Determining Geographic Representations for Arbitrary Concepts at Query Time

Andreas Henrich
University of Bamberg, Germany
andreas.henrich@uni-bamberg.de

Volker Lüdecke
University of Bamberg, Germany
volker.luedecke@uni-bamberg.de

ABSTRACT

In typical concept@location-queries, the location is sometimes given by terms that cannot be found in gazetteers or geographic databases. Such terms usually describe vague geographical regions, but might also include more general terms like *mining* or *theme parks*, in which case the corresponding geographic footprint is less obvious. In the present paper we describe our approach to deal with such vague location specifications in geographic queries. Roughly, we determine a geographic representation for these location specifications from toponyms found in the top documents resulting from a query using the terms describing the location.

In this paper we describe an efficient process to derive the geographic representation for such situations at query time. Furthermore, we present experiments depicting the performance of our approach as well as the result quality.

Our approach allows for an efficient execution of queries such as *camping ground near theme park*. It can also be used as a standalone-application giving a visual impression of the geographic footprint of arbitrary terms.

Categories and Subject Descriptors

H.3.1 [**Content Analysis and Indexing**]
H.3.3 [**Information Search and Retrieval**]: Query formulation, Search process

General Terms

Algorithms, Design, Experimentation, Performance

Keywords

Vague geographic regions, Geographic search engines

1. INTRODUCTION

In Geographic Information Retrieval, textual user queries in the format concept@location are quite common. While the concept-part usually refers to any real-world object, the location-part mostly consists of a certain place or a region, which defines a geographic constraint of the query. Examples are *hotels in Bamberg* or *jobs in southern Germany*. A geographic search engine processing such a query therefore has to be able to know the position or the boundary of the given location, which is usually done by looking it up in a gazetteer-like database [12]. Places not contained in there thus cannot be found that way.

Gazetteers usually contain information about cities, locations or regions with distinct boundaries, like administrative regions. In everyday language you can find many more region names, many of which are only vaguely or ambiguously defined and do not have strict borders.

In this paper we outline an approach that uses knowledge from the world wide web to automatically determine any location that is not yet in the database of a search engine at query time, so that any user query in the above format could be answered. While the location-part of a query is so far considered to be geographic in nature, it does not necessarily have to be, which is the second proposition we make in this paper: a geographic reference in a query must not be restricted to a place or a (vernacular) region, but may be any concept, for which the search engine then has to find out an appropriate corresponding geographic extension. Examples of such queries are *camping ground near theme park* or maybe *cycle path near brewery*.

The paper is structured as follows: Section 2 covers related work. Our approach is explained in section 3, while the usage of arbitrary terms as location-identifier is covered separately in section 4. In section 5 we give a short evaluation of the quality and performance of the system. Finally, section 6 concludes the paper and discusses future work.

2. RELATED WORK

Gazetteers or geo-ontologies form the geographical data base of most geographic search engines [15]. They usually contain a place-name, the type of place, and a geographic footprint representing its location or its geographical extent [12]. This data is used for query disambiguation, query term expansion, relevance ranking or simply toponym detection in text [14]. A review of these functions can be found in [13].

Unfortunately, not all geographic references used in queries can be found in those gazetteers, so that terms used for a geographic constraint cannot be matched to a certain region.

The boundaries of these regions are also often only vaguely defined [4, 9]. Apparently, there is a need for identifying the boundaries of vague regions.

Several different formalisms have been proposed as an appropriate representation for vague regions, including supervaluation semantics [4, 16, 3], pairs of non-fuzzy sets [5, 6] and fuzzy footprints [10, 23, 20].

In [1] regions are approximated based on Voronoi diagrams which are constructed by a set of points known to be inside the target region and another one with points outside the region, but [1] does not deal with the source of the information about the sets. A similar approach can be found in [25], where regions are represented by indeterminate boundaries with an upper and a lower approximation, which is also based on given sets with points and regions inside/outside the target region.

Methods for (semi-)automatically creating representations for vague regions usually mine the knowledge contained in the world wide web. The following approaches use documents retrieved from the www with certain queries or phrases containing the targeted region. Toponyms contained in these documents are extracted and disambiguated [18, 17] and provide the data for the algorithms.

The authors of [20] use kernel density surfaces for the representation of the imprecise regions. [21] present two approaches to compute representations of regions, based on evidence of points that are likely to lie inside or outside this region. [2] modifies the algorithm from [1]. They present methods to obtain locations inside the target region and locations outside and use trigger phrases and patterns for the document retrieval to improve precision. [23] also make use of regular expressions to acquire place names by a web search. They present a method for automatically determining vague footprints, represented as fuzzy sets.

