Generating Approximate Region Boundaries from Heterogeneous Spatial Information: an Evolutionary Approach

Steven Schockaert[a], Philip D. Smart[b], Florian A. Twaroch[b]

[a]*Department of Applied Mathematics and Computer Science, Ghent University, Krijgslaan 281 - S9, 9000 Gent, Belgium*
[b]*School of Computer Science, Cardiff University, 5 The Parade, Roath, Cardiff, UK*

Abstract

Spatial information takes different forms in different applications, ranging from accurate coordinates in geographic information systems to the qualitative abstractions that are used in artificial intelligence and spatial cognition. As a result, existing spatial information processing techniques tend to be tailored towards one type of spatial information, and cannot readily be extended to cope with the heterogeneity of spatial information that often arises in practice. In applications such as geographic information retrieval, on the other hand, approximate boundaries of spatial regions need to be constructed, using whatever spatial information that can be obtained. Motivated by this observation, we propose a novel methodology for generating spatial scenarios that are compatible with available knowledge. By suitably discretizing space, this task is translated to a combinatorial optimization problem, which is solved using a hybridization of two well-known meta-heuristics: genetic algorithms and ant colony optimization. What results is a flexible method that can cope with both quantitative and qualitative information, and can easily be adapted to the specific needs of specific applications. Experiments with geographic data demonstrate the potential of the approach.

Key words: Spatial Reasoning, Genetic Algorithms, Ant Colony Optimization, Geographic Information Retrieval

1. Introduction

To be effective, applications dealing with spatial information need to abstract away from unnecessary subtleties, focusing exclusively on those aspects of physical space that are relevant in the given context. In domains such as

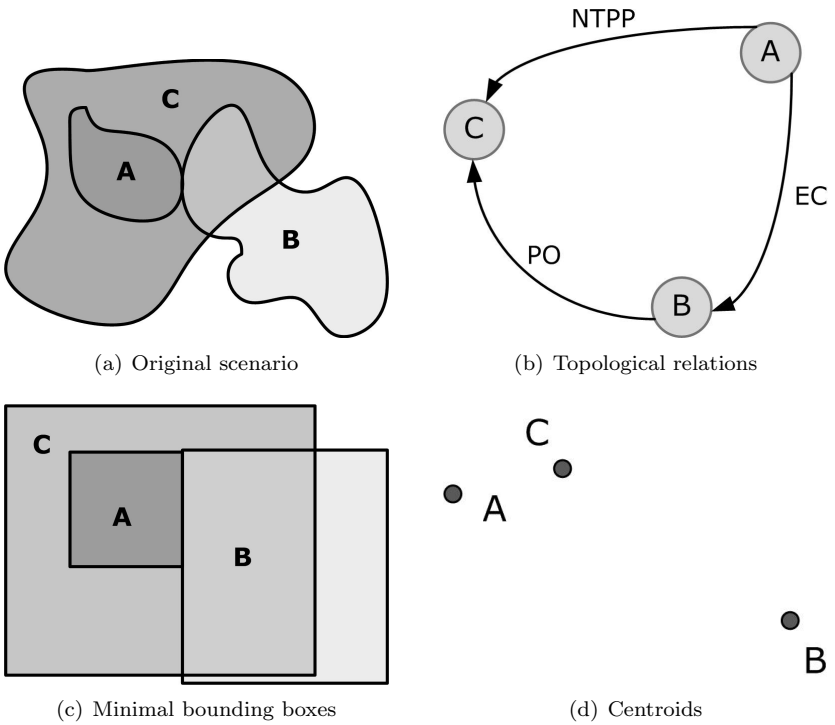
Email addresses: Steven.Schockaert@UGent.be (Steven Schockaert), p.smart@cs.cardiff.ac.uk (Philip D. Smart), f.a.twaroch@cs.cardiff.ac.uk (Florian A. Twaroch)

artificial intelligence (AI) and spatial cognition, for instance, there is a large interest in qualitative approaches to describing space [7, 19, 21, 30, 56, 59]. Spatial configurations are then typically reduced to a number of qualitative relations, of a given type, which are known to hold between a given set of regions (e.g. region A is known to be contained in region B, or is known to be adjacent with B). As an example, Figure 1(a) depicts a spatial configuration involving three regions A, B and C. One of the typical AI approaches is to model only topological relations that hold between these regions, because only topological information is relevant, or more likely, because only topological information is available. As shown in Figure 1(b), these topological relations can be summarized in a graph, where nodes correspond to regions, and edges to topological relations. Topological relations are usually modeled in either the 9–intersection model [15] or in the conceptually similar Region Connection Calculus (RCC; [56]). In Figure 1(b) and throughout this paper, the notations of the latter framework are used. Their intuitive meaning is illustrated in Figure 2. Other types of qualitative spatial relations that play an important role in AI include direction [19], distance [7], and size [23] relations. A graph of topological relations can be seen as an instance of a constraint satisfaction problem (CSP), where the number of constraints is finite, but the domains of the variables, which could be instantiated by arbitrary polygons[1], are infinite. Typical reasoning tasks of interest consist of checking whether a certain set of topological relations is consistent (or satisfiable), i.e. if there can actually exist polygons such that the given topological relations are all satisfied [59, 58, 14].

On the other hand, in the context of geographic information systems (GIS) or computer vision, quantitative information tends to play a more central role. In the case of GIS, for instance, the focus tends to be more on efficiently answering queries against large spatial databases than on reasoning about incomplete descriptions. Another important issue in GIS systems is the integration of spatial information of different resolutions [13]. To speed up computation (or because rich boundaries are not available), the exact boundaries of regions are often reduced to simpler representations. For example, a typical GIS approach is to simplify all boundaries to minimal bounding boxes (MBBs). In this case, depicted in Figure 1(c), every region is represented by the smallest rectangle encompassing the region, whose sides are parallel to the axes. Note that special care needs to be taken when modeling spatial relations between regions by comparing their MBBs [12]. Another popular alternative is to model each region as a single point, called a centroid (Figure 1(d)).

In all these application domains, different assumptions are made on what spatial information is available (qualitative/quantitative, accurate/approximate, certain/probabilistic, crisp/fuzzy, etc.), and on what the behavior of a spatial information processing module should be (proving the consistency of spatial constraints, efficiently addressing spatial queries, generating approximate spa-

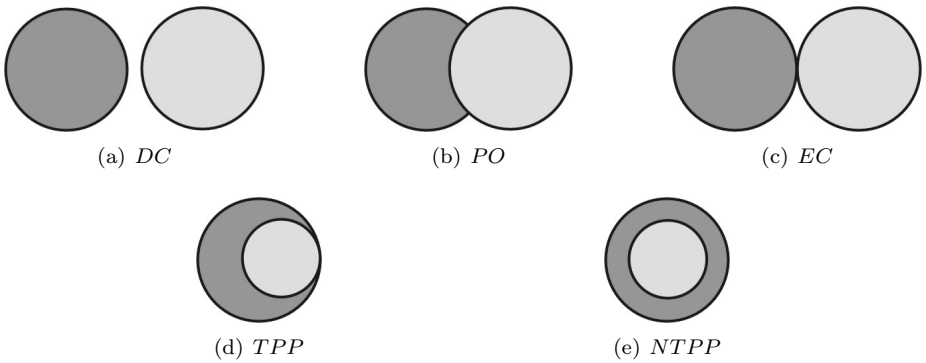
[1]In practice, the set of allowed instantiations is often even larger, allowing for instance arbitrary regular closed (or regular open) subsets in certain topological spaces.



(a) Original scenario (b) Topological relations

(c) Minimal bounding boxes (d) Centroids

Figure 1: A spatial scenario can be abstracted in many different ways.



(a) DC (b) PO (c) EC

(d) TPP (e) $NTPP$

Figure 2: Topological relations from the Region Connection Calculus: disconnected (DC), partially overlapping (PO), externally connected (EC), tangential proper part (TPP), and non–tangential proper part (NTPP)

tial models from incomplete or uncertain data, etc.). As a result, existing techniques for manipulating spatial information tend to deal exclusively with one type of information, e.g. only topological relations, only minimal bounding boxes, only points (centroids), etc. Consistency checking procedures for topological relations, for instance, cannot cope with quantitative spatial information. Nonetheless, given only the minimal bounding boxes of regions A, B and C from Figure 1(c), it may be of interest to certain applications to know what topological relations can hold between A, B and C; e.g. is it still possible that $TPP(A, C)$ and $DC(B, C)$ both hold? Moreover, consistency checking is often not sufficient, as we may be interested in obtaining visualizable spatial configurations (e.g. by constructing actual polygons). Similarly, in the GIS field, a number of techniques have been studied to generate polygons from incomplete quantitative information (e.g. sets of points that are known to lie in certain regions), but these techniques cannot deal with additional qualitative information (e.g. topological relations that are known to hold), or additional quantitative information of a different type (e.g. bounding boxes that are available for some regions).

In general, two fundamentally different approaches may be considered in the face of heterogeneous spatial information. First, techniques may be developed that derive useful conclusions by combining pieces of information from different data sets, each of which may have different characteristics. Despite that completeness of the resulting algorithms is difficult to guarantee, in many application scenarios this approach delivers satisfactory results, and may moreover be implemented efficiently, allowing on-line querying of available data. This solution has been proven particularly useful in classical GIS settings, where integrating information from different sources easily leads to problems, e.g. due to the use of different resolutions [3], and where care should be taken when quantitative descriptions are incomplete [2]. The second approach consists of constructing quantitative representations at a fixed resolution that are as compatible as possible with all available information. This is the methodology that we follow in this paper, its advantage being that once the quantitative representations have been obtained, spatial data may be queried as if complete, consistent, quantitative information were available, making it possible, for instance, to visualize information that was initially partially qualitative or otherwise incomplete. On the other hand, obtaining suitable quantitative representations may be time-consuming, requiring this step to be performed off-line. More fundamentally, the resulting quantitative representations are only approximations, without any guarantees on accuracy. Since these drawbacks are effectively problematic for many standard GIS scenarios, this second method has only received very limited attention thus far. Nonetheless, it is the second method which seems most useful for tasks such as geographic information retrieval (GIR; [32]). Indeed, by their very nature, information retrieval systems operate using a best effort principle: while search engines provide no guarantees that the returned search results are relevant, they attempt to find the most relevant ones, given available information. In the same way, in geographical information retrieval, we may attempt to find boundaries for a certain geographic region which are as mean-

ingful as possible, allowing us to find meaningful results even when available geographic information is scarce.

The main goal of GIR systems is to spatially enable search engines, allowing them to understand the meaning of geographic search terms and constraints. We may be interested, for instance, in finding information about hotels *in London's Bloomsbury neighborhood*, in blog posts about the US elections by people *from Western Europe*, or in pictures of the *Hong Kong* skyline. As the total number of queries involving place names is substantial — a survey of the 2004 Excite search engine query logs revealed that 15% of all queries contained place names [63] — proper treatment of geographic information in search engines is paramount. One problem with traditional, text–based search engines is that place names tend to be highly ambiguous. For example, dozens of places in the US are called Springfield, hence a mechanism is needed to discover if an occurrence of Springfield in a document corresponds to the same place as an occurrence of Springfield in a query [49]. A second problem is related to the variety of ways places can typically be referred to. For example, a text–based search engine cannot straightforwardly deduce that a document about Oxford Street might be relevant to a user interested in shopping in London. Despite that there is a growing interest in GIR, and in spatially aware technologies in general (e.g. Google Earth [2] or Microsoft Maps Live [3]), most existing work focuses on certain subproblems looking at one type of spatial information only. In particular, due to the lack of better tools, existing approaches to GIR treat available qualitative information more or less independently from available quantitative information, although certain hybrid representational frameworks are beginning to emerge [71, 76, 27]. Using the technique introduced below, available spatial information can be more tightly integrated, paving the way for better performing GIR systems.

The aim of this paper is to introduce a sufficiently general technique for generating appropriate polygons from heterogeneous spatial information, often involving a mixture of qualitative and quantitative information. This problem can be cast into a combinatorial search problem by treating the available spatial information as constraints on allowed instantiations (polygons) of variables (regions), and by appropriately discretizing the real plane. Specifically, we propose an approximate algorithm for spatial information processing, based on a combination of two types of evolutionary algorithms: genetic algorithms (GA; [29]) and ant colony optimization (ACO; [10]). The genetic algorithm is used to find an optimal ordering by which variables (i.e. regions) and spatial relations should be processed. In each generation, a spatial scenario is generated for each chromosome, whose quality is evaluated in terms of the number of spatial constraints that it satisfies. This evaluation is not only used to define the fitness of the chromosomes, it is also used by an ACO algorithm to favor, in future generations, spatial configurations which are similar to high–quality scenarios

[2] `http://earth.google.com`
[3] `http://maps.live.com`

that were found earlier.

The structure of this paper is as follows. In the next section, we review existing work on spatial information processing in more detail. Next, in Section 3, we show how the problem of manipulating heterogeneous spatial information can be translated to a combinatorial search problem, and we introduce a naive baseline algorithm. Subsequently, Section 4 discusses how this baseline algorithm can be improved by using a genetic algorithm. In Section 5, we present a further improvement based on ant colony optimization. An experimental validation of our techniques on real–world geographic data is presented in Section 6. Finally, Section 7 presents our conclusions.

2. Related Work

2.1. Spatial Reasoning

There has been considerable work on reasoning about topological information encoded in RCC–8, an important subfragment of the RCC consisting of eight base relations: the five relations from Figure 2, the equality relation EQ, and the inverses $NTPP^{-1}$ and TPP^{-1} of $NTPP$ and TPP. Indefinite topological information can be encoded in RCC–8 by enumerating which of the eight base relations may hold between a given pair of regions. Conceptually, spatial relations in RCC–8 are therefore represented as subsets of $\{DC, PO, EC, TPP, NTPP, EQ, TPP^{-1}, NTPP^{-1}\}$. Most reasoning tasks of interest in RCC–8 are NP–complete [59]. In consequence, and inspired by results about temporal reasoning in the interval algebra [1], many research efforts have been devoted to finding tractable subfragments of RCC–8, i.e. subsets of the 2^8 relations that can be expressed in RCC–8 for which reasoning is tractable [59, 26, 45, 57]. In [45], it was shown that reasoning in RCC–8 is tractable, provided only base relations are used (i.e. no indefinite information). Of special interest are subsets of RCC–8 relations that are maximally tractable, i.e. such that every proper superset of RCC–8 relations would result in NP–completeness. A first maximal tractable subfragment, containing 146 relations, was identified in [59]. Two additional maximal tractable subfragments were identified in [57], containing 158 and 160 relations. In [57] it was moreover shown that these three subfragments are the only maximal tractable subfragments of RCC–8 that contain all eight base relations. In all three subfragments of RCC–8, satisfiability can be checked using an $O(n^3)$ path–consistency algorithm, similar to Allen's algorithm for temporal reasoning.

Usually, a knowledge base of RCC–8 relations is called satisfiable (or consistent) if it can be realized in some topological space, i.e., if all variables can be interpreted by regions in some topological space such that all imposed relations hold [59]. In practice, however, it might be interesting to know whether a set of RCC–8 formulas can be realized by (regular closed) subsets of, for example, $\mathbb{R}^2$ or $\mathbb{Z}^2$ and, if so, which additional constraints on these subsets might be imposed (e.g., convexity, internal connectedness, etc.). In [58], it was shown that any satisfiable set of RCC–8 formulas can be realized in $\mathbb{R}^n$ for every n in $\mathbb{N} \setminus \{0\}$.

