD pairs in the experiment.

The minimax method was analyzed from different angles. As discussed in Section 3, a parameter (β) determines the weight of the similarity objective in the model. Intuitively, an increase in the value of β would improve the average and minimum dissimilarity values while the average length of the paths would increase. In order to analyze the effect of β, we applied the minimax method for a range of β values ($\beta = 1, 2, \ldots, 5$) with 100 and 200 paths. Table 4 summarizes the results for OD Pair 1 with 200 paths.

The average length (AvLe) increases slightly as the number of selected paths and the value of β increase. The average dissimilarity (AvDi) value is quite insensitive to increases in β, while the minimum dissimilarity (MiDi) value is improved for larger values of β. In general, the values of AvDi and MiDi do not change for $\beta > 3$.

If the length of the 200th path is close to the length of the shortest path, then the average similarity and the minimum similarity values for the selected path set might be very low. For example, for OD Pair 7, the 200th shortest path is only 4.9% longer than the shortest path. Even if we find only three dissimilar paths (out of 200) for this pair using the minimax method, the minimum dissimilarity for these three paths is merely 43%. We would need a much larger set of k-shortest paths to find a better result for this OD pair. In general, the number of k-shortest paths needed to find a good solution with the minimax method depends on the type of network and the location of the OD pair.

Table 4
Minimax solution for 200 k-shortest paths (O–D Pair 1; shortest length = 358)

NoP	AvLe					AvDi					MiDi				
	$B=1$	$B=2$	$B=3$	$B=4$	$B=5$	$B=1$	$B=2$	$B=3$	$B=4$	$B=5$	$B=1$	$B=2$	$B=3$	$B=4$	$B=5$
3	405.1	405.1	405.1	405.1	405.1	76.3	76.3	76.3	76.3	76.3	65.6	65.6	65.6	65.6	65.6
4	403.4	414.8	414.8	414.8	414.8	68.1	70.3	70.3	70.3	70.3	44.7	57.8	57.8	57.8	57.8
5	410.4	419.6	419.6	419.6	419.6	66.8	68.4	68.4	68.4	68.4	44.7	53.5	53.5	53.5	53.5
6	409.3	423.9	423.9	423.9	423.9	63.9	67.8	67.8	67.8	67.8	34.6	49.9	49.9	49.9	49.9
7	414.5	421.0	424.3	424.3	424.3	64.4	63.3	63.7	63.7	63.7	34.6	34.6	37.3	37.3	37.3
8	415.1	421.5	424.4	424.4	424.4	62.2	61.9	62.1	62.1	62.1	34.6	34.6	36.2	36.2	36.2
9	412.4	421.7	424.3	426.5	426.5	59.5	61.8	62.0	62.5	62.5	22.5	30.8	30.8	36.2	36.2
10	412.5	423.9	426.2	426.2	426.2	59.0	62.2	62.4	62.4	62.4	22.5	30.8	30.8	30.8	30.8
11	415.4	420.9	427.8	427.8	427.8	58.9	60.4	62.0	62.0	62.0	22.5	22.5	30.8	30.8	30.8
12	411.1	422.9	428.3	428.3	428.3	56.2	60.2	60.7	60.7	60.7	1.9	22.5	30.5	30.5	30.5
13	412.9	424.5	429.5	429.5	429.5	56.5	60.5	61.0	61.0	61.0	1.9	22.5	30.5	30.5	30.5
14	415.3	425.6	430.3	430.3	430.3	57.9	60.7	61.2	61.2	61.2	1.9	22.5	30.5	30.5	30.5
15	417.0	426.5	430.9	430.9	430.9	58.6	59.7	60.2	60.2	60.2	1.9	22.5	28.7	28.7	28.7
16	416.5	426.9	430.7	430.7	430.7	57.5	60.0	59.0	59.0	59.0	1.9	22.5	22.3	22.3	22.3
17	415.6	425.2	430.9	430.9	430.9	56.4	59.8	59.2	59.2	59.2	1.9	21.0	22.3	22.3	22.3
18	414.8	425.9	429.1	430.0	430.0	55.2	58.9	59.2	58.4	58.4	1.9	21.0	21.0	16.6	16.6
19	414.3	427.0	428.3	428.3	428.3	54.5	58.0	58.5	58.5	58.5	1.9	21.0	16.6	16.6	16.6
20	415.1	426.4	427.5	427.5	429.0	53.6	57.4	57.4	57.4	57.7	1.9	16.6	16.6	16.6	16.6
21	415.2	425.7	427.8	427.8	429.3	53.8	56.4	57.0	57.0	57.2	1.9	16.6	16.6	16.6	16.6
22	416.2	422.9	426.9	426.9	429.7	53.4	55.2	56.1	56.1	57.0	1.9	1.9	12.8	12.8	16.6
23	416.0	422.0	426.1	427.4	428.7	52.5	54.3	55.6	56.0	56.2	1.9	1.9	12.8	12.8	12.8
24	417.1	421.3	426.5	426.5	427.8	53.2	53.9	55.5	55.5	55.8	1.9	1.9	12.8	12.8	12.8
25	417.9	420.9	424.0	425.9	427.2	53.1	53.1	54.4	54.7	55.0	1.9	1.9	1.9	11.1	11.1