A side aspect of [24] deals with visualizations of geotagged Wikipedia articles.

Some approaches focus on small regions or domain specific regions. [22] presents a technique to construct representations of the spatial extent of neighbourhoods. [19] deals with the usefulness of different information resources depending on the size of the region.

While the basic principles of our approach are quite similar to some of the approaches above, we focus on integrating these mechanisms into the query process. Performance is a big issue therefore. While using phrases aiming at finding geographic terms seems to lead to promising results, we try to evaluate the application of these methods to any given term. Thus, we want to be able to provide a representation or a map for arbitrary terms, which might be a geographic region or something completely different, like *mining* or *walking*, in which case the result should be a geographic representation of locations associated with that particular term.

3. OUR APPROACH

In this section we outline a system architecture and its components for a search engine that is able to automatically compute geographic representations for arbitrary terms at query time. Since we want to focus on delimiting geographic regions while processing a geographic query and not on standard text search engines, we assume the following whenever we refer to a geographic search engine:

- This search engine has indexed a good-sized part of the world wide web, as every major search engine has, since we need the www-knowledge for determining geographic regions.
- This search engine is able to handle standard textual queries and to provide relevance rankings.
- Users can enter textual queries in the format concept@location. Whether this is done by two separate textfields or by a more sophisticated analysis of the queries does not matter for our purposes.

We will provide the necessary details on how we actually handle this for our experiments at the corresponding passages.

3.1 Aims

Since we want to delimit the query-region R_{query} at query time, the system must be able to do that very fast. So we aimed at a response time of less than one second (using a standard desktop PC). Secondly, we want to provide a geographic representation for any set of terms entered by the user and not just for terms describing a geographic region. Finally, the whole process must work completely automatically without user interaction.

Our prototype system is restricted to text in German language and to locations in Germany. While there are certain differences concerning phrases and patterns for example, the methods are applicable to any language and location data.

3.2 System Architecture

Figure 1 shows the relevant system components of our proposed geographic search engine and how they are related to each other.

Gazetteer A gazetteer provides the location data of known places (and regions) to the search engine.

Region-Engine The Region-Engine comprises all specific methods for delimiting vague or unknown regions. If a location from a concept@location-query is not contained in the gazetteer, the Region-Engine uses location information from the gazetteer for determining the boundaries of the unknown location. We will cover it later in this section in detail.

Ranking The ranking process of the geographic search engine has to use distances of geographic coordinates and regions rather than ontological connections between places for the ranking process. The geographical similarity is thus computed by degree of overlap or a distance measure between the query region and the geographic footprint of the document.

Region-Cache Caching will improve the performance of the system, but has no other specific function.

Region-Log It certainly makes sense to log the location terms entered by users of a live system. Frequently used regions should probably be manually revised and added to the database / gazetteer of the system.

User-Interface In addition to any existing user interface, we recommend giving the user feedback as to what was actually considered his query-location. Ways of providing some kind of relevance feedback are not discussed in this paper, though, but this is future work.

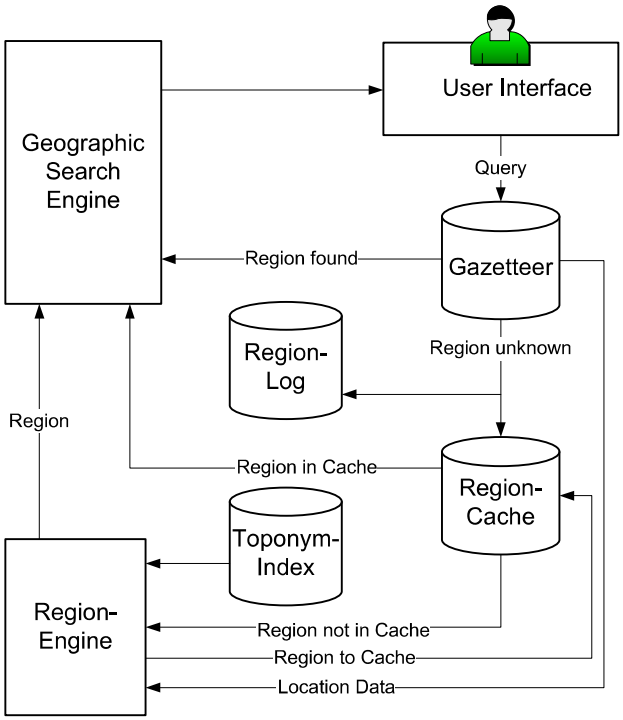

Figure 1: Proposed system architecture

Figure 2: Visualization of the concept *Weisswurstaequator*, a common word for the Main River (highlighted for comparison)

3.3 Region-Engine

The Region-Engine is the main focus of our work. It will take one or more terms and try to determine a corresponding geographic region, whenever that region is used in a query and cannot be found in the gazetteer used by the system or in the cache of previously delimited regions.