A similar realization algorithm, which runs in cubic time, was also presented in [38]. These results imply that satisfiability and realizability in $\mathbb{R}^n$ are essentially equivalent. For $n \geq 3$, this result also holds when regions are constrained to be internally connected, and even if they are constrained to be polytopes. Unfortunately, for $n = 2$ and $n = 1$ this result does not hold in general. This implies that some satisfiable sets of RCC–8 formulas cannot be realized as polygons. This latter problem — checking the realizability of RCC–8 formulas as polygons — can be reduced to the problem of recognizing a special class of graphs called string graphs [26, 35], a hard problem which until recently was not even known to be decidable. In [64], it was shown that this problem is NP–complete.

Other types of qualitative spatial relations have been considered in the literature as well, most notably cardinal directions [51, 25, 70]. Of special interest are techniques for combining different types of qualitative relations, such as the integration of topology and direction [68, 39], or topology and size [23]. However, different types of spatial relations interact in very subtle ways, which makes the search for complete reasoning algorithms highly non–trivial. Therefore, it seems not feasible to extend this approach to the kind of expressivity we require of spatial information in the context of, among others, GIR. This paper presents an alternative methodology for dealing with expressive spatial information, which, from a reasoning perspective, achieves a substantially larger degree of expressivity by sacrificing completeness.

2.2. Approximating Region Boundaries

For many of the geographic regions people refer to, either accurate boundaries do not exist (e.g. vague regions such as London's West End), or they are not available in the given context (e.g. because the licensing costs are too high). On the other hand, there is a large interest in geo-data acquisition, i.e. the discrete pointwise observation of geographical phenomena [20]. For instance, based on user surveys [43], or web mining techniques [65, 34], we may have a set of points at our disposal which we know are located in a given region. Accordingly, a number of techniques have been developed in the GIS community to construct approximate region boundaries from a given set of points. For instance, Galton and Duckham [22] review hull algorithms based on triangulation and a variation of the well known gift-wrapping method called the swing-hammer algorithm. Lam [36] proposes a non–exhaustive method to model regions based on disks. Both of these methods work with varying sizes of point sets, but the uncertainty of the region to be modeled is not considered. Kernel density estimation (KDE) methods, on the other hand, do explicitly take this uncertainty into account. They have been used in several areas of geographic information analysis such as crime and traffic investigations [5], but also for the automated definition of city centres [75] as well as geographic information retrieval [54]. The main principle behind KDE is based on determining a weighted average of data points within a moving window centred on a grid of points. In this way, KDE turns a vector into a field representation. The actual result depends on the choice of an appropriate kernel function and a bandwidth parameter, although experimental studies have found that the choice of the kernel function is less important

than the choice of the bandwidth parameter [48]. A common automated way of setting the bandwidth, based on statistical features of the point set, is the rule of thumb [69]. Different KDE representations can be combined using an indexed overlay, which allows to join different variables when modelling a region or phenomenon. Thurstain-Goodwin and Unwin [75] modelled city centres by overlaying kernel density representations of indices for property, economy, diversity and visitor attractions of a town. In order to derive an actual polygon for a region from the three-dimensional kernel density surface, an appropriate threshold needs to be chosen, which can be interpreted as a level of confidence. The resulting polygon is a contour of the kernel density surface.

The use of KDE has been controversial, its main criticism being the rather arbitrary choice of parameters [62] as well as the fact that it is only suitable for large point sets, which are typically only available for popular regions. On the other hand, given a sufficiently large point set, kernel density models are quite robust against outliers. Adaptive kernel parameters such as suggested by Brunsdon [6] or Sain [62] can consider the variation of point density in a spatial point pattern. Using a Gaussian kernel they provide a means to analyse and represent a region based on statistical grounds. As we will show in Section 6 below, in the presence of additional qualitative information, using KDEs as heuristic information in ACO algorithms can alleviate both the problem of KDEs with small data sets and the problem of selecting appropriate thresholds.

2.3. Geographic Information Retrieval

Geographic Information Retrieval (GIR) facilities the retrieval of geographic information from large document collections such as the web. For each document in the collection, a spatial footprint is generated by geo-parsing the document and extracting the geographical feature objects. These spatial footprints are then combined with a classical index of the document keywords, yielding a searchable structure to perform geospatial queries. Geospatial queries involve place names or geographic concepts along with a spatial preposition and possibly a locational component [33, 41]. Less formally, a geospatial query is a "query about the spatial relations of entities geometrically defined and located in space" [16]. To determine the relevance of a document, its spatial footprint is then compared with the locations and place names from the query. Alternatively, spatial relationships can be determined directly from qualitative relations that are stored explicitly in the document index.

Even if they do not fully support the aforementioned structured geospatial queries, web search engines such as Google are progressively being enriched with spatial semantics to facilitate more sophisticated retrieval techniques [32], including query term expansion (e.g. adding nearby places to increase recall), resolving of place name coreferences (e.g. "Paris" and "the French capital") and spatially-related place lookup (e.g. "medieval towns in Southern France"). To enable such spatial awareness, structured information from official sources is mostly used (gazetteers, ontologies). Often however, an extra effort is made to automatically expand this structured information. For instance, qualitative

spatial information about places can be extracted from natural language locational phrases on general web pages, using shallow or deep linguistic processing [55, 53, 67], or from semi-structured web resources such as Wikipedia[4] (see [55] for a comparison of different Wikipedia approaches). Quantitative information, such as point sets to generate approximate region boundaries, can be extracted by looking at the context in which addresses occur, or by relying on web 2.0 initiatives, such as the Open Street Maps Project [5], Geonames [6], Wikimapia [7], or social web sites in general. Note that the quantitative information that can be found on such free resources is often restricted to simple point based geometric data, leading to parsimonious (frugal) spatial models [31]. As an alternative, or as an addition, accurate (paid for) spatial footprints of geographical features can be obtained from official sources, such as the Ordnance Survey (UK's national mapping agency), who provide administrative subdivisions as a multilevel hierarchy of highly detailed polygons[8]. However such information is licensed and is not usually applied for large-scale web GIR systems. Importantly, all quantitative geographic resources are inherently incomplete, as free resources rely on user input, and official sources are typically restricted to administrative regions. Hence, there is a clear need to augment incomplete quantitative information with incomplete qualitative information, mined from textual resources, to obtain a richer geographic model, and thus to enhance geographic information retrieval tasks.

2.4. Evolutionary Spatial Information Processing

A number of metaheuristics have already been applied in the context of spatial information processing. In [44], for instance, simulated annealing is used to reason about possibly conflicting topological relations. The task described consists of choosing an RCC–8 base relation for each pair of regions under consideration which together form a consistent topological description. The choices should moreover lead to a topological description that is as close as possible to an initial, inconsistent set of topological relations.

Genetic algorithms have been used to solve facility layout problems [28, 40]. These problems resemble quadratic assignment problems and are thus more closely related to classical combinatorial optimization. Nonetheless, the algorithm from [28] has some similarities with ours. There as well, the main purpose of the genetic algorithm is to obtain an ordering in which variables should be processed; to find the actual regions an algorithm is used that starts from one block and maintains an increasing set of contiguous blocks. In [60], a genetic algorithm is used to efficiently solve spatial queries against a large database of polygons (e.g., "find a hospital in an urban area adjacent to a park

[4] http://www.wikipedia.org
[5] www.openstreetmap.org
[6] www.geonames.org
[7] www.wikimapia.org
[8] Of note, even after using state of the art digitization methods, some errors still exist e.g. slivers, overshoots etc.

and a highway"). A hill climbing algorithm for a similar problem was presented in [50].

On a more general level, spatial information processing can usually be seen as a special case of constraint satisfaction problems (CSP). A large number of evolutionary techniques have already been proposed for general CSPs. For example, [42] and [8] present overviews of genetic algorithms for this task, while [61, 72] discuss the application of ant colony optimization. Similarly, in [46] a tabu search approach to solving CSPs is proposed.

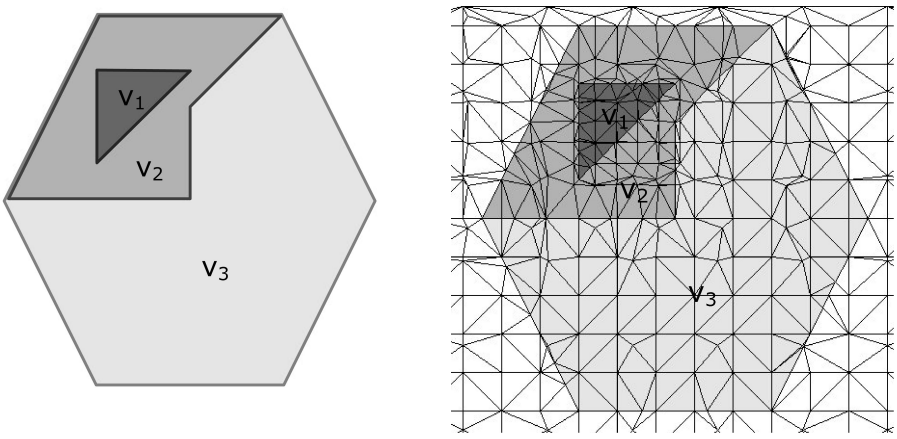
Finally, note that a preliminary version of this paper appeared in [66].

3. Constructing Polygons

3.1. Constructing One Polygon

Before we turn to the use of evolutionary algorithms, in this section we illustrate the main idea of our algorithm. In particular, we show how this problem can be seen as a *constraint satisfaction problem*, by converting available spatial information to constraints on the possible instantiations (polygons) of the variables (regions). In a first step, we consider a slightly simplified problem: given a number of regions $v_1, v_2, \ldots, v_k$ whose boundaries are known, find an appropriate polygon for region u, subject to a set of spatial constraints Θ. For the ease of presentation, we will only consider topological relations at this point. In particular, we assume that Θ contains constraints of the form $r(u, v_i)$, r being a topological relation. For example, $(NTPP \cup TPP)(u, v_2)$ indicates that the (unknown) polygon of u should be a proper part (either tangential or non–tangential) of the (known) polygon of v_2. Note, however, that the discussion below can easily be extended to other types of unary (e.g. u is convex), binary (e.g. u is north of v_1) , or n-ary (e.g. u is located between v_1 and v_2) spatial relations.

In principle, an infinite number of polygons can be considered for the instantiation of u. To make our problem tractable, we therefore need to apply some form of discretization. The most obvious choice would be to represent the plane as an infinite grid (i.e. the digital plane), and only consider polygons that can be represented as a contiguous set of cells, possibly restricted to some bounded fragment of this grid. An advantage of this technique is that the most important theoretical results of reasoning in RCC–8 remain valid for such a digital plane [38]. The problem, however, is that the polygons $v_1, \ldots, v_k$, which may be the result of quantitative information that is available a priori, are sometimes not representable in such a grid. Our solution is to discretize the plane using triangles instead. To this end, we first choose a finite set of points $\mathcal{P}$, which should at least include the vertices of the polygons corresponding to $v_1, \ldots, v_k$. In addition, $\mathcal{P}$ should contain a number of points that are chosen, randomly or uniformly, within a certain area of interest. The more points that are chosen in a certain area, the smaller the granularity of the discretization will be in this area. The actual discretization of the plane is defined by calculating



(a) (b)

Figure 3: Regions v_1, v_2, v_3 and the resulting Delaunay triangulation.

the Delaunay triangulation of $\mathcal{P}$[9] [9]. Among all possible triangulations of $\mathcal{P}$, the Delaunay triangulation is defined as the one in which triangles are as much equiangular as possible. The Delaunay triangulation is often used because it satisfies many interesting properties, which intuitively ensure that the triangulation is the most natural one. For example, the circumscribing circle of any triangle of the Delaunay triangulation contains no other points than the vertices of the triangle itself (Empty Circle property). Furthermore, every internal edge $\mathcal{E}$ is locally optimal, i.e. for $\mathcal{Q}$ the quadrilateral composed of the two triangles sharing $\mathcal{E}$, replacing $\mathcal{E}$ in the triangulation by the other diagonal of $\mathcal{Q}$ does not increase the minimum of the six internal angles. The Delaunay triangulation has been well investigated and a number of algorithms exist that allow a calculation in $\mathcal{O}(n \log(n))$ time [37, 18].

As an example, three regions v_1, v_2, v_3 and a possible corresponding triangulation are shown in Figure 3. In this figure, the points in $\mathcal{P}$ have been obtained by choosing, for each of the polygons v_1, v_2 and v_3, a fixed number of points on the boundary, inside, and in the immediate vicinity of the polygons. Hence, for instance, as v_1 is smaller than v_3, the density of the points around v_1 is higher. As possible instantiations of u, we only consider polygons that can be defined as a set of contiguous triangles from the triangulation[10].

To find a polygon, or indeed a contiguous set of triangles $\mathcal{T}$, which satisfies

[9]Note that if some of the polygons $v_1, \ldots, v_k$ would not be representable in the triangulation, this can easily be remedied by splitting some of the triangles in smaller triangles.

[10]Note that the restrictions that the triangles be contiguous is only needed if self–connected regions are desired. In some scenarios, it may be desirable to allow instantiations of variables with finite unions of disjoint polygons (e.g. to model an archipelago).

the spatial constraints in Θ, we translate topological relations to a number of simple set–theoretic restrictions on $\mathcal{T}$. In general, topological relations may result in five types of constraints on $\mathcal{T}$:

1. Upper bound: The set $\mathcal{T}$ should be a subset of $\mathcal{T}_0$, i.e. $\mathcal{T} \subseteq \mathcal{T}_0$.
2. Lower bound: The set $\mathcal{T}$ should be a superset of $\mathcal{T}_0$, i.e. $\mathcal{T}_0 \subseteq \mathcal{T}$.
3. Overlap: The set $\mathcal{T}$ should contain at least one triangle from $\mathcal{T}_0$, i.e. $\mathcal{T} \cap \mathcal{T}_0 \neq \emptyset$.
4. Partial exclusion: The set $\mathcal{T}$ may contain some, but not all triangles from $\mathcal{T}_0$, i.e. $\mathcal{T}_0 \not\subseteq \mathcal{T}$.
5. Boundary: The polygon defined by $\mathcal{T}$ should share at least one boundary point with the polygon defined by $\mathcal{T}_0$.

where $\mathcal{T}_0$ may be an arbitrary set of triangles from the triangulation. For boundary constraints, we furthermore require that $\mathcal{T}_0$ is a contiguous set of triangles, or, equivalently, that $\mathcal{T}_0$ defines a polygon. To illustrate this procedure, we consider the following example.

Example 1. *let v_1, v_2 and v_3 be defined as in Figure 3, and let Θ be defined by*

$$\Theta = \{EC(v_1, u), PO(v_2, u), TPP(u, v_3)\}$$

Furthermore, let us denote the set of triangles corresponding to the polygons for v_1, v_2 and v_3, as $\mathcal{T}_1$, $\mathcal{T}_2$ and $\mathcal{T}_3$ respectively. First, $EC(v_1, u)$ implies that (the polygons defined by) $\mathcal{T}$ and $\mathcal{T}_1$ should share at least one boundary point, and furthermore, it induces the following constraint on $\mathcal{T}$ (upper bound):

$$\mathcal{T} \subseteq \mathcal{U} \setminus \mathcal{T}_1$$

where $\mathcal{U}$ denotes the set of all triangles in the Delaunay triangulation. Next, $PO(v_2, u)$ implies that (overlap, partial exclusion, and again overlap):

$$\mathcal{T} \cap \mathcal{T}_2 \neq \emptyset$$
$$\mathcal{T}_2 \not\subseteq \mathcal{T}$$
$$\mathcal{T} \cap (\mathcal{U} \setminus \mathcal{T}_2) \neq \emptyset$$

and finally, $TPP(u, v_3)$ means that $\mathcal{T}$ and $\mathcal{T}_3$ share at least one boundary point, and that (upper bound and partial exclusion):

$$\mathcal{T} \subseteq \mathcal{T}_3$$
$$\mathcal{T}_3 \not\subseteq \mathcal{T}$$

Note that although this example is on translating topological relations to constraints on $\mathcal{T}$, the same technique can easily be applied to other types of spatial relations, such as distance or orientation relations.