5. The discrete p-dispersion problem

In the classic discrete p-dispersion problem, p out of m given points $(1 < p < m)$ are selected in some space, where the objective is to maximize the minimum distance between any two of the selected points. Defining M to be the set of candidate points with $|M| = m$, P to be a subset of M with $|P| = p$, and w_{ij} to be the distance between candidates i and j, the p-dispersion problem can be expressed as follows:

$$\underset{P\subseteq M}{\text{Maximize}}\left[\underset{i\neq j;i,j\in P}{\text{Minimize}}\{w_{ij}\}\right].$$

In our problem, w_{ij} represents the dissimilarity between any two paths (computed using (4)). Given m paths, p paths were chosen so that the minimum dissimilarity between any two of the selected paths is maximized. To solve this p-dispersion problem, a two-phase heuristic was used. The basic idea behind the heuristic is to construct an initial solution in a semi-greedy fashion, and then perform a local search to improve the initial solution (Erkut, 1990).

Consider the numerical example used in Section 3 for the analysis of the minimax method. Alternative solutions to the problem of selecting three paths from the set of four paths are

$$S_1 = \{P_1, P_2, P_3\},\ S_2 = \{P_1, P_2, P_4\},$$
$$S_3 = \{P_1, P_3, P_4\},\ \text{and}\quad S_4 = \{P_2, P_3, P_4\}.$$

The objective function value of the p-dispersion model (the minimum dissimilarity) for the solutions S_1, S_2, S_3, S_4 are 8%, 8%, 14%, and 27% (and the average dissimilarity values are 20%, 29%, 41%, and 41.3%), respectively. In Section 3, it was shown that the minimax method would result in either S_1 or S_2. However, the p-dispersion solution for this example is S_4, which is superior to the

other solutions in terms of dissimilarity. Note that the optimal solution to this p-dispersion problem does not contain the shortest path; we note that this is quite likely to happen for very small p. The three other methods discussed earlier always include the shortest path in the final set. This can be viewed as another drawback of the existing methods.

The candidate set M for the p-dispersion problem can be constructed in many different ways. We experimented with two different methods: k-shortest paths, and IPM. The k-shortest path algorithm finds the shortest possible m shortest paths for set M. However, usually there is a considerable amount of similarity between many of these paths. On the other hand, it is possible to generate a relatively dissimilar candidate set using IPM by experimenting with the penalty mechanisms. Yet the average path length in such a candidate set will be higher than the average path length in a set generated via the k-shortest path algorithm. (We note that, from a purely computational perspective, IPM is much faster than the k-shortest path algorithm.)

5.1. Constructing the candidate set m using k-shortest paths

The candidate set M was constructed with $m = 100$, and then with $m = 200$, using the k-shortest paths algorithm, and the p-dispersion problem was then solved for $p = 3, 4, \ldots, 25$, for all OD pairs. Table 5 compares the p-dispersion solution with the minimax solution with the same 200 k-shortest paths for OD Pair 1.

