Geoinformatica (2013) 17:353–385
DOI 10.1007/s10707-012-0155-x

Evolutionary search for understanding movement dynamics on mixed networks

William M. Spears · Steven D. Prager

Received: 24 November 2010 / Revised: 16 November 2011 /
Accepted: 23 January 2012 / Published online: 11 April 2012

Abstract This paper describes an approach to using evolutionary algorithms for reasoning about paths through network data. The paths investigated in the context of this research are functional paths wherein the characteristics (e.g., path length, morphology, location) of the path are integral to the objective purpose of the path. Using two datasets of combined surface and road networks, the research demonstrates how an evolutionary algorithm can be used to reason about functional paths. We present the algorithm approach, the parameters and fitness function that drive the functional aspects of the path, and an approach for using the algorithm to respond to dynamic changes in the search space. The results of the search process are presented in terms of the overall success based on the response of the search to variations in the environment and through the use of an occupancy grid characterizing the overall search process. The approach offers a great deal of flexibility over more conventional heuristic path finding approaches and offers additional perspective on dynamic network analysis.

Keywords Network analysis · Evolutionary algorithms · Functional paths · Multicriteria shortest paths · Dynamic routing

1 Introduction

Research regarding shortest path problems and algorithms for solving shortest path problems has figured prominently in a variety of application domains since Dijkstra's

W. M. Spears
Swarmotics, LLC, Laramie, WY, USA
e-mail: wspears@swarmotics.com

S. D. Prager (✉)
University of Wyoming, Laramie, WY, USA
e-mail: sdprager@uwyo.edu

algorithm was published in 1959 [12]. Dijkstra's algorithm, A* [22], and related algorithms for solving shortest path problems [3, 13] are very efficient (polynomial time in the case of Dijkstra's algorithm), but they are limited by a number of constraints depending on the algorithm. Perhaps most significantly, these techniques often require very specific tuning and selection with respect to the problems to which they are applied [42]. In many instances, however, the combination of a novel problem and mixed network structures are such that the "best" solution heuristic may be ambiguous or unknown.

Recent research demonstrates the viability of evolutionary algorithms (EAs) for shortest path (SP) problems where the problem formulation results in algorithmic and approximation schemes that are prohibitively complicated [23, 29, 35]. In [35], we characterize one example of this class of problems and how it arises when the morphology of the path requires consideration as part of the optimization process. The issue, highlighted in [25], is that optimal paths may not be comprised of optimal subpaths. This is an important consideration. Unlike Dijkstra's algorithm and other dynamic programming techniques that rely on the Principle of Optimality, our approach makes no such assumption. Problems in which paths may not be comprised of optimal subpaths are often difficult to address with typical single objective shortest path problem (SOSP) approaches because optimization must not only consider cumulative edge cost but also morphological and other constraints placed on the manner in which the path is allowed to use the underlying network. Using a priority-based evolutionary algorithm (EA), we present an approach that allows us to search the network space in a manner that considers the interplay of both path length and path morphology in discovering high quality routes similar to the potential optima.

Another interesting challenge arises with SOSP algorithms when the graph on which they are operating is dynamic. As with many of the specializations of traditional SOSP algorithms to multi-objective shortest path problems (MOSPs), the specialization of SOSP algorithms to time varying problems tends to be specific to the manner in which time varies on the network in question [31]. One advantage of time-varying graphs is that time is directed. This is leveraged in [9] where shorter hyperpaths are found (notably, in linear time) on time-expanded hypergraphs. Again, however, the specialization is specific to the problem and the hypergraph approach is intractable if constraints on route structure are imposed [9].

Evolutionary algorithms (EAs) offer an alternative to the highly specialized implementations of typically used heuristics for solving both dynamic and multiobjective problems. In applying EAs to problems on dynamic and stochastic networks, Davies and Lingras [10] demonstrate that evolutionary algorithms are sufficiently flexible so as to allow reparameterization of the optimization process as information changes within the search space. Evolutionary approaches offer an additional characteristic in contrast to many traditional SOSP and MOSP algorithms in that they can be used to present multiple candidate solutions. For example, Lup and Srinivasan [26] demonstrate an approach to routing that considers both shortest paths and shortest travel time. In this case, not only is the environment dynamic, but the user is presented with multiple potential optima given the search space and is thus equipped to make the best decision regarding their route given the current road conditions.

The idea that the discovery of potentially optimal paths in dynamic environments is a dynamic process is the motivation for this research. In this paper we present the

idea that certain dynamic path optimization problems require a shift from thinking in terms of shortest path analysis as a single solution problem. Instead, we propose that these problems are more about the use of the search space as it relates to potential solution optima. Given that the search space is fundamentally geographic, this implies that the search results are about the use of geographic space and the manner in which geographic space and geographic dynamics constrain or create potential optima. We thus see the "answer" to these types of problems as a *collection of multiple potential optima.* This idea becomes particularly important with the recognition that path finding often occurs in dynamic environments and that this has the potential to significantly modify the manner in which the search space (the network) is used.

With this research, we investigate how evolutionary approaches can be used to understand the set of potential optima associated with traversal of a mixed network space consisting of both transport features and terrain. We treat the environment as dynamic through the addition of "avoidance points" during the search process and illustrate how the evolutionary algorithm is adaptive in response to the changing environment. The potential applications of this approach range from understanding how an individual might move through space to avoid detection to understanding how animals might respond to obstacles present in their migration corridors [2]. As mentioned above, an important aspect of this research is that the approach results in a better understanding of the search space rather than a single, heuristically driven answer. This understanding can, in turn, be used in the creation of more efficient and problem-specific variations on well known SP algorithms.

In the following sections, we begin by addressing issues associated with modeling movement and the use of evolutionary approaches for modeling both static and dynamic geographic environments. We then present the problem class associated with this research and enumerate the general problem as well as specific examples. In turn, we address the unique characteristics of our approach and the manner in which we collect information regarding the algorithm performance during the course of the search process. Finally, we present the experimental results and sample runs on real data. We then conclude with comments on the potential of this approach to a variety of problems and directions for future research.

2 Modeling dynamic phenomena

Modeling geographic dynamics is a challenging task at the forefront of research in GIScience [41]. When the spectrum of models of dynamic phenomena are examined, the different approaches to understanding and modeling movement and traversal of geographic space generally fall into four categories. These include consideration of constrained versus unconstrained analytic spaces and the difference between understanding movement and traversal of space in a static context versus in a dynamic context.

In addition to understanding the different categories of modeling dynamic phenomena in geographic space, it is also useful to consider the representational approaches that can be used to populate the modeling environment. In that representation and analysis are inextricably linked, we then address the analytical approaches applied to these problems and, specifically, the role of evolutionary algorithms for solving large problems associated with dynamic environments.

2.1 Model classes

Differentiating constrained and unconstrained analytic spaces is relatively straightforward. Constrained spaces are typically limited to movement along a network or graph and the set of potential positions are limited to positions along edges and nodes in the network. If the network is geographically embedded (e.g., a road network), then the set of potential geographic positions is limited by where the network exists in geographic space. In contrast, unconstrained spaces are typically characterized with a continuous raster representation. Because of the continuous nature, position in the continuous space is limited only by the characteristics of the quanta defining the space in relation to the rules associated with traversal of the space.

The distinction between static and dynamic models of movement or traversal through space may be less intuitive than the distinction between constrained and unconstrained spaces. Static models tend to be based on single solution heuristics that globally consider the entire space and estimate paths through that space based on the costs for movement characterized by the space and the algorithm (often Dijkstra's) for minimizing total path cost. The models are static in that a) the costs for surface traversal are predetermined and unchanging, and b) the algorithms for evaluating potential paths through the surface are state driven and do not consider internal feedback or changing data. In contrast, dynamic models of movement not only take into account the cost of traversing the analytic space, but can also take into account changes in the values both of that space and in terms of internal feedbacks associated with elements within the model itself. Given the contexts of constrained and unconstrained spaces and static and dynamic models of movement, four possible combinations arise.

2.1.1 Constrained-static

The idea that a network, geographically embedded or otherwise, is a constrained representation of space is well established and has informed a variety of adaptations of metrics from continuous space to network-constrained space [30, 39, 40].

Static, network-constrained movement models are commonplace with many network routing problems falling into this category [14, 28]. While these latter models do account for potentially different results depending on input parameters to the models (e.g., time of day or next bus arrival) [33], the computation is fundamentally state driven and deterministic in the sense that given the same input parameters, the same results will be imputed.

2.1.2 Unconstrained-static

A common GIS analysis is where minimum cost paths are computed through a cost surface where each quanta or unit in the cost surface has impedance or cost associated with its traversal [18]. Analyses such as corridor analyses for exploring alternative power line siting locations [5] to understanding elephant movement between refuges [32] have long been part of this family of operations. Cost surfaces and related analyses are flexible in that their representation is limited only by the ability of the user to effectively combine costs in analytically meaningful ways. For example, in [1], the authors combine a variety of landscape variables and develop

an approach for characterizing the "effective cost" of landscape traversal in an ecological context.

As with constrained-static models, unconstrained-static models are deterministic and state driven. While the unconstrained representation allows for a greater set of possible locations being identified by the model, these models typically do not have any internal feedback mechanisms or method for responding to changes in conditions characterizing the analytical space.

2.1.3 Constrained-dynamic

The combined advent of readily available GPS-based location information and increased computational capacity led to the expansion of modeling to include approaches that are more explicitly dynamic. Dynamic models facilitate a greatly increased understanding of the ramifications of early work on spatio-temporal movements of individuals in space [19]. As data streams and requisite database technologies proliferated, many of Hägerstrand's early ideas were extended and used to inform understanding of a variety of moving entities within a database context. These new approaches quickly exploited the idea that networks offered an important constraint for managing dynamic datasets [34] and supporting effective minimization of uncertainty given that objects constrained to networks had fewer potential locations in which they could exist within a given amount of time [38].

Other constrained-dynamic models focus on using the network as a conduit over which the movement of individual agents can be modeled and analyzed. In [7] and [8], the authors illustrate how network-constrained microsimulation is a useful tool for planning evacuation scenarios. In the case of these models, the network constraint serves to reduce the number of potential interactions between agents and simultaneously provides a clear relationship to real-world geography. In contrast to the evacuation planning models, Colizza et al. [6] and Shaw et al. [36] demonstrate how the network constraints of the international and national air transport network can be used as a backdrop for modeling spread of disease over time.

Constrained-dynamic models of movement are very powerful inasmuch as the constraints associated with the network representation are realistic to the modeling scenario. When the constraints are realistic to the problem, they help to significantly bound the search space. In addition, given that network structures have distinct topological characteristics, these topological characteristics can often be exploited as predictors for understanding the process the model supports (e.g., see [27] for an example on the use of the network metric of information centrality for understanding overall network dynamics).

2.1.4 Unconstrained-dynamic

The most flexible approach to modeling dynamic phenomena is an unconstrained analytic space used in the context of dynamic modeling problems. It is important to clarify that *unconstrained* does not imply an isotropic surface with no impedance, but rather that the analytic space and corresponding representation is more field-like than object or network-like.

One common example of unconstrained dynamic problems are agent-based models wherein individual agents traverse field data. The rules for each agent determine the decisions made by the agent, and usually a stochastic component is introduced

to vary agent behavior within some appropriate bounds. As illustrated in [4], the agents interact with other agents as well as the environment, and the cumulative set of interactions is used to understand how agents (and by extension the real-world entities they proxy) learn and respond to changes in their environment.