Once all spatial relations from Θ have been translated to constraints on the set of triangles $\mathcal{T}$, we need to find an appropriate solution, i.e. an actual definition of $\mathcal{T}$. To this end, we propose a simple randomized greedy algorithm, although more complex algorithms, possibly including backtracking, could be conceived. This algorithm will form the starting point of our evolutionary approach below. In the standard version of the algorithm, we first select an initial triangle t^0 from $\mathcal{U}$ which satisfies all upper bound constraints on $\mathcal{T}$:

1. if at least one lower bound constraint has been obtained from Θ, t^0 is chosen randomly from the triangles in the available lower bounds;
2. otherwise, if at least one overlap constraint has been obtained, t^0 is chosen randomly from the triangles involved;
3. otherwise, t^0 is chosen randomly among all triangles that satisfy the available upper bounds.

After t^0 has been chosen, the algorithm repeatedly adds a neighboring triangle to the current set of triangles. In particular, let $\mathcal{T}^0 = \{t^0\}$, and let $\mathcal{T}^i$ be the set of triangles that has been obtained after i steps ($|\mathcal{T}^i| = \text{i+1}$). To find $\mathcal{T}^{i+1}$ from $\mathcal{T}^i$, we consider the set $\{t_1, t_2, \ldots, t_s\}$ containing all triangles t satisfying:

1. triangle t is bordering on one of the triangles in $\mathcal{T}^i$, but t is itself not contained in $\mathcal{T}^i$;
2. adding t to $|\mathcal{T}^i|$ does not violate any of the available constraints (e.g. t is contained in all the available upper bounds);
3. $\mathcal{T}^i \cup \{t_j\}$ does not correspond to a polygon with a hole (if polygons without holes are desired);

If a triangle t in $\{t_1, t_2, \ldots, t_s\}$ is contained in one of the available lower bounds, this triangle is chosen[11], i.e. $\mathcal{T}^{i+1} = \mathcal{T}^i \cup \{t\}$. If none of the triangles occurs in a lower bound, a triangle from $\{t_1, t_2, \ldots, t_s\}$ is chosen randomly. The algorithm stops when no candidate triangles, satisfying the three criteria above, can be found (failure), or when a solution has been found that satisfies all constraints (success). As an example, in Figure 4, a typical outcome of our algorithm is shown for Θ defined as in Example 1.

If a correct solution is found, we may choose to add further triangles (or remove some of the earlier assigned triangles) to obtain a more desirable polygon, e.g. with a more natural shape. In the experiments below, we will use the following strategy to add further triangles to a correct solution. Let t_j be the next triangle to be considered and let τ_j in $[0, 1]$ be an estimation of how desirable adding t_j would be. One possibility is to take $\tau_j = 0.5$ fixed. We proceed as follows:

1. Let p be randomly chosen from $[0, 1]$

[11] If there is more than one such triangle, we can choose one at random. Note that which one we choose is completely irrelevant in this case, as the remaining triangles will be added in the subsequent steps.

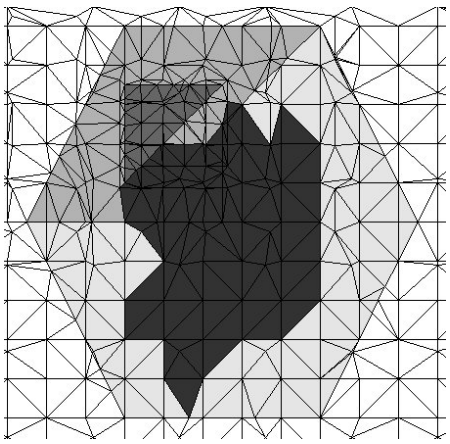


Figure 4: Possible definition of u satisfying $EC(v_1, u)$, $PO(v_2, u)$ and $TPP(u, v_3)$.

   (a) If $p < \tau_j$, add triangle t_j, i.e. let $\mathcal{T}^{i+1} = \mathcal{T}^i \cup \{t_j\}$, and repeat the process with a new triangle t'_j, chosen as explained above.
   (b) Otherwise, skip triangle t_j and repeat the process with a new triangle t'_j.
2. Repeatedly apply this strategy until either
   (a) k_{succ} successive triangles have been skipped;
   (b) or each of the candidate triangles would cause a violation of a constraint, or would result in a polygon with a hole.

3.2. Constructing Multiple Polygons

Now we turn to the original problem, where no polygons are available initially. Let $v_1, \ldots, v_k$ be variables, and let Θ be a set of spatial relations involving (only) these variables. In a first step, information that follows implicitly from the relations in Θ is made explicit. Specifically, we calculate the algebraic closure of Θ. This can be done in $O(n^3)$, n being the number of relations in Θ, using techniques that are similar to Allen's path–consistency algorithm [1]. Intuitively, such algorithms proceed by repeatedly applying composition rules (or transitivity rules). For example, from the RCC–8 composition table [56], we know that $NTPP(a, c)$ holds as soon as $NTPP(a, b)$ and $TPP(b, c)$ for some region b. Therefore, if $NTPP(a, b)$ and $TPP(b, c)$ are both contained in Θ, we can add $NTPP(a, c)$ as well. If no relations can be added to Θ in this way, Θ is called algebraically closed. By calculating the algebraic closure of Θ, implicit spatial information is made explicit, which substantially increases the effectiveness of our algorithm. Henceforth, we will denote the algebraic closure of Θ by $\widehat{\Theta}$.

Our goal is to find a model of Θ, i.e. polygons for the variables v_i which satisfy all spatial relations in Θ (or equivalently, in $\widehat{\Theta}$). Note that in general, Θ may contain a mixture of unary, binary, and n-ary relations ($n \geq 3$). To find

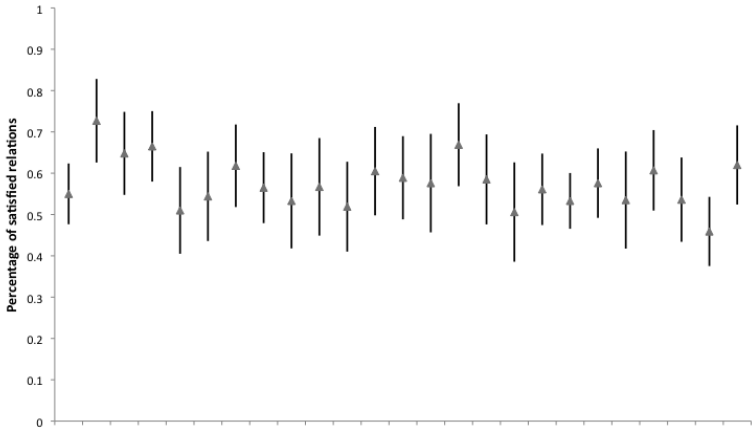


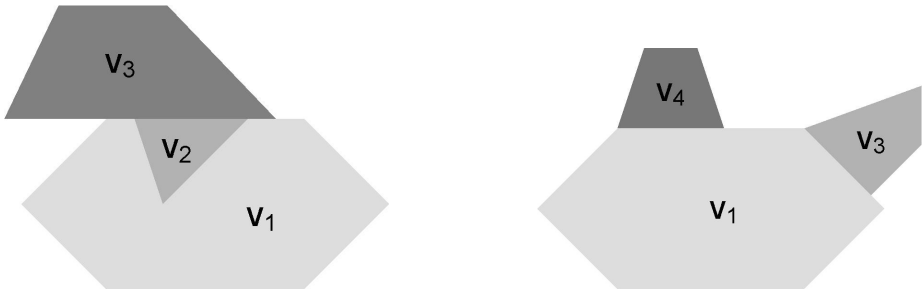
Figure 5: Mean and standard deviation of the percentage of satisfied spatial relations in scenarios generated using random variable orderings.

such a model, the technique from the previous section could be straightforwardly applied as follows. Initially, we randomly choose a polygon for v_1 that satisfies all unary relations from $\widehat{\Theta}$ (i.e. geometric constraints) involving this variable. For example, a MBB B for v_1 could be available, in which case a polygon for v_1 is constructed such that B is indeed the MBB of v_1. To find such a polygon, first a triangle t_0 is chosen randomly from the triangles that are within B, and subsequently, triangles are added until the constraint is satisfied. Next, we use the technique from Section 3.1 to find a polygon for v_2 that satisfies all unary relations from Θ involving v_2, as well as all binary relations involving v_1 and v_2 (from $\widehat{\Theta}$). Continuing in this way, in each step i, we try to construct a polygon for v_i, given the polygons for $v_1, \ldots, v_{i-1}$, which satisfies all relations from Θ that involve only variables from $\{v_1, \ldots, v_i\}$.

4. Variable Ordering

4.1. Motivation

An important characteristic of the algorithm from Section 3 is that its chances of being successful critically depend on the order in which the variables are processed. This is illustrated in Figure 5, which displays the performance of the algorithm for 25 random orderings. Each ordering is evaluated by generating a spatial scenario (i.e. polygons), and by calculating what percentage of the spatial relations in Θ are effectively satisfied. This process, for each of the 25 orderings, was repeated 25 times; Figure 5 shows the mean and standard deviations of the percentages found for a problem involving 20 variables and 20 randomly generated topological relations. Note how the best performing variable ordering (72.7% of the spatial relations satisfied on average) is substantially



(a) (b)

Figure 6: (a) After v_1, v_2 and v_3 have been instantiated, no polygon v_4 can be found anymore such that $EC(v_1, v_4)$, $EC(v_2, v_4)$ and $DC(v_3, v_4)$; (b) After v_1, v_3 and v_4 have been instantiated, a suitable polygon for v_2 can always be found.

better than the worst performing ordering (45.9% of the spatial relations satisfied on average). This dependency on the variable ordering is further illustrated in the next example.

Example 2. *Let* Θ *be given by*

$$\Theta = \{TPP(v_2, v_1), EC(v_3, v_1), EC(v_3, v_2), EC(v_4, v_1), EC(v_4, v_2), DC(v_4, v_3)\}$$

and assume that first v_1*,* v_2 *and* v_3 *are instantiated. Then it is quite likely that a scenario such as the one displayed in Figure 6(a) is obtained. In that case, no solution can be found, since* v_4 *should share at least one boundary point with* v_1 *and* v_2 *(from* $EC(v_4, v_1)$ *and* $EC(v_4, v_2)$*). However, as all boundary points that are common to* v_1 *and* v_2 *are also included in* v_3*, this would violate the requirement that* $DC(v_4, v_3)$*. Such problems could easily be alleviated if the variables were processed in the order* v_1*,* v_3*,* v_4*,* v_2*. After* v_1*,* v_3 *and* v_4 *have been instantiated, a suitable polygon for* v_2 *can always be found (see Figure 6(b)).*

As illustrated in this example, there often exists some ordering of the variables which makes finding suitable polygons trivial. Finding such an ordering deterministically, however, is difficult, since many factors may influence the optimality of a given ordering: availability of quantitative information, particularities of the given Delaunay triangulation, subtle interactions between different types of qualitative relations, etc. Even when only considering topological relations, it is not at all obvious, in general, why some orderings turn out to be better than others.

4.2. A Genetic Algorithm

Genetic algorithms (GAs) are a well–known metaheuristic, mimicking the evolution of species to tackle combinatorial optimization problems. Because of this analogy, candidate solutions are called chromosomes. A set of such chromosomes, called the population, is maintained by the algorithm, and allowed to

evolve using two problem-specific operators: crossover and mutation. Typically, a crossover operator takes two chromosomes as input (the parents) and recombines these to obtain one or more new chromosomes (the offspring), whereas a mutation operator takes one chromosome as input and alters it in some way. The sets of solutions that are repeatedly obtained are called generations of the population. To improve the average quality of the solutions found, the fitness of each chromosome, i.e. the quality of the candidate solutions, is evaluated and used to select the parents for the next generation: fitter chromosomes are more likely to be selected as parent than others.

The use of GAs to optimize orderings is well–established [52, 73], making this framework ideal for our problem. Although a large part of the crossover operators that have been proposed in the literature were developed with the traveling salesman problem in mind, some order–preserving crossover operators have been proposed that are of interest to the problem at hand, including Order Crossover (OX), Order Crossover 2 (OX2), Cycle Crossover (CX), Position Based Crossover (PBX) and Partially Mapped Crossover (PMX); we refer to [24, 47, 74] for more details on these operators. Figure 7 depicts the (average) performance of these crossover operators on 50 randomly generated problems. Each of the problems involves 20 variables and 20 topological relations. In this experiment, we used a steady–state genetic algorithm [77]: in each step only one offspring is generated, which replaces the worst individual of the population. When we refer to a generation in this context, we mean 20 of these elementary steps (e.g. 200 generations means that 4000 chromosomes have been generated), among others to allow for a fair comparison with other (genetic) algorithms. To generate an offspring, first two parents need to be selected. This selection is done using a 3–way tournament selection: we randomly choose three chromosomes from the population, and select the one with the highest fitness with probability 0.5, or one of the others, each with probability 0.25. Once two parents have been found, a crossover operator is applied with a certain probability (we used 0.25). Subsequently, a mutation operator is applied on the resulting offspring with a certain probability (we used 1). As mutation operator, we randomly swap two elements in the variable ordering corresponding to the chromosome. In our experiments, we used a population size of 20.

To evaluate the fitness of a chromosome ch_i, we apply the algorithm from Section 3 using the variable ordering encoded in the chromosome, resulting in a spatial scenario S_{ch} (i.e. polygons for every variable involved). The fitness $f(ch_i)$ of chromosome ch_i is then defined as the percentage of spatial relations from Θ that are satisfied in scenario S_{ch}:

$$f(ch_i) = \frac{|\{\gamma | \gamma \in \Theta \wedge S_{ch} \models \gamma\}|}{|\Theta|}$$

Note that in this way, due to the randomness of the procedure from Section 3, the fitness function is non–deterministic. To cope with this, and to avoid the computational overload of calculating too many scenarios, only one scenario is generated for every chromosome, which is used to assign its fitness function until the chromosome is discarded from the population.