The first step is to retrieve documents $D_{retrieved}$ relevant to the query-region R_{Query}. Since our prototype system does not have indexed enough web documents yet, we used the Google-API to get the first 500 results (html-pages only) to each query-region and archived them locally. We used the first k of these documents as $D_{retrieved}$, while k is a design parameter. That way we can better evaluate the total performance of our approach, since a working geographic search engine could simply use its own (local) data.

3.3.1 Toponyms

A time consuming process step is to detect and disambiguate toponyms in text.

We used the GeoNames gazetteer as well as the OpenGeoDB, but found the former to lead to slightly better results, since it is more comprehensive. We then parsed a German dictionary to remove all location names that were also common German words, but if a location had a population of more than 30.000 people, we did not remove that location name. These decisions are the result of manual experiments and may not be optimal for all kinds of queries or documents. That way we had a list of about 70.000 toponyms in Germany.

All potential toponyms were extracted from the retrieved documents $D_{retrieved}$. If a toponym had more than one corresponding location, we used a simple disambiguation mechanism that chose those toponyms that resulted in the smallest bounding region over all toponyms in the document.

For a better performance at query time, we built a toponym index that contains a list of toponyms with their document-weights for each document. Since the toponym extraction should be made at index time and is usually done anyway while determining the geographic footprint of the document, the time spent on retrieving the toponyms of a document at query time is reduced to a minimum. Obviously, this is not possible if $D_{retrieved}$ is derived via the GoogleAPI at query time, but it is possible and straightforward if $D_{retrieved}$ is taken from the geographic search engine.

3.3.2 Density Surfaces

We used density surfaces for the representation of the regions, as did [20], using a kernel density estimation with a standard Gaussian function as kernel. That way, the whole target area (which is Germany in our case) is divided into tiles. We chose a tile size of about one square kilometer per tile. The ideal tile size depends on the intended application. For a Germany-wide search engine, a maximum resolution of one km seems about right.

The density surface may be used in two different ways for a relevance ranking by location or distance respectively.

First, all tiles with a value greater than a threshold value T_{min} may be considered to be part of the region, while all others are not. The resulting 2D-area may then be stored in a quadtree or any other representation the search engine uses for storing geographic footprints of documents or query footprints. The geographic similarity between the query footprint and each document footprint can then be computed by region overlap or any other distance measure, which is not the focus of this paper.

Secondly, depending on the similarity measure used, the density surface representation provides additional information about the area. The higher the values of some tiles, the more likely it is that these tiles lie near the core of the

query-region R_{Query}: although in the first instance these high values result from a number of locations that are often mentioned in documents relevant to the query-region and have no immediate causal connection to the core region, experiments show that more often than not the core of the target region indeed lies near the maximum of the density function.

While the kernel density estimation is usually computed on a cell-by-cell basis, this is very time consuming, since there are usually a lot of tiles with values only slightly above zero. We therefore took a different approach and iterated over all location names found in the documents. For each location we computed the corresponding fraction of the total density function for only a number of tiles, as long as the contributing value was above a (low) threshold and added it to the result matrix. If, for example, *Berlin* was the only toponym found in all $D_{retrieved}$, the resulting density surface would have a single peak value in the tile where Berlin is located, falling off to all sides. We would compute the value of this peak as well as the values of a number of neighbouring tiles, leaving out the rest of Germany. When multiple toponyms are found, we compute a fraction for each of them and the resulting sum over all tiles is the result. This is more than 100 times faster than a computation cell-by-cell, as you will see in the evaluation section.

In order to get a satisfying result area A_{result}, it is necessary to determine a threshold value T_{min} for the density surface, with all tiles with a smaller value not being part of A_{result}. Experiments showed that the best threshold-value heavily depends on the query-region R_{query}. In the following we describe a way how to determine a good threshold-value automatically.