Alternatively, agent behavior can be used to understand how different portions of the landscape become more or less important depending on the rules governing agent behavior. In contrast to the above example where the focus is on understanding how the agents accumulate knowledge over time, Hargrove et al. [21] illustrates how agents can be used to travel the unconstrained landscape for purposes of finding high intensity travel corridors. In this latter example, the agent analyzes the environment and moves through the environment following a certain goal-seeking behavior. The cumulative paths of the agents provide insight as to how the agents (and, again, the real-world entities they proxy) might utilize different environmental configurations.

There are numerous complexities associated with the unconstrained-dynamic class of models. These complexities range from issues regarding representation of the analytic space to the nature and representation of the process and characteristics of the dynamic phenomenon.

2.2 Representation and analysis in non-stationary environments

Given the spectrum of problem classes presented in the previous section, consideration of representation of both the analytic space and process model is a very important part of the modeling process. This consideration is particularly important when considering dynamic or non-stationary data.

The intertwined nature of representation and analysis of non-stationary environments underscores the notion that data structures must be tailored to support specific types of analyses. For example, the addition of time dependence to a representation of a transportation network requires development of both an expressive approach to capture the time varying nature of the environment as well as a data structure that can be efficiently queried and analyzed [17].

The relationship between the analytic space and the analytic process is particularly emphasized when evolutionary algorithms are used to model the dynamics associated with the search space. In our previous work [35], we illustrate how functional paths (paths which embody certain behavioral preferences) can be discovered using evolutionary approaches. The resultant functional paths are a combination of the graph representation, the manner in which the graph is used to initialize the evolutionary algorithm, and the fitness function built into the algorithm itself. Similarly, in [10], the authors illustrate how rerouting on graphs can be performed with a genetic algorithm that, in essence, mimics the decision behavior of an individual moving through space. In this example, the semi-stochastic nature of the individual's decision making process in traffic conditions is brought to bear in the initial population of the GA. As traffic conditions change, the GA evolves, and the set of navigation considerations change accordingly. In both the case of functional paths and navigating through traffic, the results of the evolutionary-based analyses are dependent on the combination of the representation of the analytic space and the behaviors embodied in the analysis process itself.

These latter examples illustrate that while the classes of problems identified in Section 2.1 serve as a useful framework for positioning a research question, certain

modeling problems can be thought of as "composite" problems. These composite problems share elements from two or more classes and, consequently, often introduce new challenges to consider in terms of both representation and analysis.

3 Evolutionary approaches for modeling movement

For this research, we further refine our approach to understanding functional paths as illustrated in [35]. We address the motivation for use of an evolutionary approach, representational issues, and the dynamic nature of the model presented. We then provide an overview of the technique and discuss its relationship with more conventional approaches to shortest path routing.

3.1 Problem specification and motivation for using EAs

There are four factors that differentiate our problem from other shortest path problems and support our motivation for the use of evolutionary algorithms.

First, our research is fundamentally about understanding the use of a network space for a specific purpose. The combination of the representation of space and the evolutionary algorithm capture this purpose and the corresponding behaviors. In our motivating examples, the use of space and corresponding behaviors might mimic an individual trying to surreptitiously cross a border using road and off-road travel, or migrating animals trying to maintain paths through viable habitat while avoiding roads or other anthropogenic features. The stochastic aspects of EAs allows for a non-deterministic exploration of the search space that can be useful in a search process where the exact basis for the entity behavior guiding the use of the space is unknown.

The introduction of a stochastic component in the analysis process further underscores the idea that the analysis of movement and potential behaviors cannot be reduced to a single answer. As such, we treat the evolutionary search process as a means to enumerate the search space. The more often locations within the search space are visited, the more likely those locations are important to the behavior and functional path in question.

Next, given our philosophical shift from treating the optimization as a single solution problem to a mechanism supporting the enumeration of the search space, it also becomes possible to leverage the evolutionary process to account for dynamics or changes within the search space itself. As with [10], we take advantage of the ability of the EA to adapt and respond to new information in the search space. The combination of the enumeration of the search space and the evolutionary adaptation to new information provides substantial insight on how behaviors might change given specific interventions.

Finally, as with our earlier work, we see the behavioral aspect of the path search process as exceeding the capabilities of greedy-based search algorithms. Unlike single solution optima, the behavioral element of the search requires simultaneous consideration of characteristics associated with the whole route as well as those associated with more local subsets of the route. In that each individual created by the EA is a fully specified path, both its global characteristics and its local characteristics can be evaluated.

3.2 Representation of analytic space

The evolutionary algorithm we propose is searching the analytic space for potential paths. We thus elect to treat the representation of the analytic space generically as a graph that may be composed of one or more system subgraphs. By system, we refer to the individual networks that comprise an interconnected network space intended to serve a specific functional purpose (e.g., a multimodal transport network). With the graph structure and the EA, we can thus reason across a single network or a network of networks that serve to collectively represent analytic space.

For each network, let $G_i = (V_i, E_i, W_i)$ define an undirected graph characterizing the network. Minimally, vertices must be spatially embedded (e.g., as in an air transportation network or social network), but often vertices and edges are both spatially embedded (e.g., as in a road network). Vertices V_i thus represent n intersections or end points (airports, road intersections) in graph i. Edges E_i are a set of m undirected segments that each connect two vertices.

For each $G_x \in G_{1\ldots z}$, E_x and W_x must be cardinal, and for each $e \in E_x$, $w \in W_x$ is the weight or cost associated with the individual of the edge. For the present representation, the domain of W is not allowed to vary across networks and the units of measure must therefore be consistent across G networks. There is also an ordered set of k waypoints $(p_1, \ldots, p_k)$. Path P through G is defined as the set of vertices that start at the source node p_1, hit the intermediary waypoints in order, and end at the destination node p_k.

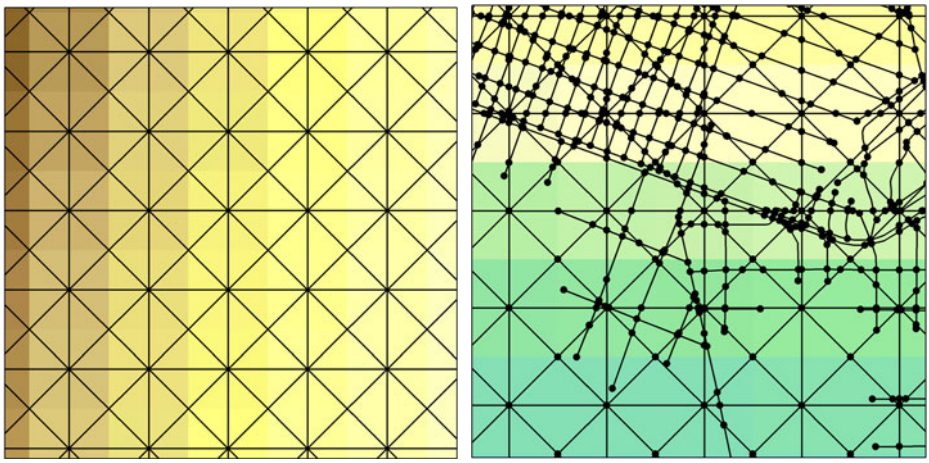
In keeping with the unconstrained class of models described in Section 2.1, an approach is also required to represent continuous or field data in network terms. An interesting aspect of continuous representations is that they are, in fact, reduced to a network representation for analytic purposes. For any given cell in a raster tessellation, that cell is functionally connected to its eight immediate neighbors. By substituting the network of neighbors, a graph-based representation can easily be constructed (Fig. 1a).

Once a surface tessellation is converted to a graph, it may be combined with any other graph. Two approaches exist to combine multiple networks. The first approach assumes $G \supset G_{1\ldots z}$, $v \subset V$, such that subset v are vertices that can be connected by a new set of edges e. The alternative approach assumes equivalent planar embedding across networks (i.e., the networks are embedded in the same Euclidian space and share a common coordinate system). In this approach, the networks are topologically planarized in such a manner that any location where an edge from one network intersects an edge from another network, a new node is added and the topology between the individual subgraphs becomes continuous based on the shared geography (Fig. 1b).

In the examples that follow, we demonstrate both synthetic and real datasets. For purposes of the present research we use the planarized approach to integrating data as described above.

3.3 A dynamic, evolutionary approach

In contrast to algorithms that result in single solutions to problems, evolutionary algorithms manage a population of individuals, with each individual representing a potential solution. In each "generation" the most fit individuals are allowed to



(a) (b)

Fig. 1 The relationship between raster and network representations. Note that points represent intersections of edges in each network and thus nodes in the graph representation

have "children" through perturbation processes such as mutation and recombination. Since the population size is constant, the most fit individuals are maintained while less fit individuals are discarded. For each successive generation, the population "evolves" to incorporate new characteristics that increase the overall fitness of the individuals. The key design decisions in the use of EAs involve the form of representation of the individuals, the particular perturbation operators used, and the method for determining the fitness of a given individual. A nice feature of an EA is that the initial population can be "seeded" by using a priori domain knowledge. We take advantage of this feature in our work.

In this research, we build on our earlier work wherein the EA evaluates fitness based on three factors: length, avoidance of obvious corridors, and proximity to specific waypoints [35]. In addition to emphasizing paths through networks of networks, we depart from our earlier work in three other ways. First, we generalize both our representation and the initialization of the initial population. Second, rather than avoiding specific waypoints along a defined path, we visit waypoints but include additional "avoidance points" as a dynamic input that can be added during the evaluation process. Importantly, while we recognize that other algorithmic approaches may be suitable for solving similar problems, we are specifically interested in both the flexibility of the EA and the manner in which the EA search serves to explore and enumerate the network space. We capture the course of this exploration as the search result surface. In doing so, the search results offer a probabilistic perspective on how the search space might be used in the context of the specific problem.

3.3.1 Representation

In evolutionary algorithms (EAs), a series of "genes" are combined to build an individual. For this research, each gene represents a node in the combined network.

The alleles (values of the genes) of an individual *encode* a path through the network. In that different paths may have different numbers of nodes, the individuals encode variable length representations of potential paths. While there are numerous approaches for representing individuals ranging from permutation representations to index encoding representations (where the value j of gene i indicates that the path contains the edge (i, j)), many became computationally infeasible with graphs of more than 100 nodes.

In [35] we adopted a node priority representation as described by [15, 16]. An elegant aspect of this representation is that a fixed-length representation can produce paths of variable length. Coupled with a breadth-first search (BFS) to initialize the population of individuals with feasible candidate paths, this approach allowed us to handle graphs with hundreds of thousands (and millions) of nodes. One drawback of this approach is that because the BFS is not reasoning about edge weights, the initial population of the EA is seeded by the number of vertices (hops) in a path rather than distance as a function of edge weight information. This works well when the variance in edge weights is minimal. However, once the variance increases sufficiently, hop distance biases the search towards favoring long hops. In networks with very different node densities (urban versus rural versus a surface network, for example), the tendency is to avoid paths that pass through the high node density areas.