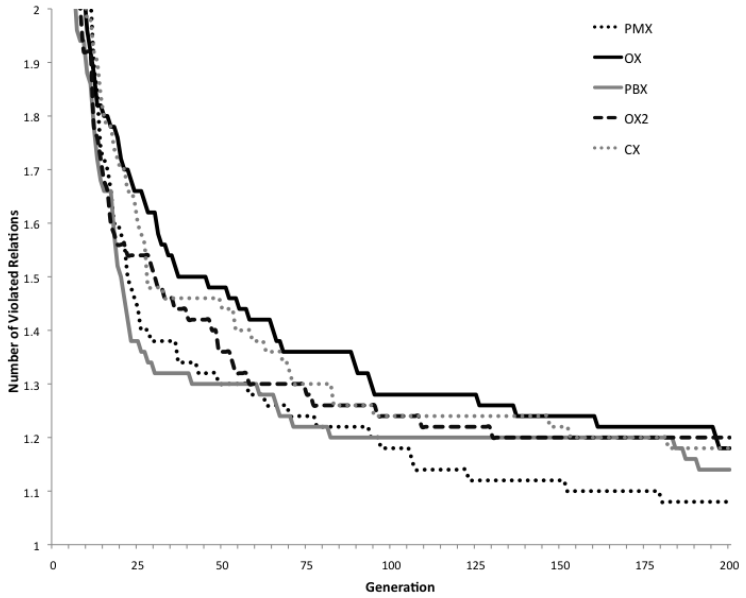


Figure 7: Performance of ordering crossover operators for generating spatial configurations.

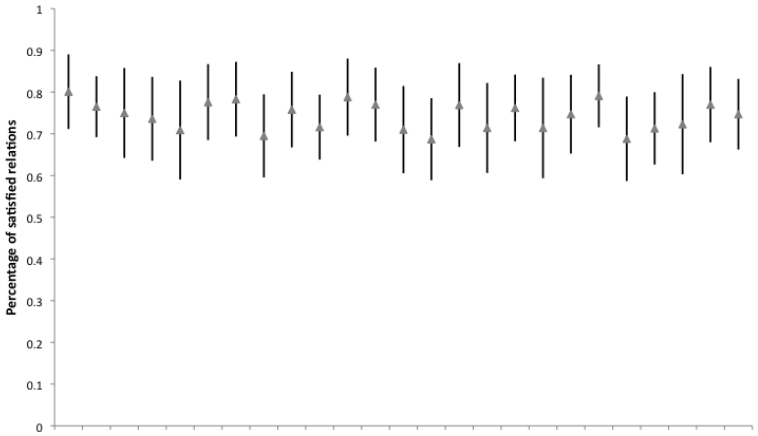


Figure 8: Mean and standard deviation of the percentage of satisfied spatial relations in scenarios generated using variable orderings that were obtained after 200 generations of a genetic algorithm.

The results from Figure 7 indicate that PMX is best suited for our purpose. Interestingly, this observation is consistent with results that have been obtained in the context of facility layout problems [28], where also optimal variable orderings are sought to generate spatial scenarios. Figure 8 presents results for the same experiment as Figure 5, but using variable orderings produced by the genetic algorithm, rather than random variable orderings. Specifically, each of the 25 orderings correspond to the best chromosome that was found after 200 generations (in 25 different runs of the genetic algorithm). In this case, the mean percentage of satisfied spatial relations ranges from 68.7% to 80.1%. This clearly shows us that the genetic algorithm improves on the quality of random orderings, resulting in spatial scenarios that satisfy more spatial relations. Note, however, that there is a rather large variability in the number of satisfied spatial relations. This implies that increases of the fitness function are a combination of the fact that better orderings are found, and the fact that a larger number of scenarios is generated, which increases the chances of being "lucky". As this latter effect is also present in simpler strategies, we compared our choice of genetic algorithm, which uses PMX crossover, to a number of alternatives in Figure 9. Specifically, we used a completely random search, a variant of our algorithm that uses no crossover operation, random–mutation hill–climbing (RMHC; [17]), and a variant that uses a generational selection scheme instead of steady–state. Unsurprisingly, random search leads to the worst performance. RMHC performs slightly better, but is clearly outperformed by a genetic algorithm without crossover, which, in turn performs similarly as the generational algorithm with crossover. Finally, the steady–state genetic algorithm with crossover performs best, so it is this algorithm that we will use in the remainder of this paper.

In addition to an optimal ordering in which the variables should be processed, we may sometimes also require an optimal ordering, for each variable, in which the spatial relations involving that variable should be considered. Note that such orderings are mainly relevant when Θ is not consistent. Indeed, if no model for Θ exists, we may still be interested in scenarios that satisfy as many of the relations in Θ as possible. To this end, we need to find out what relations are most interesting to ignore, which boils down to determining the order in which the relations should be considered. This can be done in entirely the same way as learning the optimal variable ordering.

5. Identifying Good Triangles

5.1. Motivation

In the preceding discussion we have mainly focused on topological relations, and although the algorithm can easily be generalized to other types of spatial information, there is at least one extension which is useful when quantitative information is present. To illustrate this, we will assume in this section that MBBs are available for some regions, in addition to topological relations.

Example 3. *Consider the MBBs in Figure 10(a). These rectangles give us an approximate idea of the boundaries of the unknown regions A and B: we*

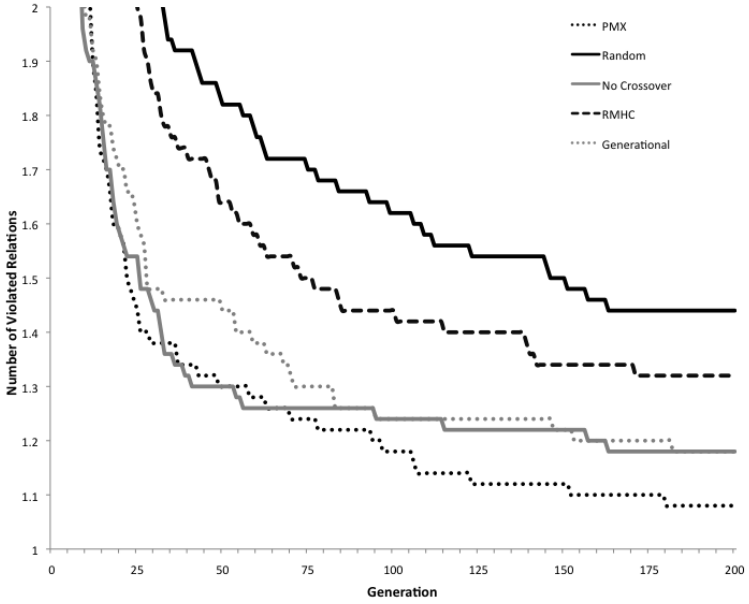
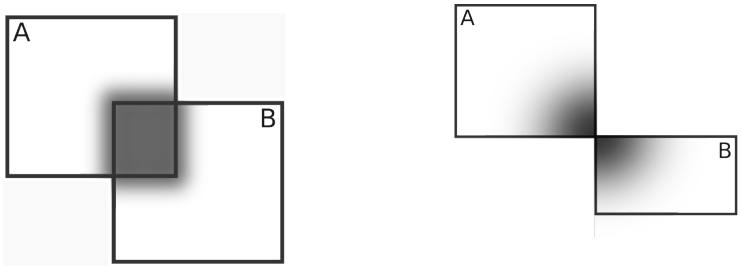


Figure 9: Comparison of the genetic algorithm with a number of baseline systems.

know that the polygons of A and B are contained in the corresponding rectangles, and that they touch each of the four sides. Such bounding box information is often available in practical applications. For example, freely available geographic knowledge bases such as the Alexandria gazetteer[12] *and the database of wikimapia*[13] *contain bounding boxes of regions, but not (usually) their actual polygons (boundaries). In terms of the constraints on triangles from Section 3.1, the information we have about the MBB of a region can straightforwardly be translated to an upper bound constraint and four boundary constraints. Now assume that we know, in addition to the MBBs from Figure 10(a), that the topological relation* $DC(A, B)$ *holds, i.e. that A and B do not share any points. Again this information can be translated to constraints on triangles. However, intuitively, from the combination of both types of information, we learn more than the union of the constraints they individually entail. In particular, the triangles in the grey zone are less likely to be contained in either of the two regions. As a second example, consider the MBBs from Figure 10(b) and assume that* $EC(A, B)$ *is known to hold. In this case, the grey zones in the rectangles for A and B are more likely to be contained in the corresponding polygons. Indeed, the only point where A and B can touch is in the lower right corner of the rectangle for A, resp. the upper left corner of the rectangle for B.*

[12] `http://www.alexandria.ucsb.edu/gazetteer/`
[13] `http://www.wikimapia.org/`



(a) $DC(A, B)$ (b) $EC(A, B)$

Figure 10: When topological information is available in addition to MBBs, some parts of the MBBs are more likely to be contained in the actual polygons than others.

Note how in both cases of the example above, our additional information is based on intuition, and expressed in terms of likelihood. It is not possible to convert such intuitions into hard constraints, and for complex scenarios it is not even feasible to make these intuitions explicit, as the available spatial information may interact in subtle ways. To cope with this, we try to learn, in an automated way, which triangles tend to appear often in good scenarios, based on the spatial scenarios that are generated by the genetic algorithm.

5.2. Ant Colony Optimization

Ant colony optimization (ACO) is a metaheuristic inspired by the foraging behaviour of ants [10]. When foraging for food, ants share their knowledge of promising paths from the nest to food sources by depositing a chemical substance called pheromones on the ground. The more pheromone on a certain path, the higher the probability that other ants will follow this same path. Such other ants then further increase the amount of pheromone on the path which, in turn, stimulates even more ants to follow it. Using this technique, ant colonies usually succeed in finding an optimal (or near–optimal) path in a short period of time. The ACO metaheuristic uses this idea to solve problems whose solutions can be built iteratively by repeatedly adding components to intermediate solutions. The likelihood that a certain component is added depends on heuristic information (if available), and experience from previous iterations. This experience is encoded by assigning a real value to each component (or to connections between components), which, in analogy, is called the amount of pheromones on that component: the more frequently a component occurs in high–quality solutions, the more pheromones will be dropped on that component, and the more likely it will appear in future solutions.

The ACO algorithm generalizes the technique from Section 3.1 for constructing polygons that satisfy a set of spatial constraints. As such, there is a tight interaction between the ACO algorithm and the GA from Section 4, where ants are essentially identified with chromosomes, and each generation of the GA corresponds to a tour of the ants. In particular, the ACO algorithm is used to

generate the spatial scenarios S_{ch} that correspond to the chromosomes ch of the genetic algorithm, and pheromone updates are performed after each generation of the GA. The details of this procedure are as follows.

5.2.1. Choosing Triangles

We assume that pheromones are dropped on the elements of the triangulation (i.e. the triangles), where a different type of pheromone is used for each variable (region). We write π_i^l to denote the amount of pheromones on triangle t_l of the type corresponding to region v_j. Now assume that, as in Section 3.1, a set of triangles $\mathcal{T}^i$ has already been chosen for region v_j and we are considering to add one of the triangles $t_1, t_2, \ldots, t_s$. Rather than choosing a triangle randomly (in the case that none of them occurs in a lower bound constraint), a bias is introduced towards more promising triangles, i.e. triangles that are more likely to be included in good solutions. In particular, the probability that triangle t_l is chosen depends on the value of $\nu_l^j = (h_l^j)^\alpha \cdot (\pi_l^j)^\beta$, where h_l^j is a heuristic value, reflecting a priori information about the likelihood that triangle t_l should be included in the polygon of region v_j, and α and β are constants in $[0, +\infty[$ that encode the relative importance of heuristic information and experience (i.e. pheromones). When no a priori information is available, we assume $h_l^j = 1$.

The triangle which is actually added to $\mathcal{T}^i$ is then defined using the ACS variant of ACO [11]: with probability λ, we add the triangle t with the highest score for ν_l^j:

$$t = \arg\max_{t_m} \nu_m^j \tag{1}$$

Otherwise (with probability $1 - \lambda$), a triangle is chosen using a roulette wheel selection, i.e. the probability of choosing t_l becomes

$$p(t_l|\mathcal{T}^i) = \frac{\nu_l^j}{\sum_{m=1}^s \nu_m^j} \tag{2}$$

The initial triangle, or equivalently the set $\mathcal{T}^1$, is obtained in a similar fashion. The only difference lies in how the candidate triangles $t_1, t_2, \ldots, t_s$ are chosen. As there are no previously assigned triangles to start from, we consider all triangles t_l, located within the available upper bound constraints, for which either h_l^j or π_l^j exceeds a certain threshold: $h_l^j > \theta_h$ or $\pi_l^j > \theta_\pi$. If there are no such triangles, we fall back on the strategy from Section 3.1.

5.2.2. Updating Pheromones

To conclude the description of the ACO algorithm, we need to specify how pheromone values are obtained and updated. Initially, the pheromone level on all triangles is equal to a small constant π_0 in $[0, 1]$. After each generation of the genetic algorithm, and for every variable v_j, each offspring ch deposits an amount of pheromones (of type j) on the triangles $\mathcal{T}_j^{S_{ch}}$ that constitute region v_j in the spatial scenario S_{ch} corresponding to chromosome ch. Note that in this way, we identify the chromosomes of the genetic algorithm with the ants of

the ACO algorithm. The amount of pheromones that are dropped on region v_j depend on the quality of the polygon for v_j in scenario S_{ch}, which we define as

$$g_j(S_{ch}) = \frac{1}{1+k} \cdot (\frac{1}{1+k_j})^2$$

where k is the number of spatial relations from Θ that are violated in S_{ch} and k_j is the number of violated spatial relations involving the variable v_j. The first factor ensures that more importance is given to spatial scenarios of a globally good quality. This means that even when all spatial relations involving v_j are satisfied, the quality of the polygon for v_j may be considered low when this comes at the cost of other spatial relations — not involving v_j — being violated. The second factor in the definition of g_j reflects that, nonetheless, the quality of v_j should mainly be evaluated based on the number of violated relations involving v_j. Our specific pheromone update scheme is based on the hypercube framework for ACO [4]. In this framework, the amount of pheromones of type j that are dropped on triangle t_l is defined in terms of

$$\Delta_l^j = \frac{\sum_{ch}\{g_j(S_{ch}) | t_l \in \mathcal{T}_j^{S_{ch}}\}}{\sum_{ch} g_j(S_{ch})}$$

In other words, the more scenarios in which t_l is contained in v_j, and the higher the quality of these scenarios, the higher the value of Δ_l^j. Finally, to forget mistakes from the past, ACO algorithms also implement some form of pheromone evaporation. This leads to the following pheromone updating rule, which is applied after each generation of the genetic algorithm:

$$\pi_l^j \leftarrow \rho\pi_l^j + (1-\rho)\Delta_l^j$$

where $\rho \in]0, 1]$.

The hypercube framework has a number of well–known advantages over alternative pheromone updating schemes [4], both theoretical (e.g. proofs of convergence) and practical (a smaller number of parameters to tune). An additional advantageous feature is that all pheromone values are contained in the unit interval $[0, 1]$, provided $\pi_0 \in [0, 1]$. Thus we can interpret pheromone values as degrees of confidence that triangles are contained in the corresponding regions. This allows us to compare the result of our algorithm to established techniques such as kernel density estimations.