3.3.3 Training data

For evaluating the quality of an automatically computed region representation (A_{result}) we need a correct representation for comparison. For that reason, two persons diligently researched various sources of information to find out the commonly accepted borders of about 120 regions R_{user} of various sizes in Germany. When both agreed to a certain border of a region, a polygon shape was drawn in GoogleMaps and later imported by GoogleEarth to get the coordinates of the polygon. Since the borders of most of the regions are vague, the resulting polygons must be seen as an approximation, but are well suited for our purposes, because all regions were created by using the same criteria and judgements. An exemplary region can be seen in figure 3, showing the *Havelland* in its geographic representation, while Havelland is also an administrative unit, which has a slightly different border.

Even with a given 'correct' region, it is not easy to define the best threshold-value T_{min}, as there is a trade-off between coverage of the targeted region and a too large, faulty shape. Therefore, we implemented two simple measures to automatically find the best threshold-values for 39 regions. One measure takes the absolute values of the density function / the tiles into account, while the other simply counts the number of tiles. Which measure makes more sense thus depends on whether or not the actual values are used in the later ranking process or not. Let sim be the similarity between a given user-created region R_{user} and the density surface with a given threshold value T, n the number of tiles being part of the region, m the number of tiles outside the

Figure 3: Polygon representation of *Havelland* (Background ©2008 Google – Map data ©2008 PPWK, Tele Atlas)

Indicator	Correlation to T_{min}
(1) Toponyms per doc	0.604
(2) Area size	-0.369
(3) Maximum Value	0.923
(4) Sum above threshold	-0.177
(5) Total sum	-0.312

Table 1: Pairwise Correlation to T_{min}

region and Val_x the value of tile x (n and m depend on the threshold-value, of course.) The measures are:

$$sim_{bin} = 2 * n - m \tag{1}$$

$$sim_{val} = 2 * \sum_{i=1}^{n} Val_i - \sum_{j=1}^{m} Val_j \tag{2}$$

For determining the optimal threshold-value T_{min} for each region, we simply maximized sim by iterating over all (sensible) threshold-values. More sophisticated measures taking distance into account are possible, of course, but probably would not result in very different values. Further experiments in this paper refer to the usage of sim_{val}.

To derive a function to predict the optimal threshold-value T_{min} for a query-region R_{query}, we computed the correlation between several indicators and the threshold-value on basis of learning data. Indicators we used were: (1) the number of toponyms per document from $D_{retrieved}$, (2) the size of the area, represented by the number of tiles with a value greater than the threshold-value T_{min}, (3) the maximum value V_{max} of any tile in the density surface, (4) the total sum of all tile values with a value greater than the threshold-value T_{min}, and (5) the total sum of values of all tiles.

Table 1 shows the resulting pairwise correlation values. Obviously, the threshold-value T_{min} correlates quite strongly with the maximum value V_{max} of the density function, which is often greater than one, because we did not normalize the function to the number of toponyms to get a better distinction. Figure 4 shows the individual optimal threshold-values T_{min} in dependency on the maximum values for each training region R_{user}. A simple linear regression lead to the

following function for the threshold value:

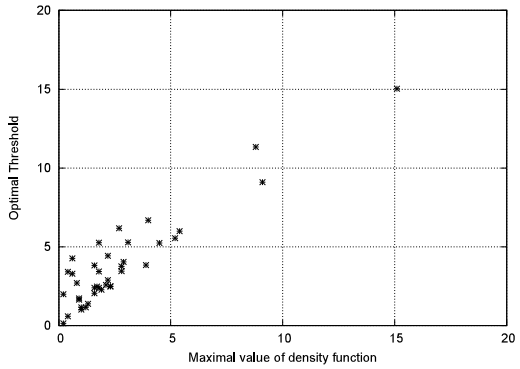

Figure 4: Correlation between V_{max} and T_{min}

$$T_{predicted} = max(0.2, 0.91 * V_{max} - 0.79) \quad (3)$$

The number of documents $D_{retrieved}$ used for delimiting a geographic region influences the performance, of course. We found that the shape of a region did not change much between 20, 100 and 500 documents. Figure 5 shows these effects by the example of the region *Harz*. The main difference between the resulting density surfaces is that the peaks get softer. For arbitrary concepts, such as *mining*, instead of real geographic regions, the number of documents seemed a lot more important, since they are not well defined and as such, a more comprehensive coverage of documents makes more sense. An argument against using too many documents is that the search results are ranked according to the relevance of the documents to the query term. The more documents we take into consideration, the less relevant the documents become to the query term. We will run experiments in the future whether the rank of a document can be used as a weight and improve the quality of the resulting region.

4. ARBITRARY TERMS AS REGIONS

There are certainly geographic information needs which result in queries that do not use toponyms as location reference. For example, a user might want to find a camping ground near a theme park or a cycle path near some breweries. In these cases, *theme park* and *brewery* would be used in much the same way as any toponym as location reference.