In order to rectify this situation, we generalized our algorithm to include edge weights. Figure 2 shows an undirected weighted graph with seven nodes. The edge weights are indicated with the smaller font numbers. Now, however, the initial node priorities (indicated with the larger font numbers) are the shortest weighted path cost to the destination D. Let P_i denote the priority of node i (not to be confused with the waypoint p_i). With our earlier representation, if the path is currently at node i, then the next node j is the neighbor with the highest priority. In our new representation we use a novel decoding mechanism—if at node i choose the node $argmin_{j \in N_i}\{|P_i - (e_{i,j} + P_j)|\}$, where N_i are the neighbors of node i and $e_{i,j}$ represents the edge weight from node i to j. For example, suppose the path is currently at node V in Fig. 2. Then node Y is next because $P_V - (e_{V,Y} + P_Y) = 6 - (1 + 5) = 0$.

In contrast to the more conventional priority representation, our generalized approached prioritizes nodes in a manner that minimizes $|P_i - (e_{i,j} + P_j)|$, as opposed to looking for a value of zero. The reason for this decision will be made clear in the next subsection. Although the initialization procedure produces values of zero, once

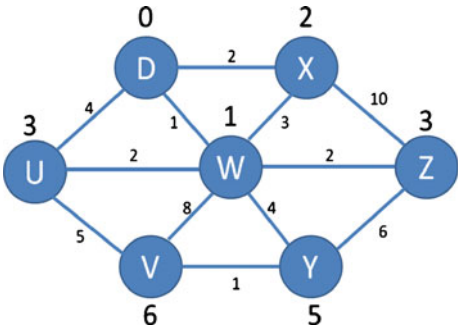


Fig. 2 Example of our generalized node priority representation

the mutation operator changes the individual, this is unlikely to occur again, and the best match is chosen.

Also in contrast to our original representation, real-valued edge weights are used in lieu of simple hop counts. Because weights can be real-valued, our representation is generalized such that the EA individual consists of $k-1$ sets of real values of length $|V|$ (because there are k waypoints denoted $p_1, \ldots, p_k$). Whereas the first representation encoded the unweighted shortest path tree, this representation encodes the weighted shortest path tree and can be computed in $O(|E|log|V|)$ time. The shortest path algorithm is run $k-1$ times to initialize the node priorities. First it is run with the assumption that p_2 is the first destination. This creates the vertex priorities for the first leg. Then it is run for each succeeding waypoint. Finally, it is run for the destination node p_k. The algorithm that decodes the EA individual to produce the feasible path switches from leg to leg as each waypoint is hit.

3.4 Mutation

A very important aspect of the evolutionary process is the manner in which the individual genes are perturbed. In the context of path discovery, the perturbation is required to preserve the essential quality that any new path *must* reach its destination while hitting each succeeding waypoint in turn.

Though evolutionary algorithms often employ some form of recombination [37], in the context of path optimization, we find that mutation offers a sufficient vehicle to drive the evolutionary process. The mutation process itself is quite simple. Recall that the initial population contains individuals that can be decoded to paths that traverse all waypoints and reach the destination. Mutation chooses one of the nodes on a path with uniform probability. If that node has only one adjacent node, it must be the source or destination. Otherwise the node has more than one adjacent node. The mutation process then randomly selects two different adjacent nodes with uniform probability and swaps the priorities associated with those two adjacent nodes. If this perturbation breaks the path (creating a dead end while not reaching the destination, or skipping a waypoint), the mutation is rejected and the priorities are returned to their previous state. Otherwise, the mutation is accepted. As a consequence the EA individuals always remain feasible.

As an example, consider Fig. 2 again. Suppose node V is chosen. Then the priorities of nodes W and Y could be swapped. But now the next node from V is no longer Y, but is in fact U, because that is the node that minimizes $|P_i - (e_{i,j} + P_j)|$. Any mutations that yield paths that do not reach the destination and all waypoints in order are rejected.

Interestingly, we experimented with other mutation operations that utilize more domain knowledge. None worked as well as the simple mutation algorithm described above. Alternatively, the various λ-opt operators described in [24] for TSP would appear to be potential candidates for mutation. However, there are two difficulties. First, while our representation semantically encodes a path, is not syntactically a path. The implementation of the λ-opt operators are thus problematic (i.e., at the node priority level, not the path level). Second, the operator is likely to produce infeasible paths, due to the sparseness of our graphs (as opposed to the fully connected TSP graphs). This problem only gets worse as path length increases.

3.5 Incorporating dynamics

In this paper the fitness landscape is modified each EA generation via the cumulative addition of the aforementioned avoidance points. It is useful to visualize each EA individual as representing the behavior of a moving agent (such as an individual trying to cross a border, or an animal trying to move through its habitat). In this context, avoidance points are nodes that individuals attempt to avoid and thus influence the morphology of the path. In contrast to our earlier work [35], the environment is now non-stationary and the addition of each avoidance point requires re-computation of individual fitness for that generation. By adding a new avoidance point during each generation, the population of individuals are under continual pressure to adapt to the changing environment.

The rate at which avoidance points are added to the representation causes the fitness landscape to change quite quickly. A consequence of this rapid change is that the mutation operator required an additional modification. Instead of being applied only once per individual, mutation is now applied far more often, namely, ten times the number of nodes in the feasible path encoded by that individual. This allows each individual to react more quickly to the changing environment by increasing the range of possible states of fitness each individual can achieve in response to the avoidance point. Also, longer paths are mutated more often, reflecting the increased complexity of the path. The factor of ten was arrived at empirically, over numerous problems, but can be modified as dictated by the problem.

3.6 Developing the fitness function

To calculate the fitness of an individual, the individual is first decoded into a path. The fitness is determined by the total edge weight of the path combined with functions characterizing the *avoidance node* and *curvature* preferences (both terms will be defined below). Since total edge length is a quantity that should be minimized, we utilize a minimizing EA. To be consistent, both avoidance and curvature are defined as penalty functions that are also minimized during the search process.

3.6.1 Total edge cost

The total edge length L is simply the sum of the cost of the edges along the nodes of the path. Due to the weighted shortest path pre-processing step and the restricted mutation operator, the path is always guaranteed to reach the destination, while hitting each waypoint in succession.

3.6.2 Computing the avoidance node penalty

Avoidance regions are defined by a region of radius d centered at the avoidance node. Beyond this radius the avoidance node has no effect on the fitness of a path; the avoidance region can thus be thought of in terms of the area an animal would avoid around an anthropogenic feature, the area that might be visible from a guard station, or the effective area of a sensor. The avoidance penalty $Avoid$ is defined with respect to the distance of a vertex in path P to each avoidance node (assuming the

vertex is in the avoidance region). The following code is executed for each node in the path:

Listing 1 Computation of avoidance node penalty.

```
for (i = 1; i < number_of_avoidance_nodes; i++) {
  if (distance(path_node, avoidance_node[i]) < d) {
   avoidance_penalty = avoidance_penalty +
    (d - distance(path_node, avoidance_node[i]));
  }
}
```

If a vertex is too close to an avoidance node, a penalty that is proportional to its closeness is computed. In that the number of avoidance nodes increases with each generation, both the environment and individual fitness are also continuously changing.

For purposes of the present research, we elect to use Euclidean distance for the avoidance node penalty. This relies on the assumption that the network is spatially embedded and that the network is reasonably dense. For example, border patrol systems include acoustic and seismic sensors that are spatial in nature. Future versions of this method shall utilize network distance, which shall also serve to extend proximity concepts into non-spatial networks.

3.6.3 Managing the curvature penalty

As defined in [35], curvature is related to the shape of a plane curve. A straight line has no curvature. In order to avoid obvious corridors in a graph we wanted a mechanism for defining the desired curvature between two waypoints, and to measure the curvature penalty when a node on the path does not match the desired curvature. In this paper we will generalize our notion of curvature to network space rather than Euclidean space, but a description of the Euclidean version is necessary for clarity.

We use as inspiration the equation for an ellipse. Given two waypoints, we consider these to be the foci p_i and p_{i+1} of an ellipse, with distance F between them. However, an ellipse does not include a straight line from p_i to p_{i+1}, which potentially represents the shortest path from p_i to p_{i+1}. As a consequence we define a slightly different class of curves (where r_1 and r_2 are the distances of a point from p_i and p_{i+1} respectively):

$$r_1 + r_2 = kF$$

This equation is easy to understand intuitively. The constant k is constrained to be greater or equal to one ($k \geq 1$). When $k = 1$ this represents the straight line between the two foci. The interpretation of k is somewhat similar to that of a spring constant. As k increases the curve bows out more, due to tension applied to the curve.

One difficulty, however, is that in a simple ellipse model where the two waypoints are foci of the ellipse, a curve can extend beyond the bounds of the foci. We further constrain our class of curves with $r_1 \leq F$ and $r_2 \leq F$. As a consequence $r_1 + r_2 \leq 2F$, and $1 \leq k \leq 2$. The desired curve is the perimeter of the region shown in black in Fig. 3.

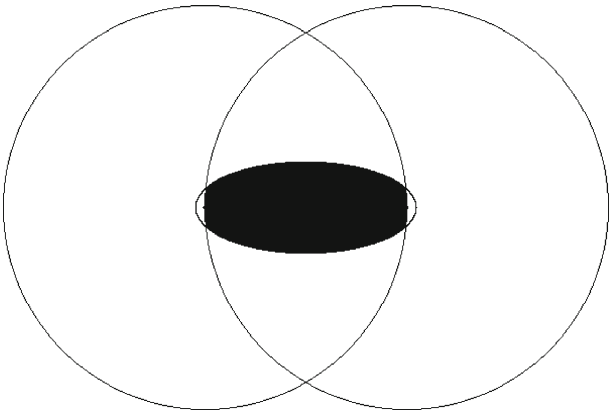


Fig. 3 Avoidance region with $F = 200$ and $k = 1.1$

If a vertex lies within the black region in Fig. 3, a penalty is applied. Consider the path from waypoint to waypoint, with nodes A, B, C, and D, as shown in Fig. 4. What penalty should be assigned to node A, since it is within the black region? Node A is closest to the left circular arc, so a reasonable penalty is $F - r_2$. Following standard conventions for ellipses, r_2 is the distance of the node to the right-most foci (waypoint). If node A moves to lie on the left circular arc, the penalty goes to 0. In a similar fashion, the penalty for node D is $F - r_1$. Node B is closest to the top portion of the curve, and a reasonable penalty is $kF - (r_1 + r_2)$. Note that if B moves to the curve perimeter, the penalty goes to 0. Finally, node C is outside the desired curve perimeter and hence there is no penalty. Putting this all together, the penalty for a vertex is $min\,(F - r_1, F - r_2, kF - (r_1 + r_2))$, if $(r_1 \leq F) \wedge (r_2 \leq F) \wedge (r_1 + r_2 \leq kF)$.

The penalty for a path between two waypoints is thus the sum of the penalties for each vertex along that path, divided by the number of vertices along the path (in order to normalize the metric). The curvature penalty *Curve* for a path with multiple waypoints is simply the average of the avoidance penalties for each leg.

One important issue is that minimizing the avoidance penalty often conflicts directly with attempts to minimize total edge length. In other words, to "bow out" more requires a longer path. This is the reason we assign a penalty of 0 to node C. If the EA can pull that node back in a bit, while still remaining outside the

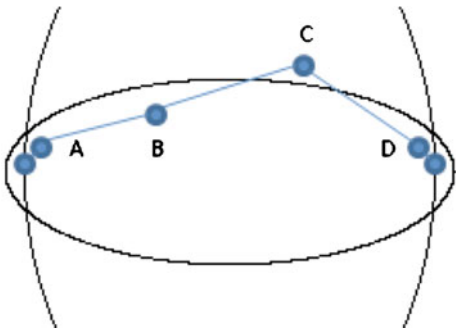


Fig. 4 A path from waypoint to waypoint

curved region, path length will diminish while curvature penalty will remain the same. Ultimately we want the shortest path that lies just outside the curve perimeter.