5.3. Kernel Density Estimations

Heuristic information is useful to guide the ants towards promising solutions, especially in early generations when reliable pheromone maps have not yet been formed. While the specific use of heuristic information is to some extent application-dependent, in many geographic applications the use of point sets recurs as coarse approximations of region boundaries. The idea is that for each region v, a set of points $\mathcal{P}_v = p_1, \ldots, p_s$ is known which lie inside it. These points may, for instance, be obtained by looking for addresses of places that are

claimed to be in v on the web, either in textual or semi-structured form. The spatial distribution of the points in $\mathcal{P}_v$ can tell us something about the spatial extent of v. This idea is often implemented using kernel density estimations (KDEs), which provide a smooth $\mathbb{R}^2 - [0, +\infty[$ mapping K_v to approximate the spatial distribution underlying the point set $\mathcal{P}_v$ (see also Section 2.2). We will use normalised KDEs that take values in [0,1]. Specifically, the normalised version K_v^* of K_v is given for each p in $\mathbb{R}^2$ by

$$K_{v_j}^*(p) = \frac{K_{v_j}(p)}{\sup_{q \in \mathbb{R}^2} K_{v_j}(q)}$$

Given the mapping $K_{v_j}^*$, we can define the heuristic score h_l^j for triangle t_l and region v_j as the average value of $K_{v_j}^*$ over the area of the triangle:

$$h_l^j = \max(h_0, \frac{\int_{t_l} K_{v_j}^*(p)\, dp}{\int_{t_l} dp})$$

where $h_0 \in [0, 1]$ represents a default (or guaranteed) heuristic value. Note that since $K_{v_j}^*(p)$ is in $[0, 1]$ for all p in $\mathbb{R}^2$, we have that h_l^j is in $[0, 1]$ as well.

5.4. A note on complexity

From a theoretical point of view, the overall computational complexity of our approach is dominated by the $O(k^3)$ complexity (k being the number of regions) of calculating the transitive closure of the given spatial relations. Calculating the Delaunay triangulation can be done in $O(n_t \cdot \log n_t)$ with n_t the number of triangles; note that this number of triangles n_t can be seen as a constant that can be chosen independently from the number of regions. When quantitative information is available, the value of n_t will influence the coarseness of the solution, however. The complexity of the genetic algorithm, with the ant colony optimization extension, depends primarily on the complexity of generating a single individual, which is quadratic in the number of regions k. Another factor that is important is the number of triangles n_t, which determines the total number of possible scenarios. As spatial regions are obtained by incrementally adding triangles, it is clear that the practical execution time will depend strongly on this number n_t. Finally, execution time will also depend (linearly) on the total number of individuals that are generated. In practice, however, the nature of the evolutionary setting makes it possible to stop the calculation after a fixed amount of time (or after a fixed number of individuals have been generated), which allows one to search for the best possible scenario, within a given time-frame. Furthermore, the use of genetic algorithms and ant colony optimization enables the use of parallelization in a natural way.

6. Evaluation

To evaluate how successful our algorithm would be on real-world geographic data, we performed a number of experiments with the boundaries of neigh-

borhoods[14] in 10 UK cities: Cardiff (28 neighborhoods), Swansea (39 neighborhoods), London (33 neighborhoods), Birmingham (41 neighborhoods), York (11 neighborhoods), Reading (17 neighborhoods), Oxford (20 neighborhoods), Bristol (38 neighborhoods), Leicester (32 neighborhoods) and Newcastle (27 neighborhoods). Starting from the exact boundaries[15], we calculated which topological relation holds between every pair of neighborhoods. For each neighborhood, we also calculated the corresponding MBB. In this way, a set Θ was obtained containing two types of spatial relations (constraints), viz. topological relations and MBBs, on which we then applied our algorithm to find a suitable model (spatial scenario). Ideally, this model would correspond to the exact boundaries Θ was generated from. This situation is highly unlikely, however, given the limited amount of spatial information that is provided to the algorithm. In a GIR context, for instance, where the only spatial information is coming from coarse representations in gazetteers and from analyzing the content of web documents, we merely intend to acquire reasonable approximations of the real boundaries. Note that in many cases, geographic regions are inherently ill-defined and "real" boundaries do not even exist. City neighborhoods, for instance, often have many alternative definitions, which are used for specific purposes; e.g. electoral wards tend to be similar, but not identical to a parish with the same name. Thus, we are mainly interested in how useful the output of the algorithm is, rather than whether or not the correct boundaries are obtained.

We are not aware of any existing techniques that could generate spatial scenarios from such heterogeneous information. Currently, GIR methods are almost always based on MBBs or a variant of KDEs. Therefore, we will focus our evaluation on assessing (i) how our overall algorithm compares to simpler variants (e.g. without the ACO component), (ii) which parameters produce the best results, and (iii) how much our techniques can improve on what can already be accomplished using MBBs or KDEs. In particular, we focus on three different use cases, each with its own peculiarities, and with different optimal parameter settings, as will be discussed in detail below.

6.1. Experiment 1: Visualizing Spatial Relations

In this first experiment, we are only interested in finding a model that satisfies all, or as many as possible, of the spatial relations in Θ. In particular, whether or not the boundaries obtained are similar to the real boundaries is assumed to be irrelevant. This situation arises when very little quantitative information is available, and a more or less schematic representation of the available spatial information is desired as the end result, e.g. to check by visual inspection whether a given qualitative description contains errors, or to

[14] Specifically, we used the boundaries of the electoral wards in each city, except for London, where we used the boundaries of the parishes to obtain a comparable number of neighborhoods.

[15] Datasource, EDINA UK Borders, © Crown Copyright/database right 2008. An Ordnance Survey/EDINA supplied service.

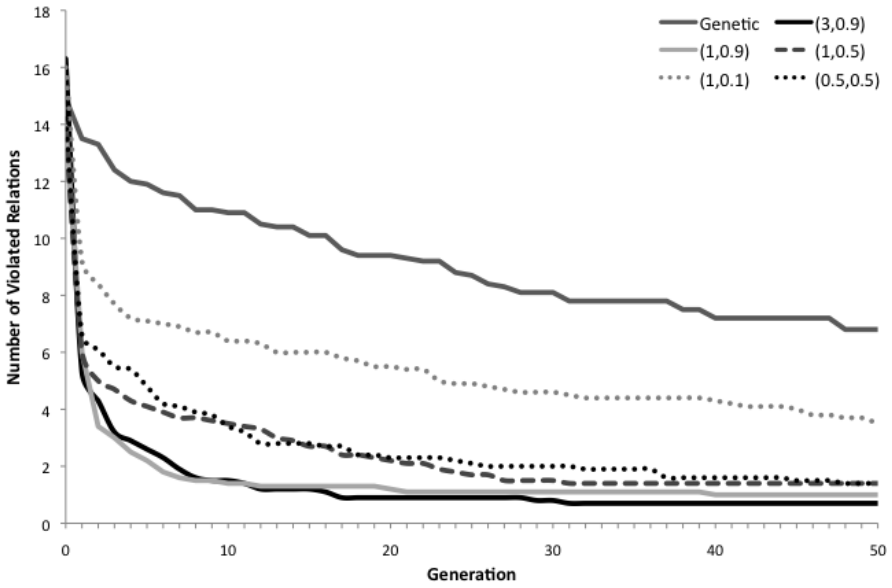


Figure 11: Number of satisfied relations during the first 50 generations, without using heuristic information and assuming that both topological relations, and MBB constraints need to be satisfied. The notation (a, b) is used to describe the configuration $\beta = a$ and $\lambda = b$.

visualise/summarize a natural language description of a scene (e.g. in a work of fiction). Compare this, for instance, with the schematic maps that are often used to depict a network of subway lines.

The results of this experiment, for a number of variants of our algorithm, are shown in Figure 11 (averaged over the 10 cities). The line labelled "genetic" corresponds to the genetic algorithm from Section 4, and is used as a baseline system here. The other lines correspond to different choices of the parameters β and λ, which reflect the sensitivity of the algorithm to the pheromones and the degree of randomness in choosing the triangles to add to intermediate solutions. The extreme value $\beta = 0$ means that the algorithm effectively ignores the pheromones, whereas $\lambda = 0$ and $\lambda = 1$ entail that, respectively, a roulette wheel and a greedy strategy are used to choose triangles. The initial amount of pheromones and the pheromone evaporation factor are kept fixed: $\pi_0 = 0.1$ and $\rho = 0.95$. For the threshold value θ_π, applied in selecting the initial triangle of a region, we use the value $\theta_\pi = \pi_0 \cdot \rho^i$ which depends on the number of the current generation i. In this way, the triangles that are initially considered are exactly those triangles that have been chosen at least once in a preceding generation (in the first generation, these triangles are chosen randomly). The best result in Figure 11 is obtained by taking β and λ sufficiently high ($\beta = 3$ and λ=0.9). Performance deteriorates when decreasing either β or λ, but all configurations perform substantially better than the genetic algorithm. Note that the genetic algorithm actually corresponds to the configuration $\beta = \lambda = 0$. To test the

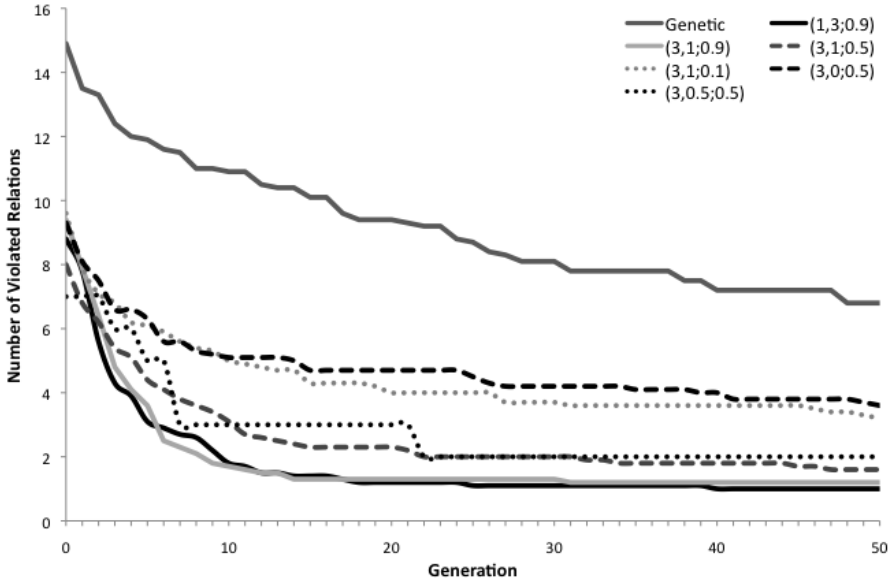


Figure 12: Number of satisfied relations during the first 50 generations, using a kernel density surface generated from 25 points (Gaussian distribution) as heuristic information, and assuming that both topological relations and MBB constraints need to be satisfied. The notation $(a, b; c)$ is used to describe the configuration $\alpha = a$, $\beta = b$ and $\lambda = c$.

statistical significance of these conclusions, we employed a (2-tailed) Wilcoxon signed-rank test to the result after 10 generations, and to the result after 50 generations. In the former case, using a p-value cut-off of 0.05, we obtain the following ordering

$$\mathit{Genetic} < (1, 0.1) < \{(1, 0.5), (0.5, 0.5)\} < \{(1, 0.9), (3, 0.9)\}$$

where we write *sysA*<*sysB* to denote that *sysB* is significantly better than *sysA*. The difference in performance between $(1, 0.5)$ and $(0.5, 0.5)$ is not statistically significant, and neither is the difference between $(1, 0.9)$ and $(3, 0.9)$. Regarding the performance after 50 generations, we found:

$$\mathit{Genetic} < (1, 0.1) < \{(1, 0.5), (0.5, 0.5), (1, 0.9), (3, 0.9)\}$$

Figure 12 depicts the result of a similar experiment, in which heuristic information was made available to help the ants find a model of the topological relations and MBBs. In this case, the configuration $(3, 0; 0.5)$ can be seen as another baseline system, in addition to the genetic algorithm, which disregards pheromones but does use the available heuristic information. These heuristic values were generated from a kernel density surface as explained in Section 5.3. The kernel density surface itself was obtained using points that were generated according to a Gaussian distribution. Specifically, the two coordinate values of

each point were chosen independently, using the middle of the MBB's projection on the X-axis (resp. Y-axis) as mean and 10% of the length of this projection as standard deviation. In this way, more points occur in the centre of the regions than at the borders. Note that a similar situation typically arises in practice, where we have a lot of information about the centre of a region, but almost no information about its outskirts. To obtain the kernel density estimations we used a Gaussian kernel, relying on Silverman's rule of thumb [69] to choose an appropriate bandwidth parameter. We used a default heuristic value of 0.1, which was also used as the threshold to choose initial triangles: $h_0 = \theta_h = 0.1$. The results from Figure 12 are slightly worse than those from Figure 11, which is somewhat surprising as one may expect that because of the heuristic information, accurate models would be found significantly faster. However, the bias introduced by the heuristic information causes the algorithm to largely ignore entire fragments of the search space. While we know that the actual boundaries cannot be found in these ignored fragments, they may very well include models of the spatial relations in Θ. In the likely situation where the actual boundaries cannot be represented using triangles from the Delaunay triangulation, the actual boundaries are, in fact, not contained in the search space. Hence, it is possible that the only models of Θ may be found in fragments that are ignored when heuristic information is added. In the case of Figure 12, this potential drawback of heuristic information (slightly) outweighs the potential advantage of faster convergence. In terms of statistical significance, we obtain

$$\mathit{Genetic} < \{(3,0;0.5),(3,0.5,0.5),(3,1;0.1)\} < (3,1;0.5) < \{(1,3;0.9),(3,1;0.9)\}$$

after 10 generations, and

$$\mathit{Genetic} < \{(3,0;0.5),(3,1;0.1)\} < (3,0.5,0.5) < \{(3,1;0.5),(1,3;0.9),(3,1;0.9)\}$$

after 50 generations.

In Figure 13, the results are shown of an experiment in which no MBBs are used. As there is less information available in this case, the boundaries that are found will be less similar to the actual boundaries. However, from the perspective of spatial reasoning, i.e. finding spatial models of Θ, the absence of MBBs means that less constraints need to be satisfied, and therefore, that convergence occurs faster. The baseline system $(3,0;0.5)$, only guided by the heuristic information, produces very similar scenarios in each generation, in which most, but not all relations are satisfied. Because it disregards previous experience (pheromones), it fails to satisfy some of the more difficult relations, repeatedly making the same "mistakes". As all the easy relations are already satisfied in the models produced by the first generation, the result does not improve in subsequent generations. The following results were found to be statistically significant:

$$\mathit{Genetic} < \{(3,0;0.5),(3,0.5,0.5),(3,1;0.1),(3,1;0.5)\} < \{(1,3;0.9),(3,1;0.9)\}$$

after 10 generations, and

$$\mathit{Genetic} < \{(3,0;0.5),(3,0.5,0.5),(3,1;0.1)\} < \{(1,3;0.9),(3,1;0.9),(3,1;0.5)\}$$

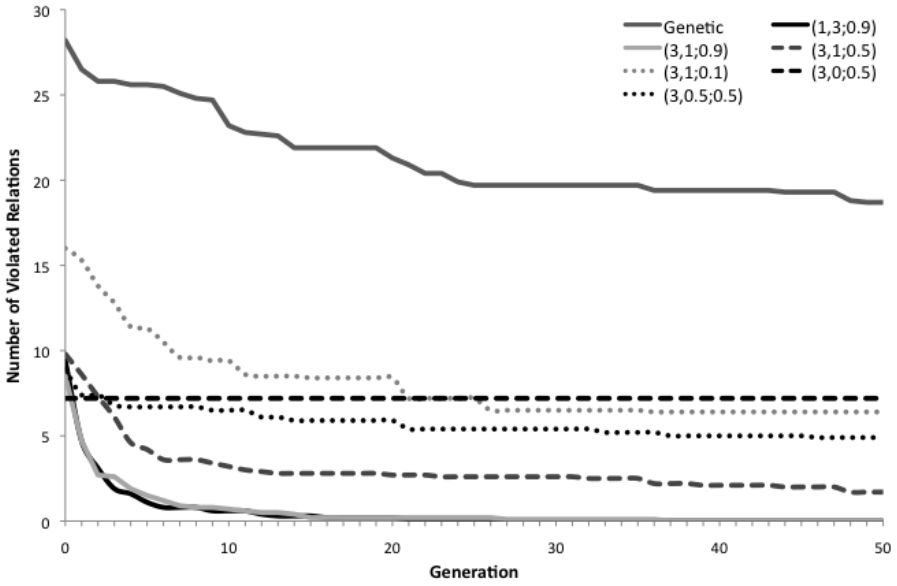


Figure 13: Number of satisfied relations during the first 50 generations, using a kernel density surface generated from 25 points (Gaussian distribution) as heuristic information, and assuming that only topological relations need to be satisfied. The notation $(a, b; c)$ is used to describe the configuration $\alpha = a$, $\beta = b$ and $\lambda = c$.

after 50 generations.