If a search engine was able to deal with arbitrary terms as regions using the same automated mechanism to delimit these *region-like references*, it could answer a lot more information needs. The prerequisite for that is that this approach is indeed applicable to any non-toponym terms and that the quality of the results is sufficiently high. With that, *where-is*-like queries can also be answered.

Another application of this approach is to find out whether certain terms have a significant geographic correlation. That may be region specific expressions for things (e.g. bread rolls have completely different names in several parts of Germany) or other things that are typical only for a certain region.

We applied all the mechanisms described in the preceding section to arbitrary terms.

5. EVALUATION

In this section we want to evaluate four aspects of our approach: (1) Performance and applicability of our system for delimiting regions at query time, (2) Quality of representations for geographic regions and (3) Use of 'geographizing' arbitrary terms.

5.1 Performance

As we already mentioned, the performance of the system is very important and mainly depends on the number of documents used for delimiting each region as well as the resolution of the density surface (the number of tiles used). The costs of looking up a region in a gazetteer or in the region cache thereafter are negligible, as is the cost of looking the toponyms up in the toponym index.

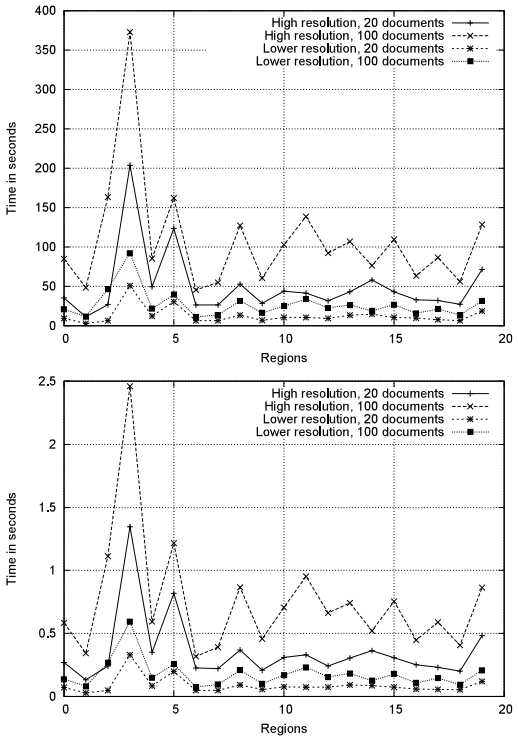

Figure 6: Performance of standard kernel density estimation (top) vs. tuned kernel density estimation (bottom) with varying resolution and number of documents $D_{retrieved}$

For performance evaluation, we computed 20 region representations and measured the time effort for each region. In figure 6 you can see the performance of building the density surface incrementally in comparison to computing it on a cell-by-cell basis, each once for 20 and once for 100 web documents $D_{retrieved}$ per query-region R_{query}. We used a tile resolution of about one square kilometer and a resolution of about four square kilometers per tile, both of which should be sufficient for the context of (not too small) regions within a Germany-wide search engine. It is clearly visible that the modification of computing the density surface improves the performance by a factor of more than 100.

Overall, the performance of our system seems to make it possible to use our approach at query time, since it usually

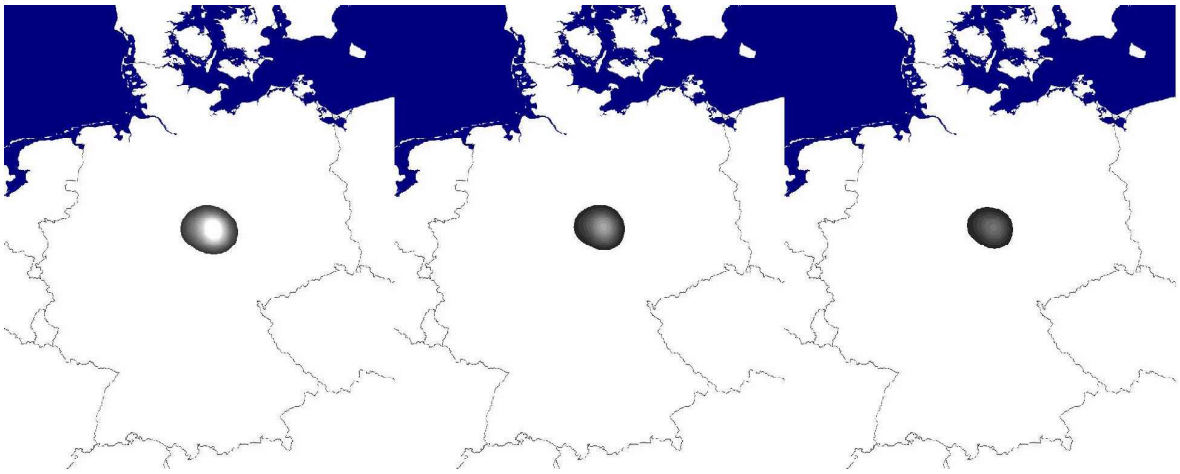

Figure 5: Region *Harz* with 20, 100 and 500 documents used

needs considerably less than one second per query representation (standard desktop PC).