Though the above emphasizes Euclidean distance, the approach makes no assumptions about how distances are calculated. In this paper r_1, r_2, and F are all computed using network distance, not Euclidean distance. Given this, what do we expect when k is increased? First, barring any network information to the contrary, we expect paths to increase both in terms of network length and in terms of the number of nodes. More important, however, in the context of this paper, is how k affects the search process itself. Intuition suggests that not only are more nodes "hit" during the search process, but that the distribution of node hits becomes more uniform as k increases. In turn, k essentially describes the amount of exploration that an agent can take—higher k implies higher exploration. These concepts will be defined precisely later in the paper.

3.6.4 Handling multiple waypoints

When there are multiple legs in the path, a mechanism is required for switching from one leg in the path to the next. This is done when a vertex on a path is the destination (waypoint) for that leg. Then that node becomes the source for the next leg of the path, and the next set of node priorities are utilized in decoding this next leg. This is equivalent to switching from one shortest path landscape to the next. Throughout this paper, waypoint locations are predetermined and do not vary during the course of the analysis.

3.6.5 Calculating fitness

To calculate the final fitness of an individual (path) we take the weighted sum of the three individual components:

$$Fitness = \alpha L + \beta Curve + \gamma Avoid \tag{1}$$

For this paper $\alpha = 1$, $\beta = 1000$, and $\gamma = 10000$. Since we are most interested in avoidance, it is given the highest weight. Curvature also has high weight, in order to help avoidance (longer paths may be able to avoid nodes more easily). Path length becomes important when avoidance and curvature have been adequately solved by the EA. Finally, the avoidance radius is $d = 10$.

3.7 Performance metrics

As previously stated, the shift to a dynamic problem space requires a shift in thinking of optimization in terms of a single solution to thinking in terms of the range of possible solutions. A dynamic search space necessarily entails the possibility that the "optima" change as new information is considered.

To this end, four criteria serve as metrics for overall algorithm performance. The first is simply the number of nodes hit by feasible paths during the course of a run of the EA. The second is the total path length (edge cost) of the best individual found by the EA. The third is the avoidance penalty. Finally, the fourth criteria complements the first, by indicating the amount to which nodes within the graph serve as critical components of complete paths. This last criteria is explained further below.

3.7.1 Occupancy-based characterization of search

As stated above, one metric of interest is the total number of nodes hit during the whole search process (i.e., the number of nodes hit by the agents (individuals) in the EA population). However, the distribution of nodes is also useful to monitor.

If we monitor the total number of times that each node is hit, and divide by the total number of hits over all nodes, the result is a probability distribution representing the overall occupancy of the nodes during the search. To compare this distribution with the uniform distribution, we first considered using the Kullback–Leibler divergence. Although the lower bound is 0.0, an upper bound is not defined, and it is not a true metric since it is a non-symmetric measure. Furthermore, it is not defined if the node distribution contains a zero (i.e., a node has never been hit). An alternative is the "earth mover's distance", which is a measure of distance between two probability distributions. Unfortunately, computation involves use of the *Hungarian algorithm*, which is $O(N^3)$. For the number of nodes used in our experiments, computation of this metric is not feasible.

Instead, we used a simple Euclidean distance metric. Consider two probability distributions P and Q. The distance metric is $\sqrt{\sum_{i=1}^{N}(P_i - Q_i)^2}$, where N is the number of nodes in the graph. Let Q be the uniform distribution. If the node distribution P is uniform, then the Euclidean distance is zero. This represents an agent that visits all nodes equally often. The maximum value of the metric can be obtained by examining the distance of the "degenerate distribution" (which has probability one at only one value) from the uniform distribution. In this case $Q = (1/N, 1/N, \ldots, 1/N)$, and $P = (0, \ldots, 0, 1, 0, \ldots, 0)$. The distance metric is:

$$\sqrt{\left(\left(\frac{1}{N}\right)^2 (N-1) + \left(1 - \frac{1}{N}\right)^2\right)}$$

Simplifying yields $\sqrt{\frac{N-1}{N}}$, which quickly approaches one as N increases. This represents an agent that never moves. Values closer to the maximum indicate that the EA is focusing strongly on a small subset of nodes. Values closer to zero indicate that the EA is focusing attention more uniformly over a larger set of nodes.

3.7.2 Influence of the curvature constant k

We are also interested in the curvature constant k, as it serves to indicate some behavioral characteristics of an agent. When $k = 1.0$, the agent follows the most direct path from source to destination. The agent does no exploration. As k increases, the agent becomes more explorative, and hence more indirect. This reflects an agent more concerned with being evasive, and not being caught by a sentry (or sensor) at one of the avoidance nodes. In this situation the total path length becomes less important to the agent. Hence, we will focus on the change in performance metrics as k changes.

4 Application and results

In our earlier work on single-network analysis [35], we developed simple lattice datasets to support experimentation and used the Colorado road network provided

by DIMACS [11] (with 435,666 vertices and 1,057,066 edges) to demonstrate the effectiveness of the approach on real-world data. Here, we emphasize datasets that integrate multiple networks using the data integration process discussed in Section 3.2. As with the earlier work, two distinct datasets are used to support the experiments in this section.

The first is a dataset created using a synthetically generated impedance surface with an arbitrary set of edges representing low impedance "roads" through the surface. The second dataset is based on a combination of real-world road data from the El Paso-Juarez border region on the US-Mexico border combined with an "exposure" surface that serves as a proxy for risk of discovery associated with overland versus road-based travel.

In the following sections we describe the two datasets, the experimental design and the results of the experimental runs on each dataset.

4.1 Synthetic dataset

The purpose of the synthetic dataset is to provide a basis for testing the ability of the EA to traverse the network space taking into account the edge costs (i.e., the penalty for overland versus on road travel) while simultaneously taking into account the other optimization criteria (route morphology and avoidance penalties).

The synthetic dataset was created by generating a kernel density surface from a series of points organized in space to have areas of varying high and low density. Once the kernel density surface was created, the surface was converted to its corresponding surface network where density values become a multiplier to weight the Euclidian distances associated with the surface network edges (using the procedure described in Section 3.2).

Following the creation of the surface network, a series of synthetic "roads" were digitized for the same geographic area. The roads extended slightly beyond the edges of the terrain network to provide for locations to begin and end the routes associated with the experimental scenarios. The intent of the road configuration was to provide for a variety of routing scenarios involving traversal of roads as well as through high and low impedance areas.

The synthetic roads and the surface network were integrated using the previously mentioned topological join process. The end result is a dataset containing 44,530 nodes connected by 130,776 edges. These edges and nodes are then post-processed and converted to a text-based adjacency list representation that can be read by the EA data input routines. The synthetic dataset is shown in Fig. 5.

4.1.1 Experimental runs using the synthetic dataset

For the synthetic dataset we chose four specific problems, as shown in Fig. 6. These test cases include problems that cover various portions of the synthetic dataset, entail searches both along roads and though the surface network, and require searches through both high and low impedance areas. Each of the four test cases consist of paths with only one leg. For each case, the population size was 20, and the EA was run for 100 generations.

To address the dynamic aspect for each problem, 80 avoidance nodes are chosen randomly along the least cost path for that problem (shown in Fig. 6). By choosing avoidance points along the shortest path, each additional avoidance point is likely

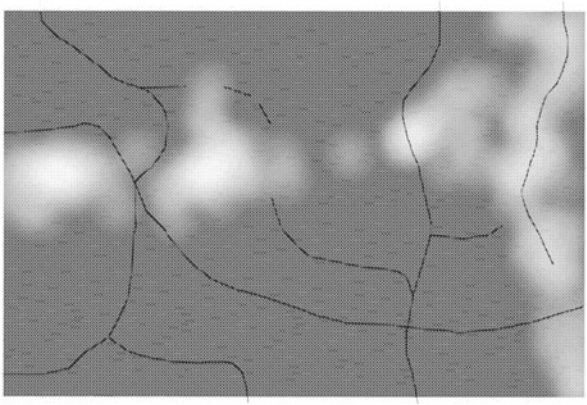
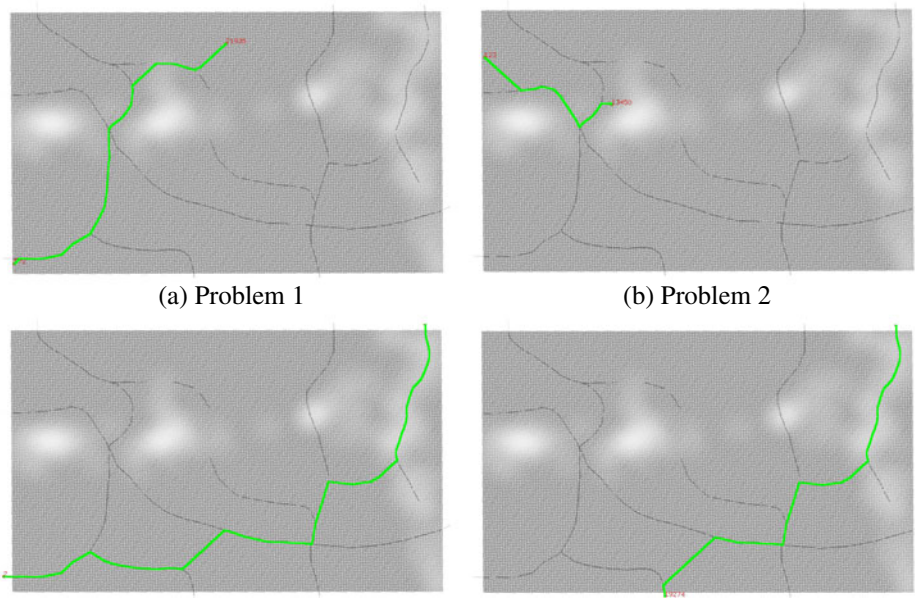


Fig. 5 The synthetic dataset. Note that low impedances are shown in darker tones, high impedances in lighter tones

to force mutation of the path and the avoidance behavior of the algorithm is easily observed. For the first 80 generations an avoidance node is added to the list of nodes to be avoided each generation. In order to see whether the EA is "keeping up" with the rate of change in the environment, it is run an additional 20 generations (from generation 81 to 100) to see if it can improve the quality of the solutions.

For the experimental runs on the synthetic data, we varied value k from 1.00 to 1.08 in steps of 0.02. For each problem and value of k, the EA was run 50 times. This



(a) Problem 1 (b) Problem 2

(a) Problem 3 (d) Problem 4

Fig. 6 The four experiments evaluated on the synthetic dataset

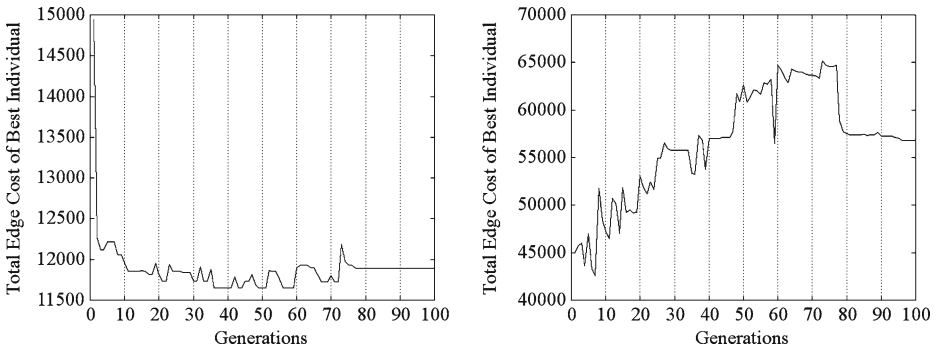
yielded 50 different sets of avoidance nodes for each problem, determined by a random seed (that varied from one to 50). This means that, for a given problem and random seed, the EA was run on a consistent set of avoidance nodes for each level of k.