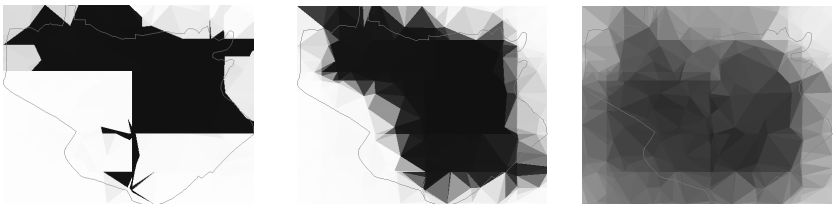
6.2. Experiment 2: Approximating Region Boundaries

In geographic applications, it may of interest to acquire boundaries that are reasonable approximations of the (unknown) true boundaries of a given region. In GIR systems such as Google Maps, for instance, approximate representations of city neighborhoods are useful to support queries like *Give me a list of hotels in the city centre of Ghent*. Whether or not a spatial scene that was generated satisfies all imposed constraints is of little value in this context, constraints are only used as a means to help the system find useful boundaries, i.e. boundaries that are as similar as possible to the actual boundaries. Therefore, in a second series of experiments, we have looked at how accurate the boundaries are that were obtained by our algorithm.

To evaluate the accuracy of region boundaries, we use the well–known Jaccard similarity measure. Specifically, let R_t be a polygon corresponding to the true boundary of a certain region R, and let R_f be the polygon that was found by our algorithm. We define the accuracy of R_f, relative to R_t, as

$$acc(R_f; R_t) = \frac{area(R_f \cap R_t)}{area(R_f \cup R_t)}$$

In other words, the accuracy is defined as the percentage of the total area occupied by R_f or R_t on which both agree. It is easy to see that $acc(R_f; R_t) = 1$



(a) $(1, 3; 0.9)$ (b) $(3, 0.5; 0.5)$ (c) $(3, 0; 0.5)$

Figure 14: Pheromone maps obtained for the Roath neighborhood (Cardiff), when only topological relations and MBBs are provided (after 50 generations).

only when R_f and R_t are identical, and $acc(R_f; R_t) = 0$ only when R_f and R_t are completely disjoint.

Table 1: Similarity between the boundaries found by the algorithm and the actual boundaries. The notation $(a, b; c)$ is used to describe the configuration $\alpha = a$, $\beta = b$ and $\lambda = c$.

	(1,3;0.9)	(3,1;0.9)	(3,1;0.5)	(3,1;0.1)	(3,0.5;0.5)	(3,0;0.5)	Genetic
MBB	0.510	0.526	0.544	0.532	**0.561**	0.479	0.483
Gauss.-25	0.290	**0.291**	0.288	0.276	0.253	0.263	0.009
Gauss.-20, Noise-5	0.290	0.262	**0.293**	0.276	0.263	0.247	0.009
Gauss.-15, Noise-10	0.258	**0.272**	0.249	0.256	0.259	0.239	0.009
Uniform-25	**0.354**	0.349	0.348	0.344	0.338	0.315	0.009
Gauss.-25 + MBB	0.555	0.565	0.567	0.540	**0.591**	0.515	0.483
Uniform-25 + MBB	0.620	**0.621**	0.615	0.576	0.617	0.593	0.483

The results are presented in Table 1, where each column corresponds to a different configuration of our algorithm (using the notations from Figure 12); the last column displays the result of the genetic algorithm, which disregards any heuristic and pheromone values, and should be seen as a baseline system. Every line of the table corresponds to a slightly different experiment. In each case, all topological relations were given to the algorithm, as well as some additional information, viz. MBBs and/or heuristic information in the form of a KDE. On the first line, the results are shown of an experiment where only MBBs and topological relations were used (cf. Figure 11). In the experiment on the second line, no MBBs were used, but heuristic information was provided in terms of a KDE generated from 25 Gaussian distributed points (cf. Figure 12). To check the robustness of the algorithm, the third and fourth line correspond to a similar experiment, in which some of the initial points (resp. 5 and 10) used to obtain the KDE were actually erroneous (but still generated using the same Gaussian distribution). Next, the algorithm was applied using KDEs generated from uniformly distributed points. Finally, for the results on the last two lines, both MBBs and heuristic information (KDEs) were used.

One important observation when comparing Table 1 to the results from Section 6.1 is that configurations which yield a good performance in the experiments of Section 6.1 do not necessarily lead to the best performance here, and

vice versa. For example, while $(1, 3; 0.9)$ turned out to be the optimal configuration in Section 6.1, it is not at all optimal in most of the experiments from Table 1. For the experiment on the first line, for example, what happens is that due to a lack of geometric information, the algorithm converges to regions that satisfy most of the available relations, but these regions may not have a very natural shape. This is illustrated in Figure 14 where the pheromone maps are depicted that were obtained for the Roath neighborhood in Cardiff. In this figure, dark regions correspond to triangles where a large amount of pheromones is found. Figure 14(a) clearly illustrates that the algorithm has converged very quickly, and that after 50 generations, no real variations occur anymore. The way by which the region touches the bottom of the MBB is very artificial, however. Because the overall scenario satisfies all available constraints, this artificial shape is not penalized by the algorithm. In fact, exactly because of this shape, there is more room to satisfy the constraints involving the neighboring regions. While this strategy led to a very good performance in Section 6.1, is does not lead to optimal boundaries. In Figure 14(c), the opposite effect occurs, i.e. as pheromone values are completely ignored, the algorithm has not converged at all, and even after 50 generations, there still is a large degree of randomness in the generation of boundaries. As it turns out, the configuration from Figure 14(b) forms the ideal trade-off between these two extremes. When heuristic information is available, the difference between $(1, 3; 0.9)$ and $(3, 0.5; 0.5)$ becomes smaller, and in some cases, $(1, 3; 0.9)$ is even better. This is mainly because the algorithm converges to a shape that is less arbitrary and therefore often less artificial, hence the fast-converging behavior of $(1, 3; 0.9)$ is not necessarily a disadvantage anymore.

As no existing techniques exist to generate boundaries from heterogeneous spatial information, assessing the overall quality of the ACO based approach is difficult. When MBBs are available, a common strategy is to use the entire MBB as an approximation of the region boundary. This strategy leads to an accuracy of 0.523 (not shown in Table 1).

In terms of statistical significance, using again the Wilcoxon signed-rank test, we obtain for the first line of the table

$$\begin{aligned}(3, 0.5; 0.5) &> \{(3, 1; 0.5), (3, 1; 0.1), (3, 1; 0.9), (1, 3; 0.9), \mathit{MBB}\} \\ &> \{\mathit{Genetic}, (3, 0; 0.5)\} \\ (3, 1; 0.5) &> \{(3, 1; 0.9), (1, 3; 0.9), \mathit{MBB}\}\end{aligned}$$

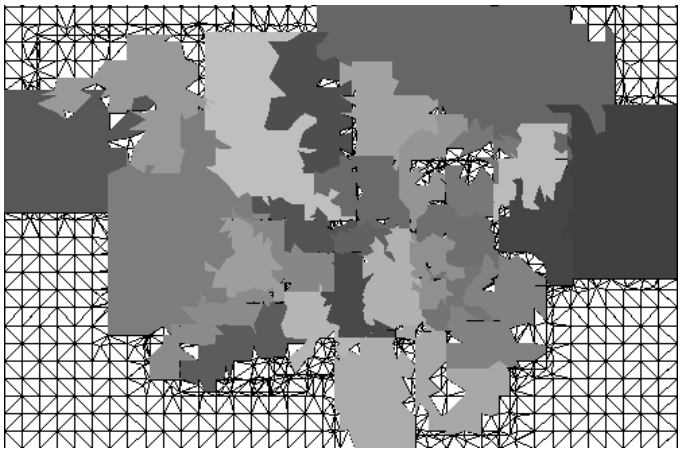
where *MBB* is the result of using the entire MBB as boundary approximation. For the second, fourth, and fifth line, we only find that the genetic algorithm is significantly outperformed by all other systems. Next, for the third line, we additionally find

$$(3, 1; 0.5) > \{(3, 1; 0.9), (3, 0; 0.5)\}$$

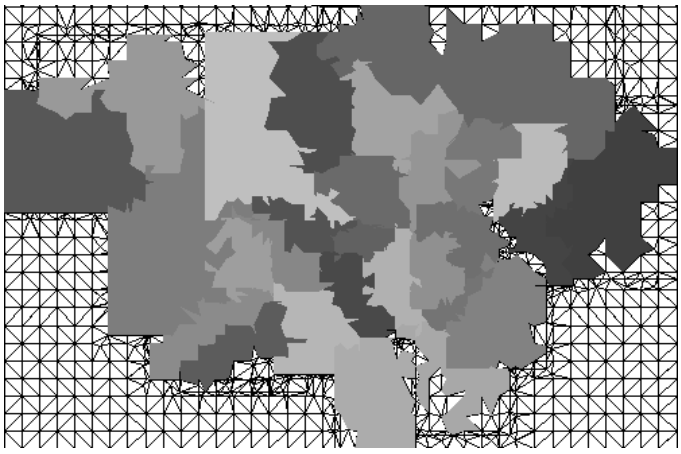
For the sixth line, we obtain

$$\{(3, 0.5; 0.5), (3, 1; 0.5), (3, 1; 0.9)\} > \{\mathit{MBB}, (3, 1; 0.1), (3, 0; 0.5)\} > \mathit{Genetic}$$

(a) Exact Boundaries



(b) $(3, 1; 0.9)$



(c) $(3, 1; 0.9)$ + partition constraint

Figure 15: Boundaries of the Cardiff electoral wards (© Crown Copyright/database right 2008. An Ordnance Survey/EDINA supplied service), as well as the approximation obtained by our algorithm starting from topological relations, MBBs, and KDEs generated from 25 uniformly distributed points.

$$(3, 1; 0.9) > (1, 3; 0.9) > \{(3, 0; 0.5), MBB\}$$

and finally, for the last line:

$$\{(3, 1; 0.9), (1, 3; 0.9), (3, 1; 0.5), (3, 0.5; 0.5)\} > N > MBB > Genetic$$
$$\{(3, 1; 0.9), (1, 3; 0.9), (3, 0.5; 0.5)\} > Y > MBB$$

When point data is available, kernel density surfaces are often used as approximations of region boundaries. A detailed comparison with this technique will be presented below. Figure 15 compares the boundaries that were obtained by the ACO algorithm with the actual boundaries, using the best configuration $(3, 1; 0.9)$ for the experiment of the last row in Table 1. Finally, note that the result in Figure 15(b) can easily be improved by adding another type of spatial information. Specifically, some of the triangles are completely surrounded by regions, but are not actually contained in any of the regions. On the other hand, because we are dealing with the electoral wards of a city, we know that such a situation does not usually occur. A useful strategy may therefore be to expand the regions, at the end of each generation, such that this kind of triangles does not occur anymore. Formally, we want the union of all regions to form a polygon without holes. The result of adding this modification to the algorithm is presented in Figure 15(c).

6.3. Experiment 3: Pheromone Maps as Vague Region Boundaries

When sufficient quantitative information is available, the ACO approach can be used to approximate region boundaries. Depending on the intended use, however, there may be alternatives to the use of polygons for representing approximate boundaries. An important disadvantage of polygons is that we cannot encode how confident we are in their shape. A polygon may be generated from very little information, in which case its exact shape is largely arbitrary, or from an abundance of highly detailed information, in which case its shape may be very reliable. Kernel density estimations, on the other hand, approximate regions from point data using a smooth surface. As such, they encode vague, gradual region boundaries, as opposed to the crisp boundaries offered by a polygon representation. Typically, such a vague region representation consists of a core, which is considered to be definitely a part of the region being modeled, together with a transition zone about which we are less confident. By increasing or decreasing the size of the transition zone, we obtain, respectively, less and more informative representations.

There is a second reason why vague region representations may be beneficial. Depending on the context, we may mainly be interested in polygons that definitely encompass the region being modeled, and care less about the resulting polygons being too large, or in polygons that are definitely contained in the region being modeled, and care less about polygons being too small. To evaluate how well the polygon R_f satisfies these two criteria, relative to the true boundary R_t, we can measure the degree to which R_f is included in R_t, and conversely, the degree to which R_t is included in R_f. The former corresponds to

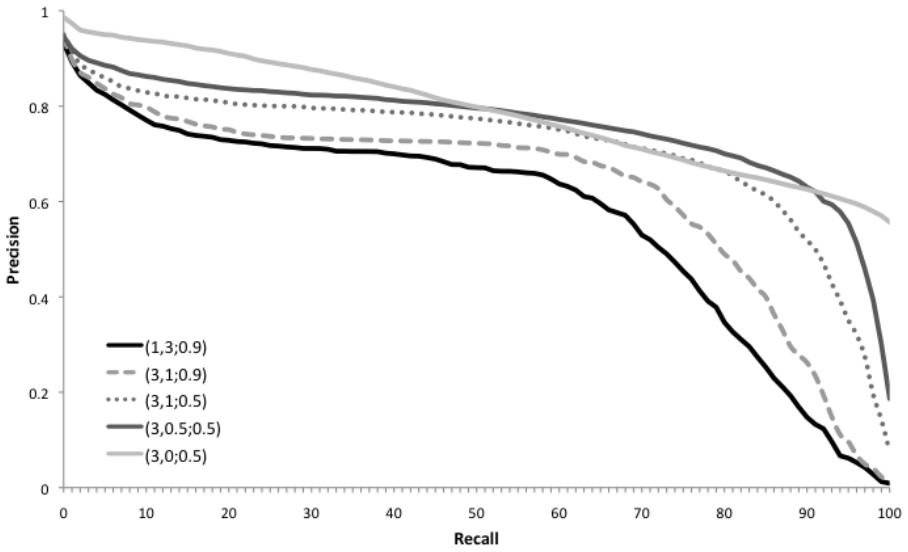


Figure 16: Precision–recall graph of the pheromone maps when only topological relations and MBBs are available.

the notion of precision from the field of information retrieval, whereas the latter corresponds to the notion of recall. Formally, we define (assuming $area(R_f) > 0$ and $area(R_t) > 0$)

$$prec(R_f; R_t) = \frac{area(R_f \cap R_t)}{area(R_f)}$$
$$recall(R_f; R_t) = \frac{area(R_f \cap R_t)}{area(R_t)}$$

Note that $prec(R_f; R_t) = 1$ iff R_f is completely contained in R_t and $recall(R_f; R_t) = 1$ iff R_t is completely contained in R_f. The situation where both $prec(R_f; R_t) = 1$ and $recall(R_f; R_t) = 1$ corresponds exactly to the situation that $acc(R_f; R_t) = 1$. A vague region can be mapped to an infinity of crisp regions, typically polygons or finite unions of polygons, by taking the α-level cuts. Specifically, we will assume that vague regions are formally represented as fuzzy sets of points, i.e. as mappings from $\mathbb{R}^2$ to $[0, 1]$. The α-level cut A_α of a fuzzy set of points (or vague region) is defined as ($\alpha \in [0, 1]$)

$$A_\alpha = \{p | p \in \mathbb{R}^2 \wedge A(p) \geq \alpha\}$$

By increasing the size of α, the value of $prec(R_f; R_t)$ will increase and $recall(R_f; R_t)$ will decrease. Hence, using this parameter α, we can control the precision–recall trade-off.