The average length of paths (AvLe) is slightly higher in the p-dispersion solution than in the minimax solution since the p-dispersion model focuses on dissimilarity, while the minimax method considers dissimilarity as well as length. The average dissimilarity value (AvDi) of the p-dispersion

Table 5
p-dispersion solution and its comparison with minimax method for 200 k-shortest paths (O–D Pair 1; shortest length = 358)

NoP	p-dispersion solution for 200 paths			Minimax solution for 2000 paths					
	AvLe	AvDi	MiDi	AvLe		AvDi		MiDi	
				$B=2$	$B=5$	$B=2$	$B=5$	$B=2$	$B=5$
3	446.0	78.1	70.5	405.1	405.1	76.3	76.3	65.6	65.6
4	441.9	74.0	70.1	414.8	414.8	70.3	70.3	57.8	57.8
5	435.1	70.7	62.9	419.6	419.6	68.4	68.4	53.5	53.5
6	442.6	70.1	58.6	423.9	423.9	67.8	67.8	49.9	49.9
7	443.1	66.9	51.8	421.0	424.3	63.3	63.7	34.6	37.3
8	441.3	66.9	50.5	421.5	424.4	61.9	62.1	34.6	36.2
9	435.0	63.7	46.2	421.7	426.5	61.8	62.5	30.8	36.2
10	438.6	63.1	44.1	423.9	426.2	62.2	62.4	30.8	30.8
11	438.5	62.6	42.4	420.9	427.8	60.4	62.0	22.5	30.8
12	431.6	60.7	41.6	422.9	428.3	60.2	60.7	22.5	30.5
13	436.0	61.1	37.8	424.5	429.5	60.5	61.0	22.5	30.5
14	437.2	61.5	36.6	425.6	430.3	60.7	61.2	22.5	30.5
15	435.3	59.7	34.8	426.5	430.9	59.7	60.2	22.5	28.7
16	436.4	60.0	34.4	426.9	430.7	60.0	59.0	22.5	22.3
17	436.6	59.5	32.7	425.2	430.9	59.8	59.2	21.0	22.3
18	438.9	59.5	30.4	425.9	430.0	58.9	58.4	21.0	16.6
19	436.0	58.0	30.1	427.0	428.3	58.0	58.5	21.0	16.6
20	436.5	58.5	29.9	426.4	429.0	57.4	57.7	16.6	16.6
21	437.4	59.4	29.1	425.7	429.3	56.4	57.2	16.6	16.6
22	434.8	57.9	28.2	422.9	429.7	55.2	57.0	1.9	16.6
23	439.1	58.8	27.5	422.0	428.7	54.3	56.2	1.9	12.8
24	436.6	57.5	27.4	421.3	427.8	53.9	55.8	1.9	12.8
25	436.2	57.1	25.8	420.9	427.2	53.1	55.0	1.9	11.1

Table 6
Classification of IPM paths into different groups by length

NoP	O–D pair										
	1	2	3	4	5	6	7	8	9	10	11
All	500	500	500	500	500	500	500	500	500	500	500
$\leqslant 1.5\times$SL	212	270	421	355	275	46	404	123	457	127	464
$\leqslant 1.4\times$SL	116	233	408	317	255	20	352	118	439	110	451
$\leqslant 1.3\times$SL	57	167	364	213	171	9	316	67	411	99	432

solution is slightly better (higher) than that of the minimax solution. Finally, the minimum dissimilarity value (MiDi) of the p-dispersion solution (which is optimized by the p-dispersion model) is considerably better than those of the minimax method for both lower and higher values of β.

The superiority of the p-dispersion solutions to the minimax solutions is most pronounced when the number of desired paths is large. For example, for $p = 25$, the MiDi of the minimax solutions for $\beta = 1$, 2, or 3 is 1.9% whereas the MiDi of the p-dispersion solution is 23.6%. We believe that the solutions found using the two-stage greedy heuristic for the p-dispersion problem are superior to the solutions generated by the minimax model for one obvious reason: the minimax algorithm is a single-phase construction heuristic with no mechanism to avoid painting itself into a corner.