4.1.2 Evaluation of results

It is useful to examine the dynamics of the EA as the experiment runs through 100 generations. For this we examine graphs of performance versus generation, for $k = 1.02$ and random seed 1 (these graphs are representative of typical behavior). The performance of the algorithm and the dynamic response to the addition of avoidance nodes is similar across Problems 1, 3 and 4. The characteristics of Problem 2 result in a different bias during the search and, as such, we illustrate the dynamics using illustrations of Problems 2 and 3.

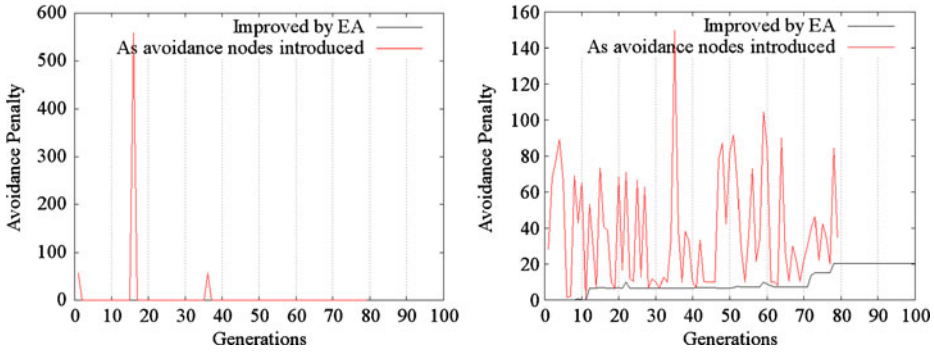
The total path length of the best individual following the EA run for 100 generations illustrates some interesting characteristics of the algorithm (Fig. 7). Note that the path length increases for Problem 3 (in order to avoid the increasing number of avoidance nodes). This is not true for Problem 2 (we will see why this is the case later). Note also that there is little change in performance after 80 generations, when no more avoidance nodes are added. This indicates that the EA is keeping up and does not appear to require more generations to further adapt to the changing landscape.

Figure 8 shows the results for the avoidance penalty. Consider the search dynamics illustrated by the two lines characterizing Problem 3 (Fig. 8b). The red line represents the introduction of new avoidance nodes, resulting in a large increase in avoidance penalty. The black line indicates the performance after the mutation operator has been executed, indicating that mutation does a good job of driving the avoidance penalty back down. To put into context of an applied problem, this means that the addition of a new sentry or sensor disrupts the performance of the agent. However, the agent adapts quickly and learns to avoid the new sensor. The left graph shows a very different scenario. Only rarely do the new avoidance nodes disrupt behavior, and the EA quickly drives the penalty back to zero.



(a) Problem 2 (b) Problem 3

Fig. 7 Total path length following a 100 generation run of the EA, with $k = 1.02$

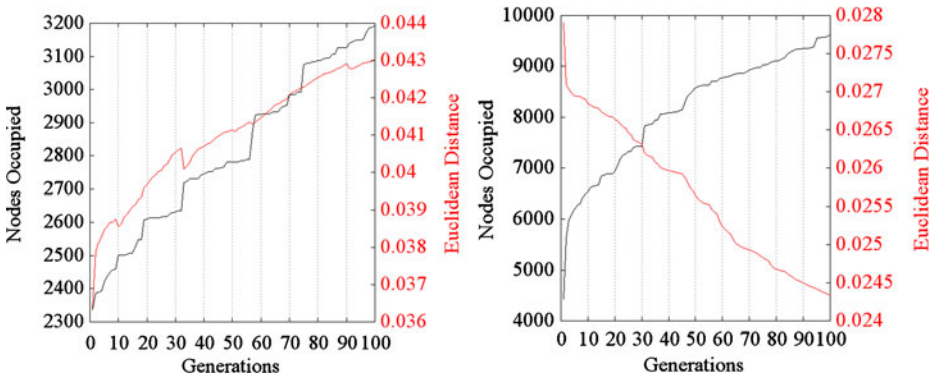


(a) Problem 2 (b) Problem 3

Fig. 8 Avoidance penalty as the EA runs for 100 generations

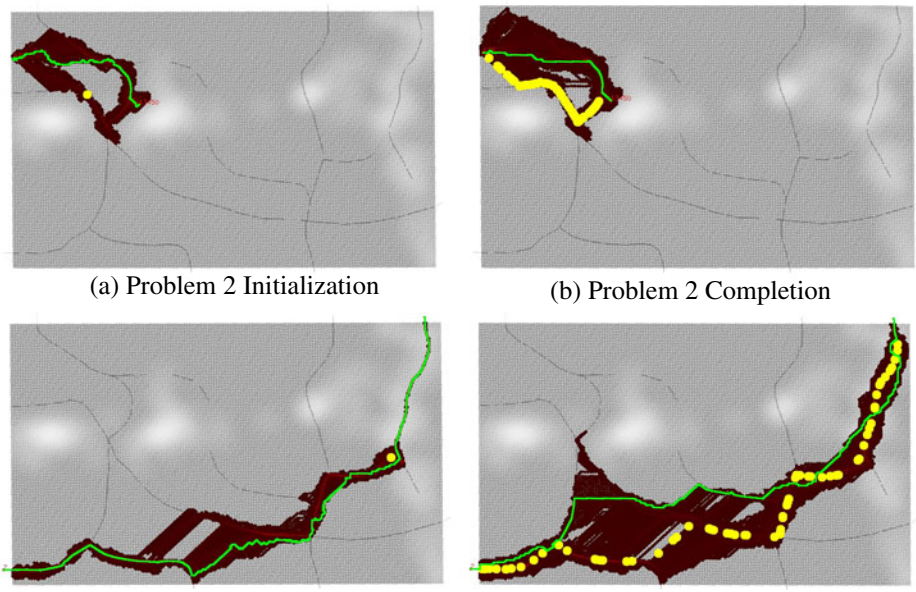
The key to understanding the difference between the two problems is illustrated in Fig. 9. Again, consider the right graph. The black line indicates that the number of nodes examined by the EA increases as the EA runs. The curve is necessarily monotonically non-decreasing—once a node is hit, it can never be removed from the pool of visited nodes. The red line indicates that these nodes hits become more and more uniformly distributed over time, and this is consistent with our expectation. However, Fig. 9a tells a different story. While the number of hit nodes increases, the Euclidean distance from uniformity increases. This indicates that the EA has settled on a subset of nodes and thus potential feasible paths to examine.

This result, though perhaps counterintuitive, can be understood by examining the visual occupancy grid during the search. Figure 10a and b show the occupancy scenarios for Problem 2 at the beginning and end of the search. What has happened is clear. Immediately after the first avoidance node appears (the yellow dot), the EA finds an alternative path that is completely different from the first shortest path. This



(a) Problem 2 (b) Problem 3

Fig. 9 Node count and Euclidean distance as the EA runs for 100 generations



(a) Problem 2 Initialization (b) Problem 2 Completion

(c) Problem 3 Initialization (d) Problem 3 Completion

Fig. 10 Initial and final occupancy grids for Problems 2 and 3. Higher probability is indicated with lighter tones in the occupancy areas

new path almost never intersects the later avoidance nodes (except near the source and destination). The EA focuses attention on the area around the new path and the search emphasizes this path until its fruition.

In contrast, the third problem depicts more typical algorithm behavior. As avoidance nodes are added, the discovered paths go "off road" to avoid them. The search is always drawn back to the main road if possible, because that has least cost. If there are too many avoidance nodes on the least cost path, and a reasonable off road alternative is available, the EA will explore that possibility. The occupancy grid shown in Fig. 10c and d illustrate the range of behavior changes exhibited by the agents, as they attempt to avoid detection.

Experimental runs across multiples of k also prove reliable. Figure 11 show the graphs of the performance metrics for the four problems as k is varied through each experimental run. The previously characterized role of k holds—on average the total path length increases and the number of nodes occupied increases with k. Also, the occupancy grid becomes more uniform. In terms of avoidance, three of the four problems were very easy to solve. However, Problem 4 was more interesting, with the avoidance penalty smoothly decreasing with k.

The occupancy grid offers a useful diagnostic to understand Problem 4. In examining the occupancy grid, it becomes evident that there is a horizontal sequence of avoidance nodes that are difficult for the algorithm to route around (Fig. 12). The horizontal sequence of avoidance nodes (as highlighted) is difficult to avoid as there is no global pressure towards the gap in the center—success is a hit or miss proposition given the nature of the algorithm. However, with a higher k, that

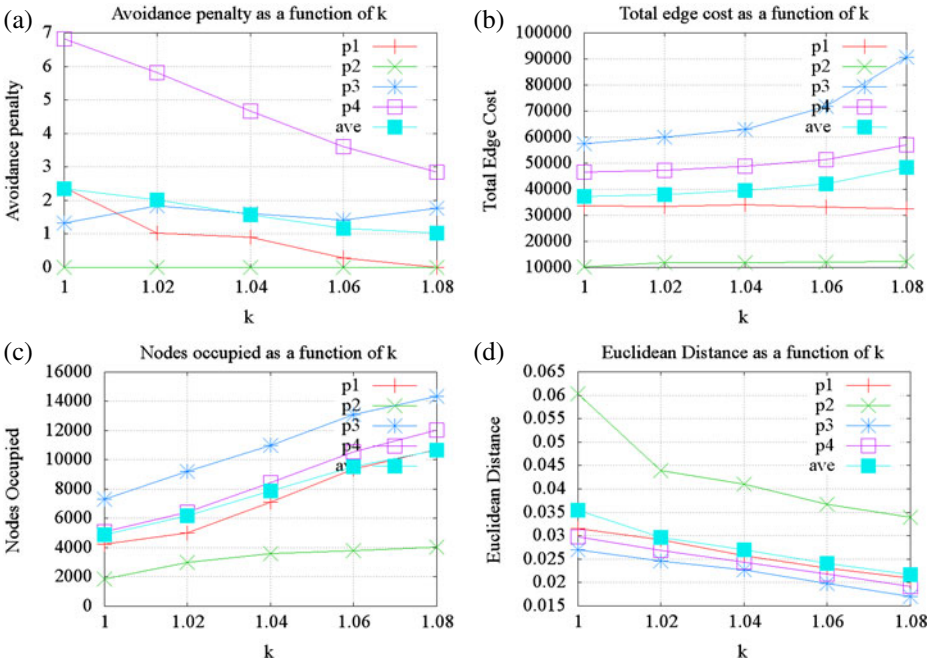


Fig. 11 Performance metrics as k varies

horizontal set can be avoided altogether. Note also that the top vertical set is much easier to avoid with higher k. The number of nodes explored is much less when $k = 1.02$ and the avoidance penalty is 50.12 (Fig. 12a) than when $k = 1.08$ and the avoidance penalty is 3.04 (Fig. 12b).