The pheromone maps produced by the ACO algorithm can be regarded as vague region representations in the sense described above. The quality of these

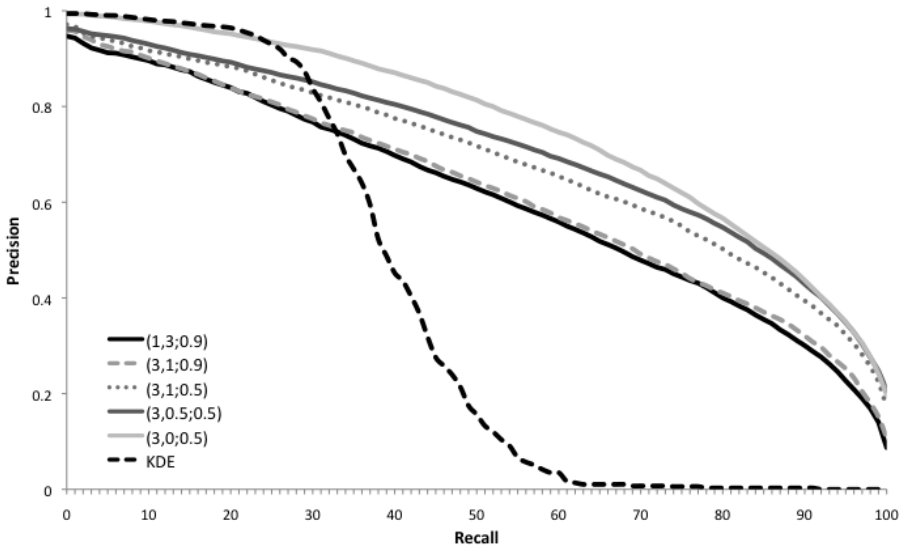


Figure 17: Precision–recall graph of the pheromone maps when only topological relations and kernel density surfaces generated from 25 points (Gaussian distribution) are available.

representations can be evaluated by looking at the corresponding precision–recall graph, which is a visual representation of the precision–recall trade-off. In Figure 16, the precision-recall graph is shown for various configurations of the ACO algorithm, in the scenario where only topological relations and MBBs are available (cf. Figure 11 and the first line of Table 1). In the case of configuration $(1, 3; 0.9)$, for instance, a recall value of 0.5 corresponds to a precision of 0.671. This means that if we want a recall value of 0.5, precision can at most be 0.671. Interestingly, the performance of the various configurations is almost the opposite of their performance in the experiments of Section 6.1. For example, the configuration $(1, 3; 0.9)$, which was optimal in Section 6.1 performs now clearly worst. In particular, it appears that slow convergence, or even no convergence at all, is a necessary condition to perform well on this task. The reason for this is clearly illustrated in Figure 14, where only the configuration $(3, 0; 0.5)$ produced a truly vague boundary. Using the other configurations, an artificially crisp pheromone map resulted, due to convergence of the algorithm.

In Figures 17–22, the results of a number of alternative scenarios are presented. Note that the relative performance of the various ACO configurations is more or less the same in all cases: $(3, 0; 0.5)$ consistently performs best and $(1, 3; 0.9)$ consistently performs worst. Interestingly, since all these scenarios are based on heuristic information in the form of KDEs, we can compare the performance of the pheromone maps with the performance of the KDEs themselves. Overall, the configuration $(3, 0; 0.5)$ substantially outperforms the KDEs, while, with the exception of scenarios where uniformly distributed points are used (Figures 18 and 20), the KDEs only outperform the other configurations

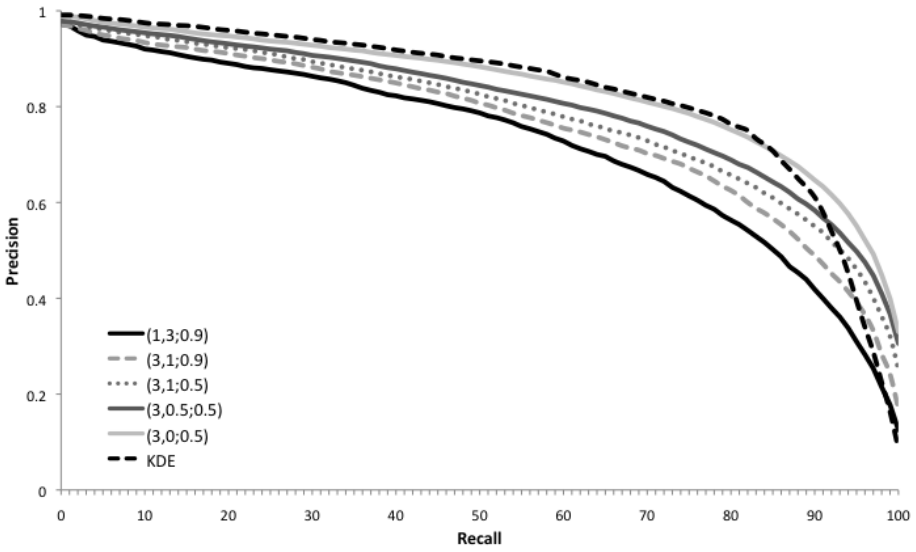


Figure 18: Precision–recall graph of the pheromone maps when only topological relations and kernel density surfaces generated from 25 points (uniform distribution) are available.

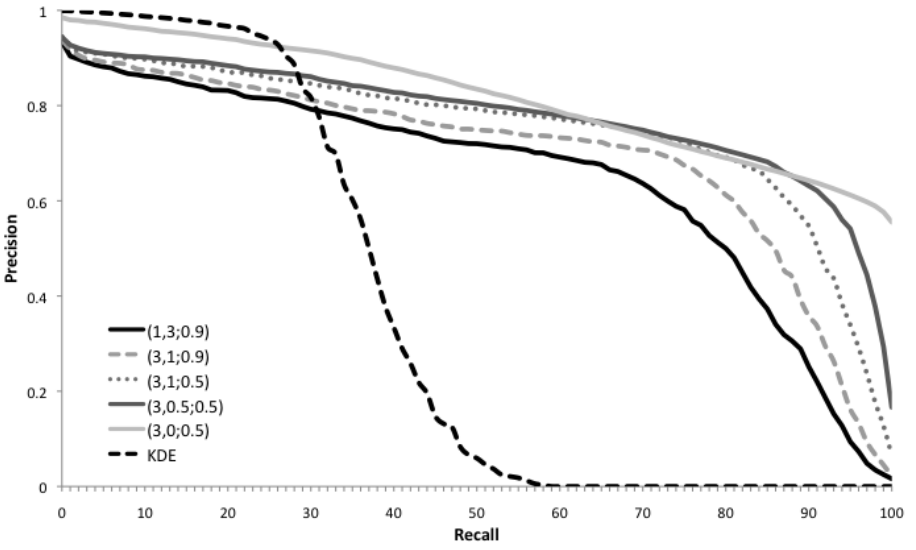


Figure 19: Precision–recall graph of the pheromone maps when only topological relations, MBBs and kernel density surfaces generated from 25 points (Gaussian distribution) are available.

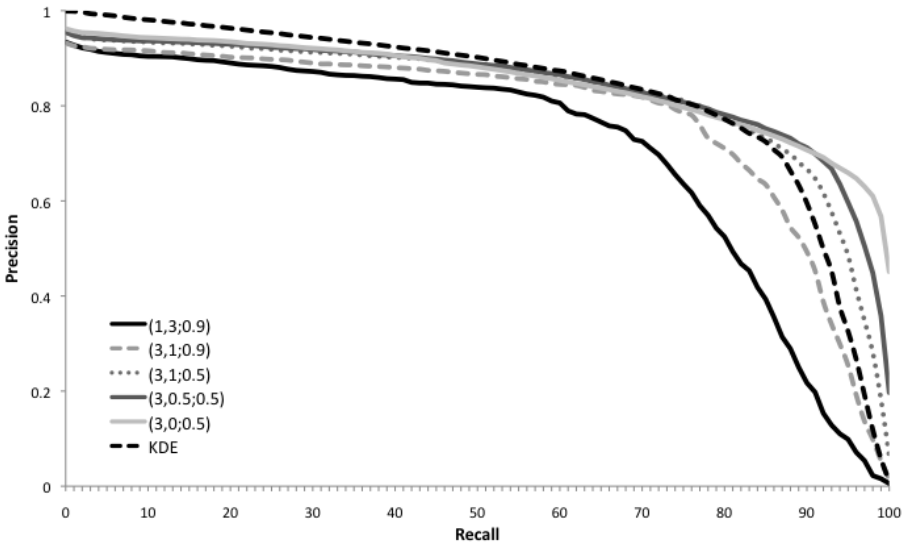


Figure 20: Precision–recall graph of the pheromone maps when only topological relations, MBBs and kernel density surfaces generated from 25 points (uniform distribution) are available.

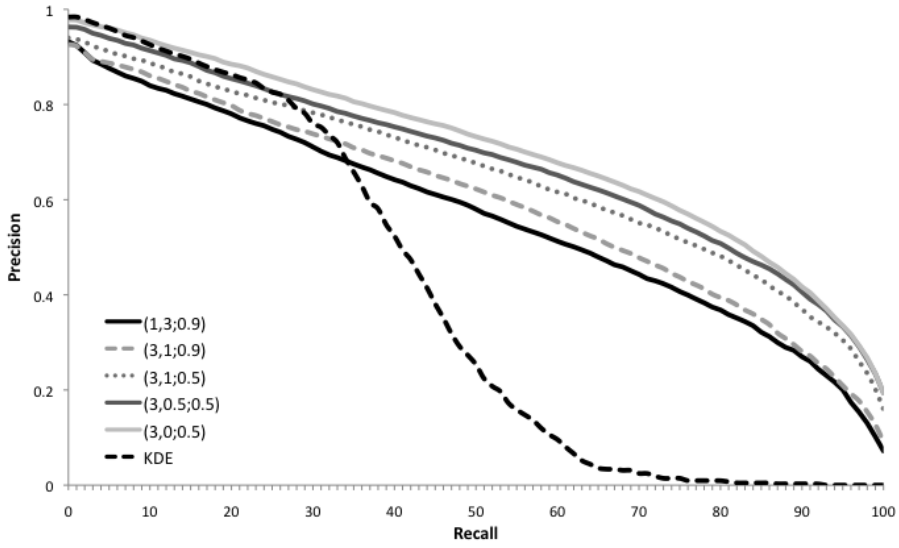


Figure 21: Precision–recall graph of the pheromone maps when only topological relations and kernel density surfaces generated from 25 noisy points (Gaussian distribution, 20 correct and 5 erroneous) are available.

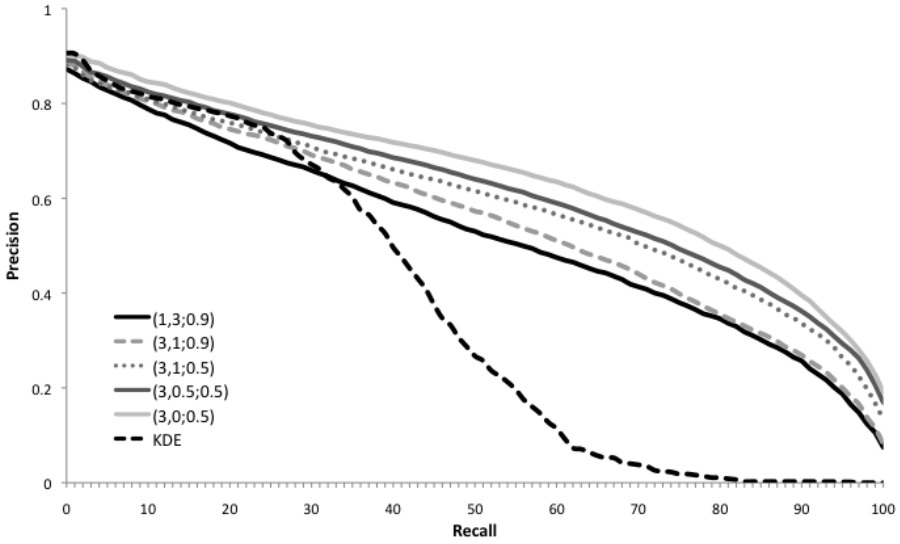


Figure 22: Precision–recall graph of the pheromone maps when only topological relations and kernel density surfaces generated from 25 noisy points (Gaussian distribution, 15 correct and 10 erroneous) are available.

at small recall levels. This result shows that for sufficiently small values of the parameter β, the ACO approach successfully combines the quantitative information from kernel density surfaces with additional spatial information that may be available. Comparing Figures 17 and 19, or Figures 18 and 20, we can notice that the availability of MBBs is especially beneficial at medium and high recall levels. Finally, as Figures 21 and 22 illustrate, the ACO approach is quite robust to errors in the available point data.

7. Conclusions

While existing techniques for processing spatial information mainly look at one type of information only, applications such as GIR are faced with a much larger diversity, often involving a mixture of qualitative (topological relations, cardinal directions) and quantitative (point data, minimal bounding boxes, polygons) spatial information. Motivated by this observation, we have proposed a methodology to process heterogeneous spatial information.

Two evolutionary strategies lie at the hearth of our solution: genetic algorithms and ant colony optimization (ACO). Genetic algorithms are used to find an optimal ordering in which variables and spatial constraints should be processed, a technique which is similar in spirit to previous work on facility layout problems [28]. Through a hybridization with ant colony optimization, our algorithm is, in addition, able to learn geometric constraints that are implicit in the available information. In particular, pheromone maps, which are generated based on previous experience, are used to encode what parts of the

plane are likely to be contained in, or excluded from a certain region. Existing techniques for processing imperfect information can be added to this algorithm as the heuristic information of the ACO algorithm. This idea was exemplified with the popular technique of generating kernel density surfaces from possibly imperfect point data.

A number of experiments were described to illustrate the potential of this technique, focusing on three possible use cases. First, our approach may be useful to obtain a visual summary of available spatial information. In this case, the focus is more on spatial reasoning, and on finding models of sets of constraints, than on finding accurate geographical models. Thus, our algorithm is used as an incomplete approach to expressive spatial reasoning. In the second use case, we investigated the possibility of generating approximate boundaries of geographic regions. Finally, the third use case extended this idea to vague region boundaries, interpreting the pheromone maps as degrees of confidence that certain points are contained in the corresponding region. Wherever baseline systems were available (e.g. kernel density surfaces in the third use case), our approach compared favorably.

Acknowledgments

Steven Schockaert was funded as a postdoctoral fellow of the Research Foundation – Flanders. Philip Smart was funded by the European commission FP6 project TRIPOD: TRI-Partite multimedia Object Description and its funding is duly acknowledged. Florian Twaroch would like to thank Ordnance Survey for funding his research on representation of place for geographic information retrieval.