5.2. Constructing the candidate set m using IPM

We found 500 distinct paths with IPM for all OD pairs by using a low penalty factor (1%) and the original link impedance to determine the new impedance of a link. The paths were then post-processed by the p-dispersion algorithm for $p = 3, 4, \ldots, 25$ paths for each OD pair. The shortest path algorithm was used at most 7538 times for OD Pair 10 to find 500 distinct paths. Yet, the total amount of computation time required to find 500 distinct paths for all OD pairs was only around 0.3 hours.

The average dissimilarity (AvDi) values for the 500 distinct paths are around 80% for all OD pairs. As expected, the application of the p-dispersion algorithm to these 500 distinct paths resulted in very dissimilar solutions. However, the selected sets contained some very long paths (up to twice as long as the shortest path). This may be unacceptable depending on the particular application. To prevent the selection of unreasonably long paths, it was decided to eliminate from M all paths that were above a certain threshold. Three thresholds for path lengths was experimented with: 30%, 40% and 50% longer than the shortest path. In Table 6, the number of IPM-generated paths that satisfy these length constraints is stated. In general, if the dissimilarity is the only critical issue, then a very large m can be used to find the dissimilar set. However, if the length is an important factor, then a fairly tight constraint (such as 10% above the length of the shortest path) on the length of paths could be used.

Table 6 provides some interesting information on the characteristics of the OD pairs. For example, for OD Pair 6, there are only 46, 20 and 9 paths for OD Pair 6, that are at most 50%, 40% and 30% longer than the shortest path, respectively. These numbers imply that there may not be many dissimilar and relatively short paths for this pair. In contrast, for OD Pair 11, 432 of the 500 IPM paths are at most 30% longer than the shortest path. Hence, this table makes it clear that the quality (dissimilarity) of the selected set will strongly depend on the OD pair (and the network), regardless of which algorithm is used for selection.

Table 7 shows the results of the application of the p-dispersion algorithm to the IPM paths for OD Pair 9. It also gives the p-dispersion solution for the same pair with 200-shortest paths for comparison. We observed that if the 200th shortest path is considerably longer than the shortest path, then the p-dispersion solution based on IPM paths is at least as good as the solution based on k-shortest paths in terms of dissimilarity. For pairs

Table 7
p-dispersion solution for the paths found by IPM (O–D Pair 9; shortest length = 384.2)