Overall, the evolutionary algorithm performs as anticipated. Similarly, the response to the avoidance point dynamics is systematic and appropriate. Given the success with the synthetic dataset, we developed a much larger and more varied dataset to simulate "real-world" scenarios.

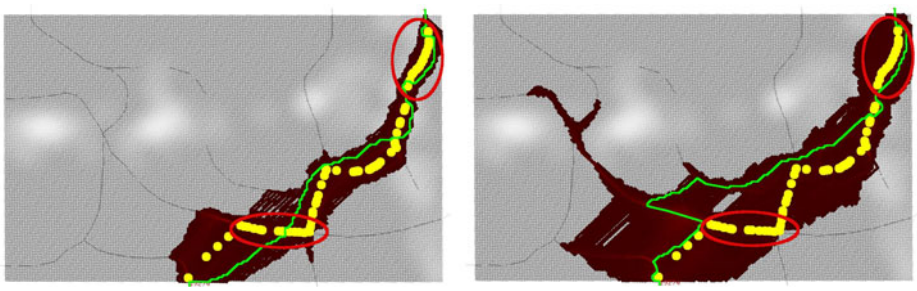


Fig. 12 Search results with parameter relaxation

4.2 The El Paso-Juarez dataset

Once our generalized EA was successfully executed on the synthetic dataset, we constructed a larger, real-world dataset to provide for more realistic test scenarios. Using the readily available Open Street Map open source road data [20], we began by constructing a road network for the cross-border area of El Paso, Texas in the United States and Ciudad Juarez, Chihuahua, Mexico.

The El Paso-Juarez dataset is intended to demonstrate scenarios that might involve surreptitious crossing of the US-Mexico border and, consequently, movement through Mexico and the United States (US) involving some combination of road and overland travel. The conceptual model for this dataset is based on the premise that areas of higher density roads represent greater risk for a traveler attempting to avoid detection. Similarly, off-road travel near such areas represents greater risk than off-road travel in more isolated areas.

To implement the above scenario we computed a line density surface based on the US subset of the road data (Fig. 13). The surface was computed with a 5 km search radius and at 250 m resolution. This search radius served two specific purposes: it extended the overland travel surface associated with US roads into Mexico, and the density estimator results in a surface that naturally decays in value with increasing distance from roads. The former detail allows for scenarios where individuals might travel via road while in Mexico then switch to overland travel as they approach the border, the latter simply serves as a natural proxy for increasing risk near areas with a greater density of roads.

In addition to using the line density surface to generate a surface network to proxy risk of detection, for US roads we sampled the density surface to create a cost multiplier. Whereas the cost associated with road travel in Mexico was 1:1 with road network distance, costs to traverse roads in the US were computed as network distance weighted by the average density of the cells traversed in the line density surface. Also, for high density areas in the US, we eliminated the corresponding edges from the surface network due to the improbability of using overland travel in heavily populated areas.

The Mexico roads, US roads and subset surface network were then integrated via the same procedure used to create the synthetic dataset. The final El Paso-Juarez dataset consists of 187,129 nodes connected by 438,950 edges (see Fig. 14). As with

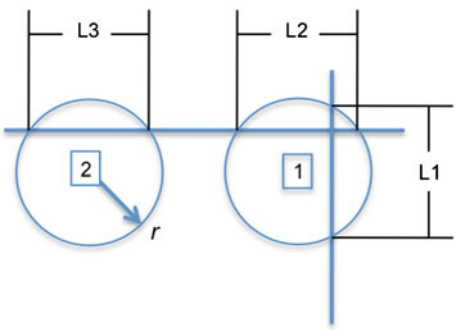


Fig. 13 The approach for calculating line density. For each cell, the total length of the network falling within the search radius of that cell is divided by the area included in the search. For cell 1, line density is computed as $(L1+L2)/\pi r^2$ whereas for cell 2, simply $L3/\pi r^2$

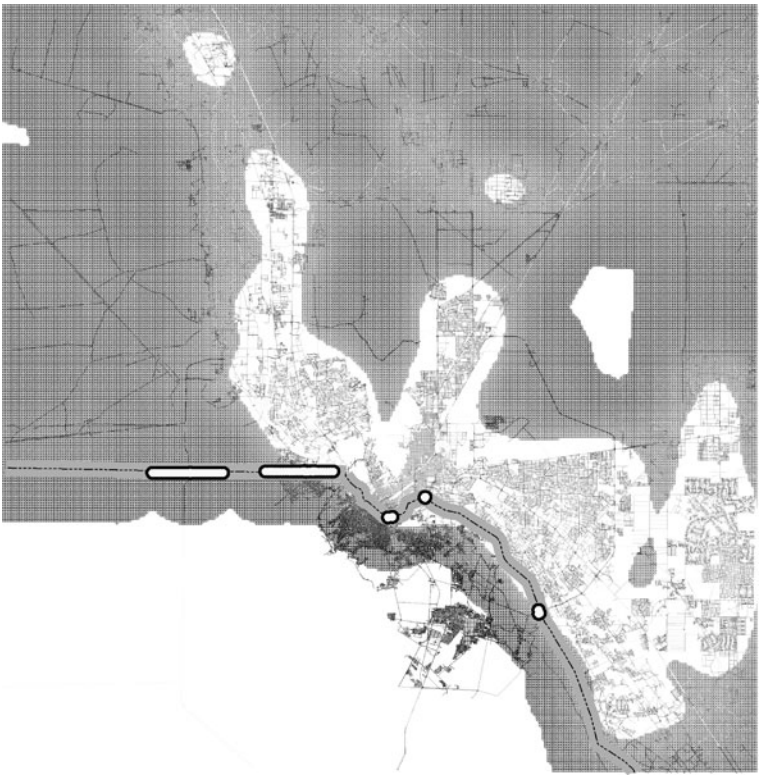


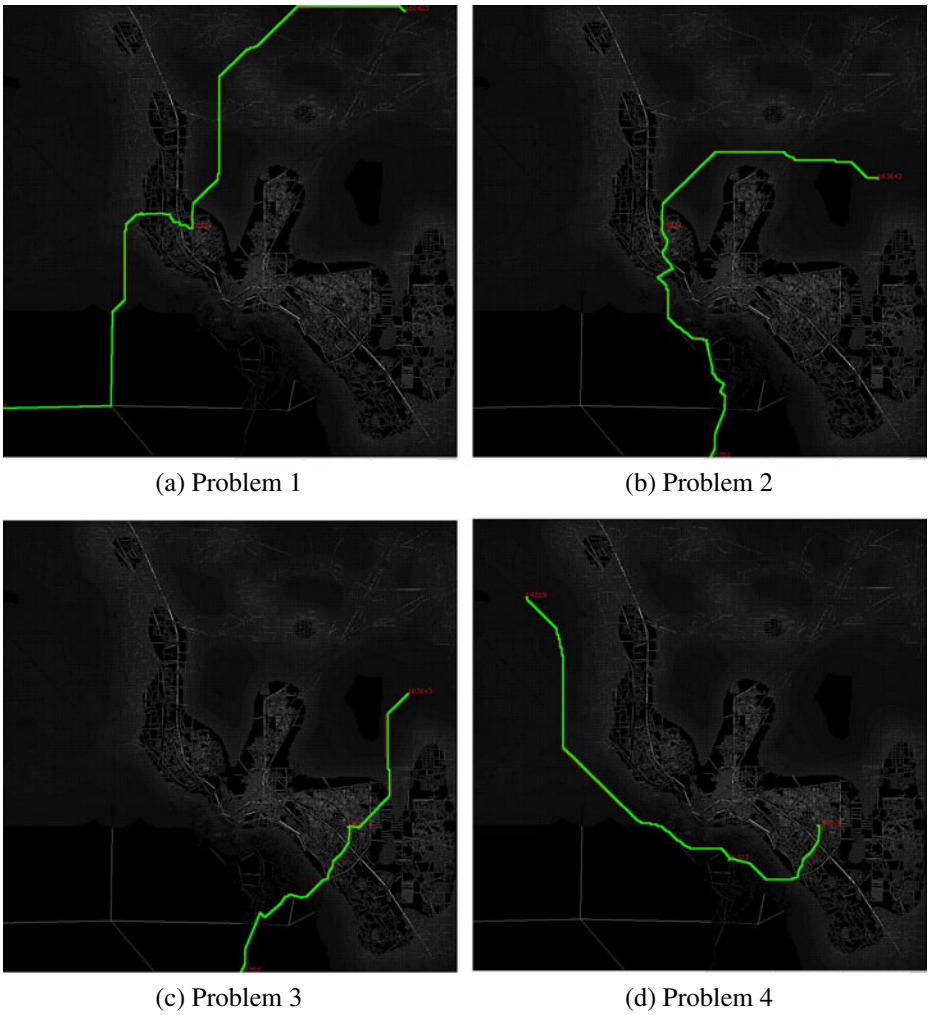
Fig. 14 The El Paso-Juarez dataset. The US-Mexico border is shown as a grey band with a *dot-dash overlay*. Again, note that low costs are shown in *darker tones*, high costs in *lighter tones*. The border control areas associated with 100 avoidance points used in the experimental runs are shown in outline along the border

the synthetic dataset, the final step in processing the integrated dataset required conversion to the adjacency list representation used by the EA.

4.2.1 Experimental runs using the El Paso-Juarez dataset

As with the synthetic dataset, we identified four problems against which to evaluate the developed algorithm (Fig. 15). The complexity was increased to two legs of travel (one intermediary waypoint) for each problem. To address the avoidance aspect of the problem, we identified 100 specific avoidance nodes for the El Paso-Juarez scenarios and introduced a new avoidance node each generation. Again, the EA was run for 100 generations.

The avoidance nodes were selected to cover a large portion of the overland border area west of El Paso and Juarez as well as the more tightly controlled crossings directly between the two cities. For all experiments the set of avoidance nodes is the same, however, the order of insertion changes for each unique random seed. Due to the larger size of these problems, the population size was reduced to two (without noticeable change in performance) in order to increase computational speed.



(a) Problem 1 (b) Problem 2

(c) Problem 3 (d) Problem 4

Fig. 15 Four problems generated to test the EA on the El Paso-Juarez dataset. The *shortest path* individual is shown for context

The four experimental problems were designed to address a variety of possible dynamics in route behavior. The specific problems are hypothetical and designed to simulate potential movement behaviors relative to a series of would-be surreptitious objectives.

– *Problem 1:* In this scenario, the route is initiated in the northeast quadrant of the data, stops in the dense area to the north and west of El Paso, then heads into Mexico and the southwest corner of the study area. This study illustrates the extensive use of overland travel to avoid large traversals in the urban area and to dodge the avoidance points that emerge at the border during the course of successive generations.

- *Problem 2:* This scenario starts in Mexico, stops in the high density area north and west of downtown El Paso, then heads north and east. This scenario might simulate the movement of contraband from Mexico into the United States with an intervening drop point. The dynamics of this scenario illustrate how early intervention along the border crossing influences behavior forward from that point in time. The early intervention disrupts the optimal path and following that disruption, finding an alternative route becomes much more difficult.
- *Problem 3:* The third scenario is similar to Problem 2 in that it simulates the would-be movement of contraband from Mexico into the United States. In this scenario, the origin of the path is in Mexico at the southernmost extent of the study area. Like the previous scenario, this scenario illustrates the pressure associated with moving through the high density downtown area and the consequences of early intervention along the border control points.
- *Problem 4:* The final scenario offers an alternative perspective and simulates someone leaving the United States, acquiring contraband in Mexico and returning to the United States. The scenario begins in downtown El Paso, moves into downtown Juarez, then returns back into the United States and heads toward a less developed area (e.g., a rendezvous point). Again, this simulation illustrates how the establishment of intervention points near the logical border crossing quickly affects potential route behavior.