5.2 Quality of region representations

There are two aspects to consider for evaluating the quality of the region representations: First, the computation of the predicted optimal threshold value $T_{predicted}$ for the density surface in comparison to the optimal threshold value for each region T_{min}, which leads to a relative evaluation. Second, an absolute evaluation of the region representation by comparing it to the target region. The latter is more difficult to compute automatically, though, and depends heavily on the general methods used. The focus of this paper clearly is the relative performance, evaluating the ability to process a query automatically and fast.

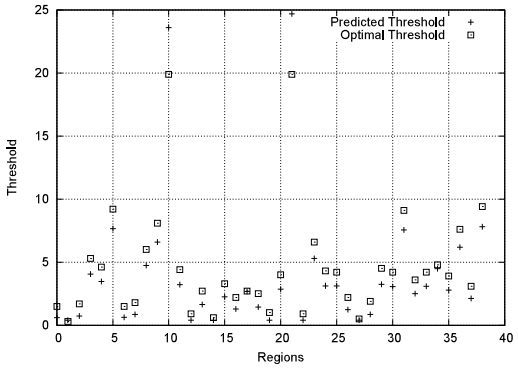

Figure 7: Predicted threshold value in comparison to optimal threshold value T_{min}

We tested formula 3 by comparing the predicted threshold values for another 39 regions R_{user} to their corresponding computed T_{min}, so we could see how good the regression function works for further regions. Figure 7 shows the results of this comparison. Obviously, the predicted values come close the the optimum.

Figure 8 shows an example of the absolute quality of the region representations. While the coverage of the Havelland is quite good here, not all representations fit that good. Especially the representation for small or longish regions are typically too large and mis-shaped. We will optimize our methods in the future, hoping to improve the overall quality of the representations.

Figure 8: Comparison of R_{user} and computed region representation for region *Havelland*

5.3 Geographizing arbitrary concepts

Evaluating the geographic representations for arbitrary terms is much more difficult, since there is usually no correct result to compare it to. We will discuss the applicability of our approach on the basis of some examples.

Weisswurst (Bavarian veal sausage) is especially popular in the south of Germany. It is often said that the border for this is the Main River, which is therefore often called the *Weisswurstaequator*, the equator of veal sausage. Geographic references like *south of the Weisswurstaequator* are commonly used in German language. Figure 2 shows the result of representing it by means of our approach. It is a bit wide, but otherwise represents the term quite correctly. Since this is more or less a geographic region, it is one of the easier examples.

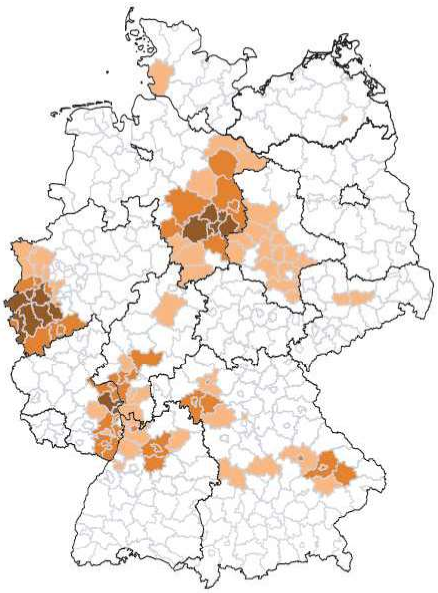

Figure 9: Cultivation of sugar beets in Germany, source: i.m.a – information.medien.agrar e.V.