NoP	*p*-dispersion solution for IPM														
	AvLe					AvDi					MiDi				
	All	50%	40%	30%	*k*-SP	All	50%	40%	30%	*k*-SP	All	50%	40%	30%	*k*-SP
3	624.6	529.7	498.3	453.6	396.7	99.4	99.5	97.2	94.8	79.0	98.7	98.9	94.6	93.5	73.3
4	599.1	507.7	478.6	450.0	399.3	94.8	95.7	92.6	91.4	73.2	87.8	89.2	87.7	86.1	50.6
5	534.0	494.4	473.4	449.5	398.5	94.4	94.8	92.1	90.6	66.6	76.7	87.7	86.2	82.6	37.1
6	624.3	492.7	464.8	450.3	400.3	95.2	93.9	89.1	88.6	68.4	75.4	84.3	80.4	81.0	36.6
7	514.2	474.9	462.8	447.5	400.7	89.1	89.2	85.5	86.9	65.5	71.1	77.1	76.4	75.7	33.3
8	570.1	469.4	446.6	441.0	399.6	91.6	87.9	85.2	84.0	64.2	73.1	74.4	72.2	71.2	31.8
9	559.0	465.8	450.5	447.1	400.7	90.7	86.1	83.8	83.7	63.1	68.1	72.0	70.2	71.3	28.3
10	566.8	459.6	450.2	443.5	400.5	90.3	83.4	81.4	81.5	61.0	71.3	68.1	64.1	64.4	23.2
11	554.5	469.2	445.1	444.5	400.8	88.8	85.0	81.5	79.8	62.7	63.8	63.9	62.8	60.9	21.7
12	542.1	467.5	451.3	442.6	400.8	88.3	83.3	82.1	80.6	61.5	55.9	63.3	61.3	59.9	19.7
13	555.0	455.3	449.7	443.3	401.7	88.8	83.0	81.6	79.2	60.5	56.8	59.6	58.4	57.9	17.2
14	535.5	460.1	443.2	436.9	400.6	86.3	82.9	79.4	79.4	61.6	51.8	59.4	59.1	57.5	14.7
15	548.5	459.7	441.2	441.6	400.0	87.8	82.3	79.5	78.4	61.7	51.6	58.2	57.3	55.0	14.6
16	508.8	455.1	447.8	440.4	400.4	85.4	81.7	79.5	78.0	61.1	54.2	58.2	54.9	56.7	12.0
17	535.3	457.6	440.1	437.7	400.5	86.7	82.0	77.2	78.4	60.3	52.0	57.6	53.7	50.9	11.5
18	514.7	459.0	445.9	438.8	400.5	85.6	81.6	78.2	77.7	59.2	49.7	49.8	50.6	50.4	11.2
19	526.3	456.4	449.1	438.6	400.0	85.5	81.3	77.0	77.0	59.0	44.0	50.4	50.4	49.6	11.0
20	516.1	456.3	441.0	439.7	400.1	85.8	80.7	77.4	77.2	58.1	45.3	50.1	46.8	47.9	10.9
21	523.7	459.4	445.1	437.6	400.2	85.0	80.1	77.9	76.6	58.3	47.0	49.6	46.4	45.5	10.5
22	527.8	455.3	440.9	438.4	400.3	84.0	80.1	76.8	76.8	57.9	39.1	49.5	45.8	46.1	10.5
23	512.1	452.0	446.0	437.1	399.8	83.9	79.8	77.3	76.0	57.7	37.1	47.3	45.0	42.6	8.2
24	512.3	454.9	443.7	440.8	400.7	84.4	79.5	78.1	75.5	58.8	44.2	45.0	42.5	41.7	8.2
25	495.3	456.9	443.6	437.7	400.3	82.4	80.2	76.6	74.8	58.5	44.0	43.6	43.8	41.9	8.0

where the 200th shortest path is close to the shortest path in length, the IPM-based solution is much better than the solution based on the *k*-shortest paths. The OD Pair 9 is an example for the latter case.

As the constraint on the path length is tightened, the average dissimilarity (AvDi) and the minimum dissimilarity (MiDi) values go down. The average length in the IPM-based solution with the 30% threshold is approximately 10% more than the average length in the *k*-shortest path based solution. However, the AvDi and MiDi values in this IPM-based solution are considerably better than those for the *k*-shortest path based solution.

We conclude that the application of the *p*-dispersion model to a large number of alternate paths provides good solutions for the dissimilar path problem. Generating a large number of candidate paths with IPM for the *p*-dispersion model seems to be a good alternative. However, it is not suggested that IPM-based *p*-dispersion will always result in a more dissimilar solution than *k*-shortest path-based *p*-dispersion. Suppose we are only interested in paths with lengths that are below a given threshold. If we are able to generate all feasible paths using a *k*-shortest path algorithm, then applying the *p*-dispersion algorithm to this set will produce results that are no worse than the results produced by the IPM-based *p*-dispersion method. This is because IPM is able to generate only a subset of all feasible paths. For example, the length of the 200th path for OD Pair 6 is 38.5% more than the length of the shortest path, and it is possible to select 25 paths with an average dissimilarity of 66.4% and a minimum dissimilarity of 29.5% by using these 200 paths as the candidate set. However, IPM could only find 20 paths that are at most 40% longer than the shortest path with the penalty mechanism used. The average and the minimum dissimilarity of these 20 paths are 66.5% and 2.6%, respectively.

In this case, the *k*-shortest path-based approach proved to be superior.