The above problems include one scenario originating in the US and entering Mexico (Problem 1), two scenarios originating in Mexico (Problems 2 and 3), and one scenario that originates in the US and returns to the US passing through Mexico (Problem 4). The four scenarios illustrate movement through the varied network topologies, multiple encounters with avoidance points and movement through areas where the border crossing is readily constrained versus more open areas.

In order to demonstrate algorithm consistency on the two differently structured datasets, we repeat the sensitivity analysis to variation in parameter k. Consistent with the experimental runs on the synthetic data, we varied value k from 1.00 to 1.08 in steps of 0.02. In order to test higher values of k, we also ran for $k = 1.16$. For each El Paso-Juarez problem scenario and value of k, the EA was run 50 times. In contrast to the synthetic scenarios, the avoidance points for the El Paso-Juarez scenarios are deterministic and occur at the previously mentioned predetermined locations (shown in Fig. 14). However, as mentioned earlier, the order of insertion of the avoidance nodes into the network is random and depends on the random seed.

4.2.2 Evaluation of results

Due to the arguments presented earlier, we are using the EA due to its flexibility to accommodate a problem that would need to be very tightly specified in the context of a variety of alternative SP algorithms. As such, our focus on evaluation is not on run time or big O performance but rather in terms of whether the EA shows relatively consistent behavior and reliability for a variety of data. As such, we repeat the use of metrics associated with total path length, avoidance penalty, node count, and Euclidean distance.

In terms of path length, each of the experimental runs demonstrates a relatively quick increase in path length (see Fig. 16). This is generally consistent with the

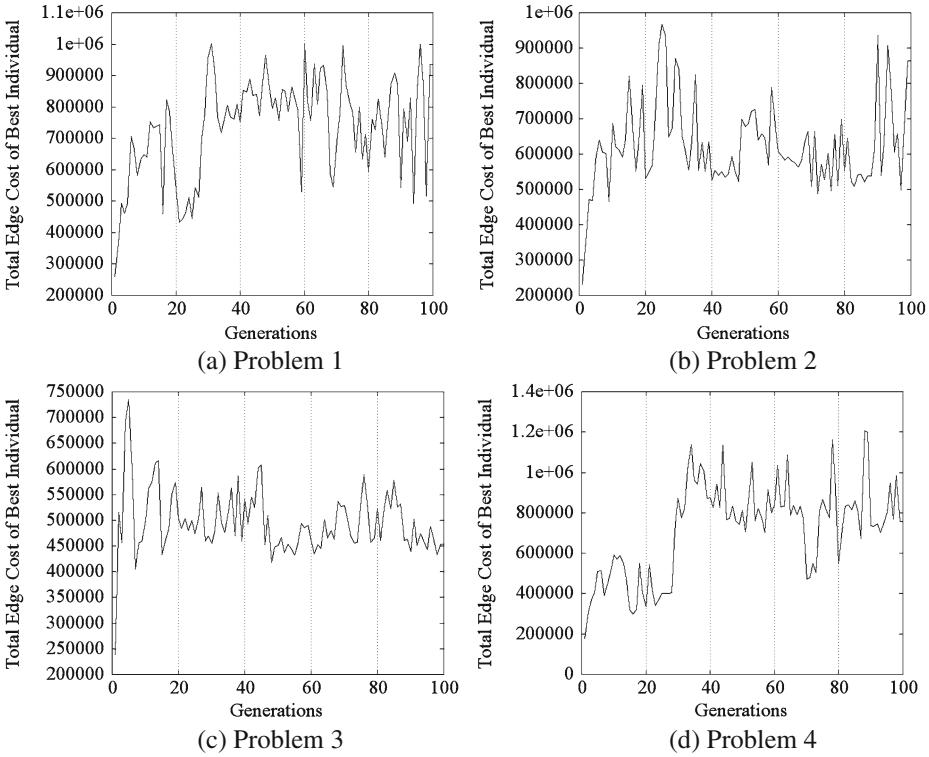


Fig. 16 Total path length as the EA runs for 100 generations

synthetic data as the algorithm quickly departs from the initialized shortest path, due to the introduction of the avoidance nodes. As more avoidance nodes are introduced and the search becomes more difficult, there is a continual tension between avoidance (by increasing path length) and minimizing path length, as "work arounds" are found. Our results with respect to total edge cost, node count and node distribution are consistent with the synthetic data—as k increases the path length and number of nodes hit increases consistently. Also, the node hit distribution becomes more and more uniform (we omit these graphs for the sake of brevity).

Analysis of the trends in avoidance penalty yields additional insight (Fig. 17). Both Problems 3 and 4 have ideal shortest paths that require use of the tightly controlled border area. In that these areas are blocked by avoidance points (the small cluster in the southeast), the EA has a difficult time finding functional solutions via the mutation operator. This is particularly problematic for Problem 3 wherein the border control area is approached from Mexico and the EA is unable to find a solution in 100 generations (see Fig. 17c—the algorithm is unable to reduce the avoidance penalty). In contrast, in a similar scenario but entering Mexico from the US (see Fig. 17d), the greater degrees of freedom (i.e., higher road density and opportunity for overland travel) allow the EA to occasionally find solutions that avoid the tightly controlled border area altogether.

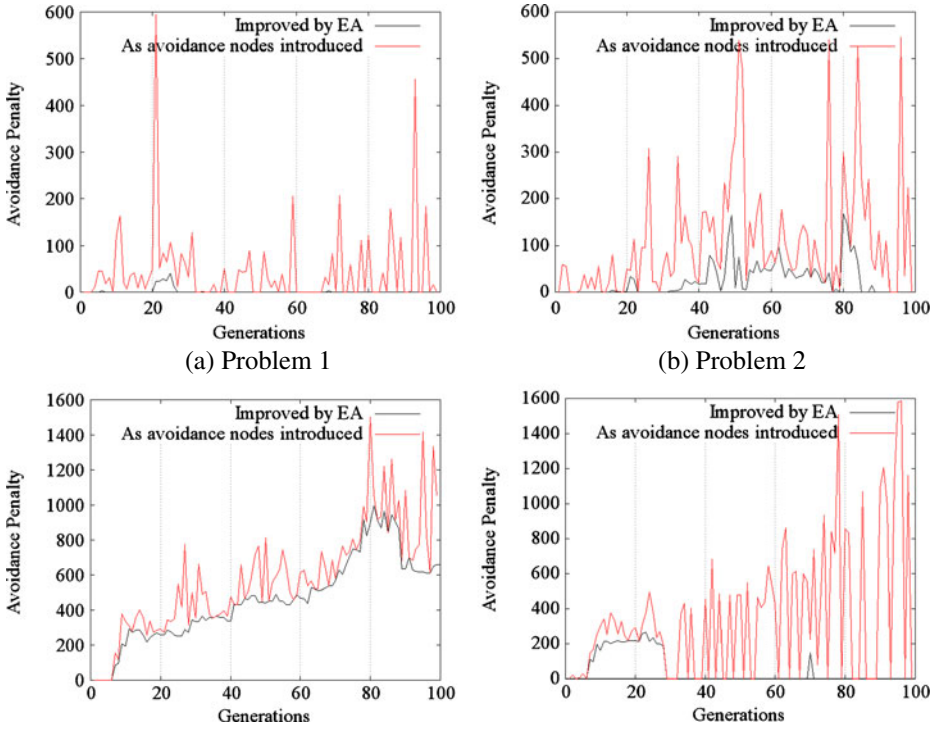
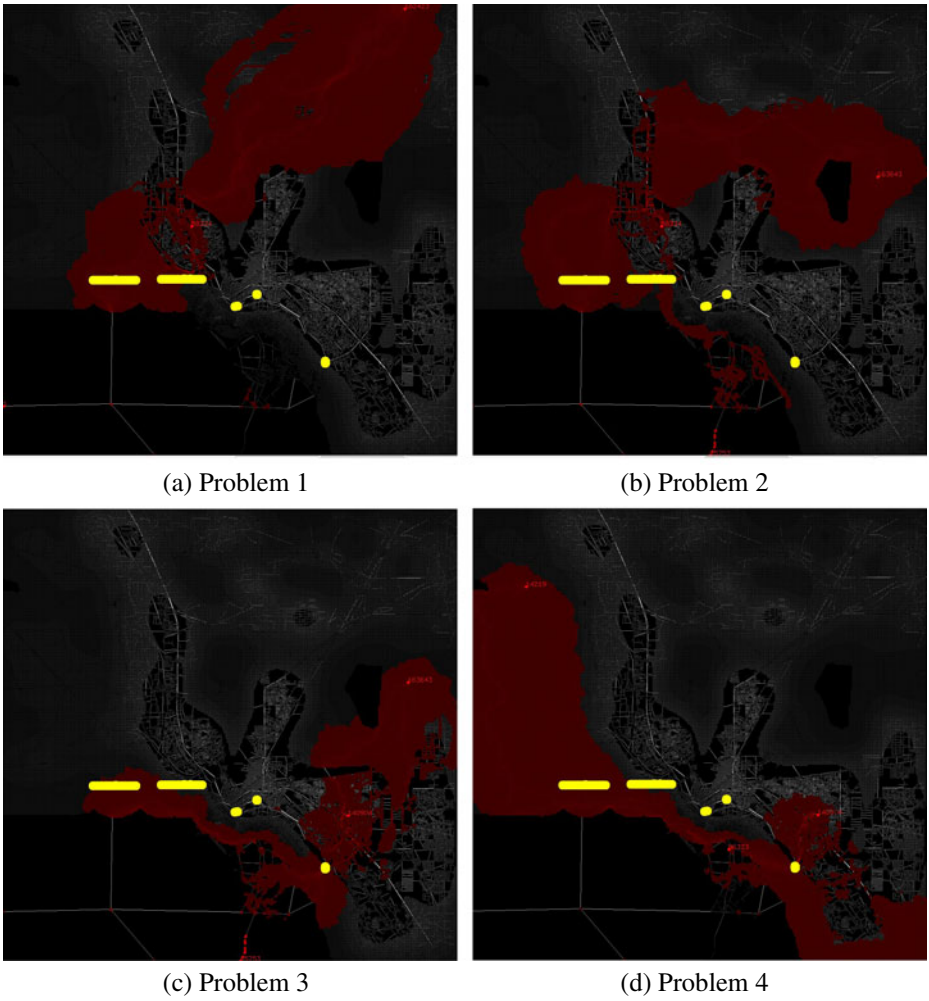


Fig. 17 Avoidance penalty as the EA runs for 100 generations

This behavior can be seen clearly in the final occupancy graphs for the four problems (Fig. 18). For the first two problems, the EA is able to discover the "gap" between the two horizontal avoidance segments. The fourth problem can be solved by taking a very wide detour to the southeast from the US into Mexico. However, this detour is harder to find coming up from Mexico in the third problem, and the path is unable to avoid the southeast cluster of avoidance nodes. In fact, due to the lack of connections between the US and Mexico in this area, what our results indicate is that the southeast cluster of avoidance nodes are extremely effective in terms of inhibiting route options.