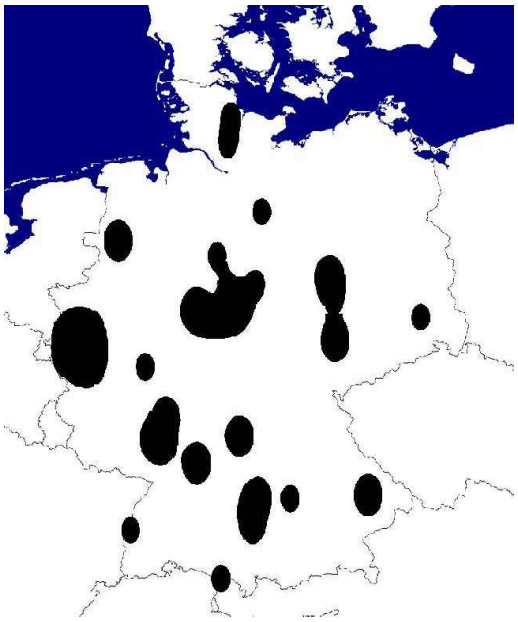

Figure 10: Cultivation of sugar beets in Germany as result area A_{result}

We found a map on the cultivation of sugar beets in Germany (see figure 9)[1]. We used the heading of that map as query-region. The resulting area of that query can be seen in figure 10, which is by no means exact, but the general idea is correct, anyway.

We discovered a project run by linguists, who manually created maps about the region-specific use of language [8].Figure 12 shows the regional distribution of various terms used for *bumper cars*. The circle highlights the region where the very uncommon word *Knuppautos* is used for that. Our automatically created representation for *Knuppautos* can be seen in figure 11.

While there were a lot of interesting and promising results, there were of course also many terms, for which we could not find a suitable geographic representation, while suitable refers to what we *expected*, since there does not seem to be a correct result to compare against, so we do not provide a numerical evaluation of the overall quality of this kind of representation. Nevertheless, we think that applying this approach to arbitrary concepts will be of some use for geographic search engines as well as standalone applications.

6. CONCLUSION AND FUTURE WORK

In this paper we described an approach how to integrate a mechanism for delimiting geographic regions efficiently into a geographic search engine and showed that the performance of our system is good enough to provide such a feature at query time and that the quality of the representations were also appropriate for that application.

In addition to that we applied this approach to arbitrary terms instead of geographic regions only and showed that it produced reasonable results, leading to interesting further applications.

Future work will certainly be to further optimize the results for geographic regions as well as for arbitrary concepts. An interesting piece of work is how to best mine the information provided by the density surface for the ranking process. We also are curious to see whether the retrieval of documents for a given query-region can be improved, for example by selecting only local pages, i.e. pages with locality [11, 7].

[1]http://www.ima-agrar.de/_redesign/Dateien/Zuckerrueben anbau_.pdf, 01.02.2008

7. REFERENCES

[1] H. Alani, C. B. Jones, and D. Tudhope. Voronoi-based region approximation for geographical information retrieval with gazetteers. *Intl. Journal of Geographical Information Science*, 15(4):287–306, 2001.

[2] A. Arampatzis, M. J. van Kreveld, I. Reinbacher, C. B. Jones, S. Vaid, P. Clough, H. Joho, and M. Sanderson. Web-based delineation of imprecise regions. *Computers, Environment and Urban Systems*, 30(4):436–459, 2006.

[3] B. Bennett. Application of supervaluation semantics to vaguely defined spatial concepts. In *COSIT 2001*, pages 108–123, London, UK, 2001. Springer-Verlag.

[4] B. Bennett. What is a forest? on the vagueness of certain geographic concepts. *Topoi*, 20(2):189–201, 2001.

[5] T. Bittner and J. G. Stell. Vagueness and rough location. *Geoinformatica*, 6(2):99–121, 2002.

[6] E. Clementini and P. D. Felice. Approximate topological relations. *Int. J. Approx. Reasoning*, 16(2):173–204, 1997.

[7] R. Eckstein, A. Henrich, and V. Lüdecke. Towards the determination of the locality of german web pages (in German). In K.-D. Althoff and M. Schaaf, editors, *LWA*, volume 1/2006 of *Hildesheimer Informatik-Berichte*, pages 69–76. University of Hildesheim, Institute of Computer Science, 2006.

Figure 11: Geographic representation for *Knuppauto*

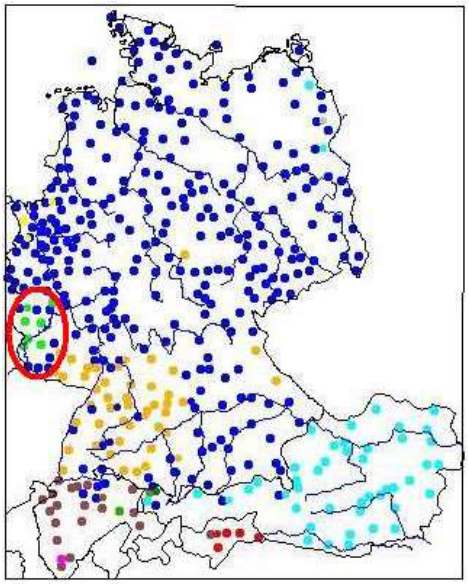

Figure 12: Geographic extension of the language use of *Knuppauto*

[8] S. Elspaß. Variation and change in colloquial (standard) german – The Atlas zur deutschen Alltagssprache (AdA) Project. *Standard, Variation und Sprachwandel in germanischen Sprachen / Standard, Variation and Language Change in the Germanic Languages*, 41:201–216, 2007.