6. Concluding remarks

In this paper, a detailed analysis of three existing methods to generate a set of dissimilar paths is provided and computational experience with the methods is reported. It is concluded that each of these three methods has a number of drawbacks. An alternative solution technique is proposed by posing the problem as a *p*-dispersion problem. If a large number of suitable candidate paths can be generated, then the application of the *p*-dispersion model is an effective way to solve the problem.

We now discuss a possible extension of our problem. Throughout this paper, links shared by two paths were considered in computing our similarity index. Hence, we may identify two paths as "dissimilar" that have long portions which run parallel to one another, and are separated by a short distance, such as one mile. In the context of hazmat transport, these two paths might not be judged as dissimilar. A weather system that impacts one will probably impact the other one as well. Also, since these two paths will impose similar risks on residents living in areas located between these two paths, the spatial risk equity in the system cannot be improved by alternating shipments between these two paths. Hence, in the hazmat context, a different definition of similarity appears to be needed. We can define a buffer zone around a path, where the width of the zone is determined according to the intended application. When computing the similarity between two paths, the area of the intersection of the two buffer zones can be used. (If we are concerned about population exposure, the number of people living in the intersection, as opposed to the area of the intersection, in computing similarity can be used.) Given this new definition of similarity, all algorithms discussed in this paper still apply. The difficulty in implementing this extension is in the computation of the area of (or the population in) the intersection between pairs of paths. This task can be simplified by using a geographic information system (GIS). Furthermore, using a GIS one could test different road networks, as there is considerable variability both in topological and distance structure across the spectrum of all networks.

Acknowledgements

This research has been supported in part by grants from the Natural Sciences and Engineering Research Council of Canada (OGP 25481, CPG 18134). We appreciate the support of ALK Associates by allowing us to use a copy of PC*HazRoute for empirical analysis. Finally, the comments of the anonymous referees are sincerely appreciated.

References

Ahuja, R.K., Magnanti, T.L., Orlin, J.B., 1993. Network Flows: Theory, Algorithms and Applications. Prentice-Hall, Englewood Cliffs, NJ.

Akgün, V., Erkut, E., Batta, R., 1997. On finding dissimilar paths. Research report 97/3. Faculty of Business, University of Alberta, Edmonton.

Erkut, E., 1990. The discrete *p*-dispersion problem. European Journal of Operational Research 46, 48–60.

Erkut, E., Ülküsal, Y., Yeniçerioğlu, O., 1994. A comparison of *p*-dispersion heuristics. Computers Operations Research 21 (10), 1103–1113.

Erkut, E., Verter, V., forthcoming. Modeling of transport risk for hazardous materials. Operations Research.

Johnson, P.E., Joy, D.S., Clarke, D.B., Jacobi, J.M., 1992. HIGHWAY 3.01, An Enhanced Highway Routing Model: Program, Description, Methodology, and Revised User's Manual. Oak Ridge National Laboratory, ORNL/TM-12124, Oak Ridge, TN.

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

Kuby, M.J., 1987. Programming models for facility dispersion: The *p*-dispersion and maximum dispersion problems. Geographical Analysis 19 (4), 315–329.

Kuby, M., Zhongyi, X., Xiaodong, X., 1997. A minimax method for finding the *k* best differentiated paths. Geographical Analysis 29 (4), 298–313.

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

Miaou, S.P., Chin, S.M., 1991. Computing *k*-shortest path for nuclear spent fuel highway transportation. European Journal of Operational Research 53, 64–80.

Ruphail, N.M., Ranjithan, S.R., ElDessouki, W., Smith, T., Brill, E.D., 1995. A decision support system for dynamic

pre-trip route planning. Applications of advanced technologies. In: Transportation Engineering: Proceedings of The Fourth International Conference, pp. 325–329.

Shier, D.R., 1979. On algorithms for finding the k shortest paths in a network. Networks 9, 195–214.

Skiscim, C.C., Golden, B.L., 1987. Computing k-shortest path lengths in Euclidean networks. Networks 17, 341–352.

Yen, J.Y., 1971. Finding the K shortest loopless paths in a network. Management Science 17 (11), 712–716.