This results are emphasized in Fig. 19, which shows the avoidance penalty. The first two problems are quite easy. The third problem is very difficult. No matter what the value of k is, the EA is unable to find a path up from Mexico that totally avoids the southeast avoidance nodes. The fourth problem, in contrast, has intermediary difficulty. But this is not because the avoidance penalty is generally half that of the third problem. It is because the EA can avoid the southeast nodes approximately 50% of the time (hence almost halving the penalty on average). Again, this is due to the greater degrees of freedom that allows the EA to occasionally find the wide detours (as can be seen in Figs. 17d and 18d).



(a) Problem 1 (b) Problem 2

(c) Problem 3 (d) Problem 4

Fig. 18 Final occupancy grids for the four problems

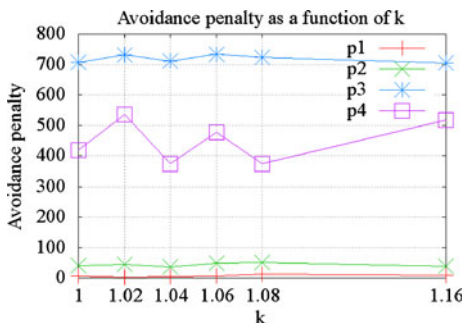


Fig. 19 The avoidance performance metric as k varies

Although k appears not to have an overall dramatic effect on most problems, it does appear to influence the ability of the EA to arrive at a solution for Problem 4. The effect is non-linear—this may be a function of the cost-based as opposed to Euclidean nature of the graph. For this problem the value of k appears to affect the probability that the EA can find the wide detour (from the worst value of 36% for $k = 1.02$ to the best value of 50% for $k = 1.04$). Research is ongoing to understand this dynamic.

5 Conclusion

This paper describes an approach using evolutionary algorithms to search and enumerate geographic information. It details how evolutionary algorithms support the idea of encapsulating "real-world" behaviors into the search process through the use of parameters such as k and the curvature and avoidance penalties associated with the fitness function. Four test scenarios run on each of two datasets demonstrate the manner in which the algorithm uses the input parameters and the fitness function to respond to a dynamic environment. In doing so, the presented research offers several extensions to route finding algorithms associated with more typical single objective shortest path problems.

While the presented scenarios involve the movement of a hypothetical human agent through the environment, the approach is relevant to modeling other phenomena. For example, the approach could easily be adapted to model potential alternative traffic routes given disruption to a critical link in the transportation infrastructure. Similarly, as mentioned in the text, the approach is also ideally suited to model animal movement corridors and the potential impact of anthropogenic features on the animal movement patterns.

Overall, the flexibility of our approach is underpinned by three critical concepts. First, each of the test datasets represents a network of networks. While the test datasets are currently comprised of networks that are geographically embedded, there is no requirement that the networks be strictly spatial (e.g., a social network, a financial network). Second, given that each generation of the evolutionary algorithm encodes potentially feasible paths, we leverage this information rather than discarding it in favor of the "best" potential path. Finally, the approach is fundamentally dynamic; the dynamic response and adaptation of the feasible paths to the new information for each generation offers significant flexibility over more heuristically driven approaches.

In terms of future research, we are currently working in three areas. First, the shift to multiple networks with different cost measures will require a shift to true multi-objective optimization approaches. Second, given the possibility of multiple networks with different cost variables, a more flexible approach to incorporate network proximity in lieu of Euclidian distance for the avoidance function is required. Finally, in order to make the approach relevant to a broader array of problems, we would like to make the parameter set and the fitness function more modular (e.g., the ability to drive the reasoning by flexibly specifying the parameters and fitness function relevant to the particular problem). The flexibility of the evolutionary approach will support these research goals. Overall, however, the demonstrated approach is flexible and extends our thinking about reasoning on complex networks.

References

1. Adriaensen F, Chardon JP, De Blust G, Swinnen E, Villalba S, Gulinck H, Matthysen E (2003) The application of 'least-cost' modelling as a functional landscape model. Landsc Urban Plan 64(4):233–247
2. Beier P, Majka D, Newell S (2009) Uncertainty analysis of least-cost modeling for designing wildlife linkages. Ecol Appl 19(8):2067–2077
3. Bellman R (1958) On a routing problem. Q Appl Math 16:87–90
4. Bennett DA, Tang W (2006) Modelling adaptive, spatially aware, and mobile agents: Elk migration in yellowstone. Int J Geogr Inf Sci 20(9):1039–1066
5. Church RL, Loban SR, Lombard K (1992) An interface for exploring spatial alternatives for a corridor location problem. Comput Geosci 18(8):1095–1105
6. Colizza V, Barrat A, Barthelemy M, Vespignani A (2006) The role of the airline transportation network in the prediction and predictability of global epidemics. Proc Natl Acad Sci 103(7):2015–2020
7. Cova TJ, Johnson JP (2002) Microsimulation of neighborhood evacuations in the urban-wildland interface. Environ Plann A 34(12):2211–2230
8. Cova TJ, Johnson JP (2003) A network flow model for lane-based evacuation routing. Transp Res Part A Policy Pract 37(7):579
9. Pretolani D (2000) A directed hypergraph model for random time dependent shortest paths. Eur J Oper Res 123(2):315–324
10. Davies C, Lingras P (2003) Genetic algorithms for rerouting shortest paths in dynamic and stochastic networks. Eur J Oper Res 144(1):27–38
11. Demetrescu C (2006) DIMACS challenge benchmarks: Colorado, USA. http://www.dis.uniroma1.it/~challenge9/download.shtml
12. Dijkstra EW (1959) A note on two problems in connexion with graphs. Numer Math 1(1):269–271
13. Floyd RW (1962) Algorithm 97: shortest path. Commun ACM 5(6):345
14. Frank WC, Thill J-C, Batta R (2000) Spatial decision support system for hazardous material truck routing. Transp Res Part C Emerg Technol 8(1–6):337–359
15. Gen M, Cheng R, Wang D (1997) Genetic algorithms for solving shortest path problems. In: IEEE international conference on evolutionary computation, pp 401–406
16. Gen M, Lin L (2005) Multi-objective hybrid genetic algorithm for bicriteria network design problem. Complex Int 11:73–83
17. George B, Shekhar S (2008) Time-aggregated graphs for modeling spatio- temporal networks. In: Spaccapietra S, Pan J, Thiran P, Halpin T, Staab S, Svatek V, Shvaiko P, Roddick J (eds) Journal on Data Semantics XI. Lecture notes in computer science, vol 5383. Springer Berlin, Heidelberg, pp 191–212
18. Goodchild M (1977) An evaluation of lattice solutions to the problem of corridor location. Environ Plann A 9(7):727–738
19. Hägerstrand T (1970) What about people in regional science? Pap Reg Sci 24(1):6–21
20. Haklay MM, Weber P (2008) Openstreetmap: user-generated street maps. IEEE Pervasive Computing 7:12–18
21. Hargrove WW, Hoffman FM, Efroymson RA (2005) A practical map-analysis tool for detecting potential dispersal corridors. Landsc Ecol 20(4):361–373
22. Hart P, Nilsson N, Raphael B (1968) A formal basis for the heuristic determination of minimum cost paths. IEEE Trans Syst Sci Cybern 4(2):100–107
23. Huang B, Cheu RL, Liew YS (2004) GIS and genetic algorithms for HAZMAT route planning with security considerations. Int J Geogr Inf Sci 18(8):769–787
24. Lin S, Kernighan B (1973) An effective heuristic algorithm for the traveling-salesman problem. Oper Res 21(2):498–516
25. Loui RP (1983) Optimal paths in graphs with stochastic or multidimensional weights. Commun ACM 26:670–676
26. Lup L, Srinivasan D (2007) A hybrid evolutionary algorithm for dynamic route planning. In: IEEE Congress on Evolutionary Computation, 2007. CEC 2007, pp 4743–4749
27. McDonald DB (2007) Predicting fate from early connectivity in a social network. Proc Natl Acad Sci 104(26):10910–10914
28. Miller HJ, Wu Y-H (2000) Gis software for measuring space-time accessibility in transportation planning and analysis. Geoinformatica 4(2):141–159

29. Mooney P, Winstanley A (2006) An evolutionary algorithm for multicriteria path optimization problems. Int J Geogr Inf Sci 20(4):401–423
30. Okabe A, Okunuki K (2001) A computational method for estimating the demand of retail stores on a street network and its implementation in GIS. Trans GIS 5(3):209
31. Orda A, Rom R (1990) Shortest-path and minimum-delay algorithms in networks with time-dependent edge-length. J ACM 37:607–625
32. Osborn FV, Parker GE (2003) Linking two elephant refuges with a corridor in the communal lands of Zimbabwe. Afr J Ecol 41(1):68–74
33. O'Sullivan D, Morrison A, Shearer J (2000) Using desktop GIS for the investigation of accessibility by public transport: an isochrone approach. Int J Geogr Inf Sci 14(1):85–104
34. Pfoser D, Jensen C (2003) Indexing of network constrained moving objects. In: Proceedings of the 11th ACM international symposium on advances in geographic information systems. ACM, p 32
35. Prager SD, Spears WM (2009) A hybrid evolutionary-graph approach for finding functional network paths. In: GIS '09: Proceedings of the 17th ACM SIGSPATIAL international conference on advances in geographic information systems. ACM, New York, pp 306–315
36. Shaw L, Spears WM, Billings L, Maxim P (2010) Effective vaccination policies. Inf Sci 180(19):3728–3744
37. Spears WM (2000) Evolutionary algorithms: the role of mutation and recombination. Natural computing series. Springer
38. Trajcevski G, Wolfson O, Hinrichs K, Chamberlain S (2004) Managing uncertainty in moving objects databases. ACM Trans Database Syst 29(3):463–507
39. Xie Z, Yan J (2008) Kernel density estimation of traffic accidents in a network space. Comput Environ Urban Syst 32(5):396–406
40. Yamada I, Thill J (2004) Comparison of planar and network K-functions in traffic accident analysis. J Transp Geogr 12(2):149–158
41. Yuan M, Hornsby K (eds) (2007) Computation and visualization for understanding dynamics in geographic domains: a research agenda. CRC
42. Zhan F, Noon C (1998) Shortest path algorithms: an evaluation using real road networks. Transp Sci 32(1):65–73

William M. Spears received a Ph.D. in Computer Science from George Mason University in 1998. He has an international reputation for his expertise in evolutionary computing and has a published book on the topic. He has co-edited books on swarm robotics and evolutionary computation. His current research includes distributed robotics, the epidemiology of virus spread, evolutionary algorithms, particle swarm optimization, complex adaptive systems, and learning and adaptation. He invented the field of physicomimetics and was co-founder of the University of Wyoming Distributed Robotics Laboratory. He has approximately 80 publications and is the CEO of Swarmotics, LLC.

Steven D. Prager (PhD, Simon Fraser University) conducts research in the areas of spatial and spatiotemporal uncertainty, complex networks and geographic semantics, and teaches geographic information science at the University of Wyoming. Dr. Prager's publications and research interests include research on the use of ontology for minimizing spatiotemporal uncertainty, dynamics of geographically embedded complex networks, semantically driven Bayesian approaches to reconciling positional error, and understanding human-environment interactions as complex adaptive systems.