References

[1] J. Allen, Maintaining knowledge about temporal intervals, Communications of the ACM 26 (11) (1983) 832–843.

[2] A. Belussi, B. Catania, E. Bertino, A reference framework for integrating multiple representations of geographical maps, in: GIS '03: Proceedings of the 11th ACM international symposium on Advances in geographic information systems, ACM, New York, NY, USA, 2003, pp. 33–40.

[3] A. Belussi, B. Catania, P. Podestà, Towards topological consistency and similarity of multiresolution geographical maps, in: GIS '05: Proceedings of the 13th annual ACM international workshop on Geographic information systems, ACM, New York, NY, USA, 2005, pp. 220–229.

[4] C. Blum, M. Dorigo, The hyper–cube framework for ant colony optimization, IEEE Transactions on Systems, Man, and Cybernetics, Part B 34 (2) (2004) 1161–1172.

[5] A. J. Brimicombe, Cluster detection in point event data having tendency towards spatially repetitive events, in: 8th International Conference on GeoComputation, Ann Arbor, Michigan (CD), 2005.

[6] C. Brunsdon, Estimating probability surfaces for geographical point data: An adaptive kernel algorithm, Computers & Geosciences 21 (7) (1995) 877–894.

[7] E. Clementini, P. Di Felice, D. Hernández, Qualitative representation of positional information, Artificial Intelligence 95 (2) (1997) 317–356.

[8] B. Craenen, A. Eiben, J. van Hemert, Comparing evolutionary algorithms on binary constraint satisfaction problems, IEEE Transactions on Evolutionary Computation 7 (5) (2003) 424–444.

[9] B. N. Delaunay, Sur la sphere vide., Izvestia Akademia Nauk SSSR,Otdelenie Matematicheskii i Estestvennyka Nauk 7 VII Seria (1934) 793–800.

[10] M. Dorigo, G. Di Caro, The ant colony optimization meta–heuristic, in: D. C. et al. (ed.), New ideas in optimization, McGraw–Hill Ltd., 1999, pp. 11–32.

[11] M. Dorigo, L. Gambardella, Ant colony system: a cooperative learning approach to the traveling salesman problem, IEEE Transactions on Evolutionary Computation 1 (1) (1997) 53–66.

[12] S. Du, L. Guo, Q. Wang, A model for describing and composing direction relations between overlapping and contained regions, Information Sciences 178 (14) (2008) 2928 – 2949.
URL http://www.sciencedirect.com/science/article/B6V0C-4S49NP7-1/2/ec9958768aee4087ecfd8a8bc4102b84

[13] S. Du, Q. Qin, Q. Wang, H. Ma, Evaluating structural and topological consistency of complex regions with broad boundaries in multi-resolution spatial databases, Information Sciences 178 (1) (2008) 52 – 68.
URL http://www.sciencedirect.com/science/article/B6V0C-4PBDR3N-1/2/42da53a316e820d8ebe0d44fb3a1b5f8

[14] M. Duckham, J. Lingham, K. Mason, M. Worboys, Qualitative reasoning about consistency in geographic information, Information Sciences 176 (6) (2006) 601 – 627.
URL http://www.sciencedirect.com/science/article/B6V0C-4H2FWXX-1/2/4f17464e8e722e9b1e825629963511f4

[15] M. Egenhofer, R. Franzosa, Point–set topological spatial relations, International Journal of Geographical Information Systems 5 (2) (1991) 161–174.

[16] L. D. Floriani, P. Marzano, E. . Puppo, Spatial queries and data models, in: I. C. A. U. Frank, U. Formentini (eds.), Information Theory: a Theoretical Basis for GIS, Springer-Verlag, Lecture Notes in Computer Science, N.716, 1992, pp. 113–138.

[17] S. Forrest, M. Mitchell, Relative building-block fitness and the building-block hypothesis, in: D. Whitley (ed.), Foundations of genetic algorithms 2, Morgan Kaufmann, 1993, pp. 109–126.

[18] S. Fortune, A sweepline algorithm for voronoi diagrams, Algorithmica 2 (2) (1987) 153–174.

[19] A. Frank, Qualitative spatial reasoning: Cardinal directions as an example, International Journal of Geographic Information Systems 10 (3) (1996) 269–290.

[20] A. U. Frank, Tiers of ontology and consistency constraints in geographic information systems, International Journal of Geographical Information Science 15 (7 (Special Issue on Ontology of Geographic Information)) (2001) 667–678.

[21] C. Freksa, Qualitative spatial reasoning, in: D. Mark, A. Frank (eds.), Cognitive and Linguistic Aspects of Geographic Space, Kluwer Academic Publishers, 1991, pp. 361–372.

[22] A. Galton, M. Duckham, What is the region occupied by a set of points?, in: 4th International Conference, GIScience 2006, vol. 4197 of Lecture Notes in Computer Science, 2006, pp. 81–98.

[23] A. Gerevini, J. Renz, Combining topological and size information for spatial reasoning, Artificial Intelligence 137 (2002) 1–42.

[24] D. Goldberg, R. Linge, Alleles, loci and the traveling salesman problem, in: Proceedings of the First International Conference on Genetic Algorithms and their Applications, 1985, pp. 154–159.

[25] R. Goyal, Similarity assessment for cardinal directions between extended spatial objects, Ph.D. thesis, University of Maine (2000).

[26] M. Grigni, D. Papadias, C. Papadimitriou, Topological inference, in: Proceedings of the 14th International Joint Conference on Artificial Intelligence, 1995, pp. 901–906.

[27] V. Haarslev, C. Lutz, R. Moller, Foundations of spatioterminological reasoning with description logics, in: Principles of Knowledge Representation and Reasoning, 1998, pp. 112–123.

[28] S. Hamamoto, Y. Yih, G. Salvendy, Development and validation of genetic algorithm-based facility layout: a case study in the pharmaceutical industry, International Journal of Production Research 37 (4) (1999) 749–768.

[29] J. Holland, Adaptation in Natural and Artificial Systems, University of Michigan Press, 1975.

[30] J. Hong, Qualitative distance and direction reasoning in geographic space, Ph.D. thesis, University of Maine (1994).

[31] C. Jones, H. Alani, D. Tudhope, Geographic information retrieval with ontologies of place, in: Proceedings of the International Conference on Spatial Information Theory: Foundations of Geographic Information Science, 2001, pp. 322–335.

[32] C. Jones, R. Purves, A. Ruas, M. Sanderson, M. Sester, M. van Kreveld, R. Weibel, Spatial information retrieval and geographical ontologies: an overview of the SPIRIT project, in: Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval, 2002, pp. 389–390.

[33] C. B. Jones, A. I. Abdelmoty, D. Finch, G. Fu, S. Vaid, The SPIRIT spatial search engine: Architecture, ontologies and spatial indexing, in: M. J. Egenhofer, C. Freksa, H. J. Miller (eds.), Geographic Information Science, Third International Conference, GIScience 2004, Adelphi, MD, USA, October 20-23, 2004, Proceedings, vol. 3234 of Lecture Notes in Computer Science, Springer, 2004, pp. 125–139.

[34] C. B. Jones, R. S. Purves, P. D. Clough, H. Joho, Modelling vague places with knowledge from the Web, International Journal of Geographic Information Systems 22 (10) (2008) 1045–1065.

[35] J. Kratochvíl, String graphs II: Recognizing string graphs is NP–hard, Journal of Combinatorial Theory, Series B 52 (1) (1991) 67–78.

[36] C. Lam, J. Wilson, D. Holmes-Wong, Building a neighborhood specific gazetteer for a digital archive, in: ESRI User Conference 2002, 2002. URL `http://gis.esri.com/library/userconf/proc02/pap0300/p0300.htm`

[37] D. Lee, B. Schachter, Two algorithms for constructing a delaunay triangulation, International Journal of Computer and Information Sciences 9 (3) (1980) 219–242.

[38] S. Li, On topological consistency and realization, Constraints 11 (1) (2006) 31–51.

[39] S. Li, Combining topological and directional information for spatial reasoning, in: Proceedings of the 20th International Joint Conference on Artificial Intelligence, 2007, pp. 435–440.

[40] K. Lunn, C. Johnson, Spatial reasoning with genetic algorithms – an application in planning of safe liquid petroleum gas sites, in: Selected Papers from AISB Workshop on Evolutionary Computing, LNCS 1143, 1996, pp. 73–84.

[41] B. Martins, M. J. Silva, S. Freitas, A. P. Afonso, Handling locations in search engine queries, in: R. Purves, C. Jones (eds.), Proceedings of the 3rd ACM Workshop On Geographic Information Retrieval, GIR 2006, Seattle, WA, USA, August 10, 2006, Department of Geography, University of Zurich, 2006.
URL http://www.geo.unizh.ch/~rsp/gir06/papers/individual/martins.pdf

[42] Z. Michalewicz, A survey of constraint handling techniques in evolutionary computation methods, in: Proceedings of the Fourth Annual Conference on Evolutionary Programming, 1995, pp. 135–155.

[43] D. Montello, M. Goodchild, J. Gottsegen, P. Fohl, Where's downtown?: behavioral methods for determining referents of vague spatial queries, Spatial Cognition and Computation 3 (2-3) (2003) 185–204.

[44] R. Moratz, C. Freksa, Spatial reasoning with uncertain data using stochastic relaxation, in: W. Brauer (ed.), Neuro–fuzzy systems, St. Augustin: Infix, 1998, pp. 106–112.

[45] B. Nebel, Computational properties of qualitative spatial reasoning: first results, in: Proceedings of the 19th German Conference on Artificial Intelligence, 1994, pp. 233–244.

[46] K. Nonobe, T. Ibaraki, A tabu search approach to the constraint satisfaction problem as a general problem solver, European Journal of Operational Research 106 (1998) 599–623.

[47] I. Oliver, D. Smith, J. Holland, A study of permutation crossover operators on the traveling salesman problem, in: Proceedings of the Second International Conference on Genetic Algorithms and their Applications, 1987, pp. 224–230.

[48] D. O'Sullivan, D. Unwin, Geographic Information Analysis, Wiley, 2002.

[49] S. E. Overell, S. M. Rüger, Identifying and grounding descriptions of places, in: R. Purves, C. Jones (eds.), Proceedings of the 3rd ACM Workshop On Geographic Information Retrieval, Department of Geography, University of Zurich, 2006.
URL http://www.geo.unizh.ch/~rsp/gir06/papers/individual/overell.pdf

[50] D. Papadias, Hill climbing algorithms for content-based retrieval of similar configurations, in: Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 2000, pp. 240–247.

[51] D. Papadias, T. Sellis, Qualitative representation of spatial knowledge in two–dimensional space, VLDB Journal 3 (1994) 479–516.

[52] P. Poon, J. Carter, Genetic algorithm crossover operators for ordering applications, Computers and Operations Research 22 (1) (1995) 135–147.

[53] A. Popescu, G. Grefenstette, P.-A. Moëllic, Gazetiki: automatic creation of a geographical gazetteer, in: R. L. Larsen, A. Paepcke, J. L. Borbinha, M. Naaman (eds.), ACM/IEEE Joint Conference on Digital Libraries, JCDL 2008, Pittsburgh, PA, USA, June 16-20, 2008, ACM, 2008, pp. 85–93.

[54] R. Purves, P. Clough, H. Joho, Identifying imprecise regions for geographic information retrieval using the web, in: Proceedings of the 13th Annual GIS Research UK Conference, 2005.

[55] R. Purves, C. Jones (eds.), A comparison of methods for the automatic identification of locations in wikipedia, Proceedings of the 4th ACM Workshop On Geographic Information Retrieval, 2007 (2007).

[56] D. Randell, Z. Cui, A. Cohn, A spatial logic based on regions and connection, in: Proceedings of the 3rd International Conference on Knowledge Representation and Reasoning, 1992, pp. 165–176.

[57] J. Renz, Maximal tractable fragments of the Region Connection Calculus: A complete analysis, in: Proceedings of the 16th International Joint Conference on Artificial Intelligence, 1999, pp. 448–454.

[58] J. Renz, A canonical model of the Region Connection Calculus, Journal of Applied Non-Classical Logics 12 (3–4) (2002) 469–494.

[59] J. Renz, B. Nebel, On the complexity of qualitative spatial reasoning: A maximal tractable fragment of the Region Connection Calculus, Artificial Intelligence 108 (1–2) (1999) 69–123.

[60] M. Rodríguez, M. Jarur, A genetic algorithm for searching spatial configurations, IEEE Transactions on Evolutionary Computation 9 (3) (2005) 252–270.

[61] A. Roli, C. Blum, M. Dorigo, ACO for maximal constraint satisfaction problems, in: Proceedings of the Fourth Metaheuristics International Conference, volume 1, 2001, pp. 187–191.

[62] S. R. Sain, On locally adaptive density estimation, Journal of the American Statistical Association 91 (1996) 1525–1534.

[63] M. Sanderson, J. Kohler, Analyzing geographic queries, in: Workshop on Geographic Information Retrieval SIGIR, 2004.

[64] M. Schaefer, E. Sedgwick, D. Stefankovic, Recognizing string graphs in NP, Journal of Computer and System Sciences 67 (2003) 365–380.

[65] S. Schockaert, M. De Cock, Neighborhood restrictions in geographic IR, in: Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 2007, pp. 167–174.

[66] S. Schockaert, P. Smart, Evolutionary strategies for expressive spatial reasoning, in: Proceedings of the Workshop on Soft Methods for Statistical and Fuzzy Spatial Information processing, 2008, pp. 26–39.

[67] S. Schockaert, P. Smart, A. Abdelmoty, C. Jones, Mining topological relations from the web, in: Proceedings of the 19th International Workshop on Database and Expert Systems Applications (FlexDBIST), 2008, pp. 652–656.

[68] J. Sharma, Integrated spatial reasoning in geographic information systems: Combining topology and direction, Ph.D. thesis, University of Maine (1996).

[69] B. Silverman, Density Estimation: For Statistics and Data Analysis, Chapman and Hall, London, 1986.

[70] S. Skiadopoulos, M. Koubarakis, Composing cardinal direction relations, Artificial Intelligence 152 (2004) 143–171.

[71] P. D. Smart, A. I. Abdelmoty, B. A. El-Geresy, C. B. Jones, A framework for combining rules and geo-ontologies, in: Web Rule Reasoning Systems - RR2007, 2007, pp. 133–147.

[72] C. Solnon, Ants can solve constraint satisfaction problems, IEEE Transactions on Evolutionary Computation 6 (4) (2002) 347–357.

[73] T. Starkweather, S. McDaniel, K. Mathias, D. Whitley, C. Whitley, A comparison of genetic sequencing operators, in: Proceedings of the fourth International Conference on Genetic Algorithms, 1991, pp. 69–76.

[74] G. Syswerda, Schedule optimization using genetic algorithms, in: L. Davis (ed.), A Handbook of Genetic Algorithms, Van Nostrand Reinhold, 1991, pp. 332–349.

[75] M. Thurstain-Goodwin, D. Unwin, Defining and delineating the central areas of towns for statistical monitoring using continuous surface representations, Transactions in GIS 4 (4) (2000) 305–317.

[76] M. Wessels, R. Moller, A flexible dl-based architecture for deductive information systems, in: IJCAR-06 Workshop on Empirically Succesful Computerized Reasoning (ESCoR), 2006, pp. 92–111.

[77] D. Whitley, J. Kauth, GENITOR: A different genetic algorithm, in: Proceedings of the Rocky Mountain Conference on Artificial Intelligence, 1988, pp. 118–130.