[9] M. Erwig and M. Schneider. Vague regions. In *SSD '97: Proc. of the 5th Intl. Symposium on Advances in Spatial Databases*, pages 298–320, London, UK, 1997. Springer-Verlag.

[10] M. F. Goodchild, D. R. Montello, P. Fohl, and J. Gottsegen. Fuzzy spatial queries in digital spatial data libraries. In *Fuzzy Systems Proc.*, volume 1, pages 205–210, Anchorage, AK, USA, 1998.

[11] L. Gravano, V. Hatzivassiloglou, and R. Lichtenstein. Categorizing web queries according to geographical locality. In *CIKM '03*, pages 325–333, New York, NY, USA, 2003. ACM.

[12] L. L. Hill. Core elements of digital gazetteers: Placenames, categories, and footprints. In *ECDL '00*, pages 280–290, London, UK, 2000. Springer-Verlag.

[13] C. B. Jones, A. I. Abdelmoty, and G. Fu. Maintaining ontologies for geographical information retrieval on the web. In *CoopIS/DOA/ODBASE*, pages 934–951, 2003.

[14] C. B. Jones, H. Alani, and D. Tudhope. Geographical information retrieval with ontologies of place. In *COSIT 2001*, pages 322–335, London, UK, 2001. Springer-Verlag.

[15] C. B. Jones, R. Purves, A. Ruas, M. Sanderson, M. Sester, M. van Kreveld, and R. Weibel. Spatial information retrieval and geographical ontologies an overview of the spirit project. In *SIGIR '02*, pages 387–388, New York, NY, USA, 2002. ACM.

[16] L. Kulik. A geometric theory of vague boundaries based on supervaluation. In *COSIT 2001*, pages 44–59, London, UK, 2001. Springer-Verlag.

[17] R. Larson. Geographic information retrieval and spatial browsing. In Smith and M. Gluck, editors, *Geographic Information Systems and Libraries: Patronsand Mapsand and Spatial Information*, pages 81–124, 1996.

[18] J. L. Leidner. Toponym resolution in text (abstract only): "Which sheffield is it?". In *SIGIR '04*, pages 602–602, New York, NY, USA, 2004. ACM.

[19] R. C. Pasley, P. D. Clough, and M. Sanderson. Geo-tagging for imprecise regions of different sizes. In *GIR '07: Proc. of the 4th ACM workshop on Geographical information retrieval*, pages 77–82, New York, NY, USA, 2007. ACM.

[20] R. Purves, P. Clough, and H. Joho. Identifying imprecise regions for geographic information retrieval using the web. In *Proc. of GIS RESEARCH UK 13th Annual Conf.*, pages 313–318, Glasgow, UK, 2005.

[21] I. Reinbacher, M. Benkert, M. J. van Kreveld, J. S. B. Mitchell, and A. Wolff. Delineating boundaries for imprecise regions. In G. S. Brodal and S. Leonardi, editors, *ESA*, volume 3669 of *LNCS*, pages 143–154. Springer, 2005.

[22] S. Schockaert and M. D. Cock. Neighborhood restrictions in geographic ir. In *SIGIR '07*, pages 167–174, New York, NY, USA, 2007. ACM.

[23] S. Schockaert, M. D. Cock, and E. E. Kerre. Automatic acquisition of fuzzy footprints. In R. M. et al, editor, *OTM Workshops*, volume 3762 of *LNCS*, pages 1077–1086. Springer, 2005.

[24] J. Schöning, B. Hecht, M. Raubal, A. Krüger, M. Marsh, and M. Rohs. Improving interaction with virtual globes through spatial thinking: Helping users ask "Why?". In *Proc. of the Intl. Conf. on Intelligent User Interfaces (IUI)*, 2008.

[25] T. J. Vögele, C. Schlieder, and U. Visser. Intuitive modelling of place name regions for spatial information retrieval. In *COSIT*, pages 239–252, 2003.