Proceedings of the Twenty-Seventh Conference on Innovative Applications of Artificial Intelligence

Using Qualitative Spatial Logic for Validating Crowd-Sourced Geospatial Data

Heshan Du
University of Nottingham
Nottingham, UK
hxd@cs.nott.ac.uk

Hai Nguyen
University of Aberdeen
Aberdeen, UK
hai.nguyen@abdn.ac.uk

Natasha Alechina
University of Nottingham
Nottingham, UK
nza@cs.nott.ac.uk

Brian Logan
University of Nottingham
Nottingham, UK
bsl@cs.nott.ac.uk

Michael Jackson
Nottingham Geospatial Institute
Nottingham, UK
Mike.Jackson@nottingham.ac.uk

John Goodwin
Ordnance Survey
Southampton, UK
John.Goodwin@ordnancesurvey.co.uk

Abstract

We describe a tool, MatchMaps, that generates $sameAs$ and $partOf$ matches between spatial objects (such as shops, shopping centres, etc.) in crowd-sourced and authoritative geospatial datasets. MatchMaps uses reasoning in qualitative spatial logic, description logic and truth maintenance techniques, to produce a consistent set of matches. We report the results of an initial evaluation of MatchMaps by experts from Ordnance Survey (Great Britain's National Mapping Authority). In both the case studies considered, MatchMaps was able to correctly match spatial objects (high precision and recall) with minimal human intervention.

Introduction

Crowd-sourced or volunteered geospatial data has emerged and developed rapidly in recent years. Crowd-sourced data involves non-specialists in geospatial science, collecting, editing and displaying geospatial data. Compared to authoritative data from national mapping agencies, e.g. Ordnance Survey of Great Britain (OSGB) (Ordnance Survey 2012), crowd-sourced data, e.g. from OpenStreetMap (OSM) (OpenStreetMap 2014), is often less accurate, but could contain more recent and richer user-based information (Jackson, Rahemtulla, and Morley 2010). After it is validated by generating correspondences with authoritative data, this information can then be used to extend and enrich authoritative data.

Establishing correspondences (matches) between spatial objects is a non-trivial task. One of the challenges is that the geometry representations of the same place in different datasets are usually not exactly the same. The datasets may use different coordinate systems and/or vector representations. More importantly, the accuracy differs between datasets, and objects may be represented at different levels of granularity. As an example, consider the geometries of two objects in Nottingham city centre given by OSGB (Ordnance Survey 2012) and by OSM (OpenStreetMap 2014) shown in Fig.1. Fig.1(a) shows the representation of a restaurant *Prezzo Ristorante* in both OSGB and OSM. In OSGB, the object has geometry shown in light colour and the label 'PREZZO RISTORANTE, 21-23 FORMAN STREET, RESTAURANT'; in OSM the object has geometry shown in dark colour and the label 'Prezzo Ristorante, restaurant'. Fig.1(b) shows the representation of a shopping centre in Nottingham city centre. The OSGB dataset contains an object with geometry shown in light colour and the label 'JOHN LEWIS, 175-182, VICTORIA CENTRE, DEPARTMENT STORE'; the OSM dataset has only one object (shown in dark colour) corresponding to the shopping centre that contains this store along with several others, with the label 'Victoria Centre'. In order to match objects in the datasets, we need to determine which objects are the same and sometimes (as in the example of John Lewis and Victoria Centre) which objects in one dataset are parts of objects in another.

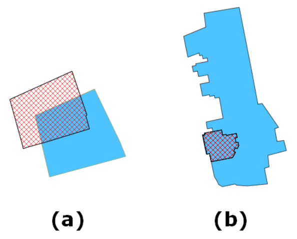

Figure 1: **a.** Prezzo Ristorante; **b.** John Lewis in Victoria Shopping Centre

A spatial object in a geospatial dataset has an ID, location information (coordinates and geometry) and meaningful labels, such as names or types, and represents an object in the real world. The problem of matching two sets of spatial objects can be stated as follows: given two sets of spatial objects $\mathcal{A}$ and $\mathcal{B}$, find the set S of all true matches: sentences of the form $sameAs(a, b)$ and $partOf(a, b)$ (where $a \in \mathcal{A}, b \in \mathcal{B}$). A sentence $sameAs(a, b)$ is true if a

and b refer to the same object in the real world. Similarly, $partOf(a, b)$ is true if the object represented by a is part of the object represented by b in the real world.

We present a tool, MatchMaps, which generates $sameAs$ and $partOf$ matches between spatial objects. A preliminary version of MatchMaps (previously called GeoMap) was described in (Du et al. 2013b); this paper presents a version with significantly modified match generation algorithms and the addition of qualitative spatial reasoning. MatchMaps works by generating candidate matches and checking their consistency with respect to several sets of sentences: ontology axioms, correspondences between type descriptions across ontologies, disjointness of types, and qualitative spatial relations between objects. If an inconsistency is found, a minimal set of statements required to derive it is produced. One of the statements in this set must be retracted to restore consistency. The decision about which statement in the minimal inconsistent set is incorrect and should be retracted is made by a human expert, as no heuristic for making this decision automatically gives sufficiently reliable results.

In this paper, we present an evaluation of MatchMaps on two case studies. We report the precision (correctness of matches) and recall (how close the set of generated matches is to the set of all true matches) of the set of matches between OSGB and OSM representations of Nottingham and of Southampton city centres relative to ground truth. In both case studies, MatchMaps was able to correctly match spatial objects (precision 90% for Nottingham and 98% for Southampton, recall 84% for Nottingham and 97% for Southampton) with minimal human intervention.

Buffered Equal and PartOf

Before we describe MatchMaps and the logical reasoning involved in it, we need to briefly introduce some relevant notions. As mentioned above, the geometries of the same objects in different datasets are not necessarily exactly the same. This is due to differences in representations, coordinate systems, and errors or imprecision in crowd-sourced data. Hence we use buffers (ISO Technical Committe 211 2003) to relax equality and the inclusion relation between geometries.

As shown in Fig. 2(a), by buffering the solid circle x by σ, we obtain a larger circle, denoted as $buffer(x, \sigma)$, where every point is within σ distance from x. A geometry a is a buffered part of (BPT) of x if all points of a are within the buffer of x (Fig. 2(b)). Two geometries are buffered equal (BEQ) with respect to σ, if they fall into the σ-buffer of each other (Fig. 2(c)). If two geometries are BEQ with respect to an appropriate level of tolerance σ, then they probably represent the same real world location, otherwise, they represent different locations.

The level of tolerance σ intuitively corresponds to how different geometry representations of the same spatial features can be. The value of σ can be established empirically by looking at the two datasets side by side and aligning geometries of several features which are known to be the same in both datasets. The maximal value required to make any two geometric representations of the same object buffered equal (BEQ) gives the value of σ for the given pair

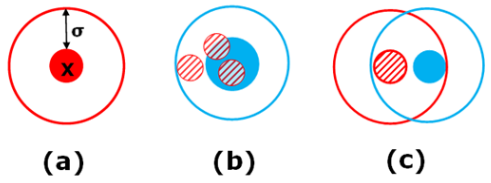

Figure 2: **a.** a buffer; **b.** three hatched circles are buffered part of (BPT) the solid circle; **c.** buffered equal (BEQ)

of datasets. For example, in the case of Nottingham city centre, the value of σ for the OSGB and OSM datasets is 20 m. Interestingly, OSM positional accuracy (maximal error) has been estimated to be about 20 m in UK (Haklay 2010).

MatchMaps

The main idea underlying MatchMaps is to generate a set of candidate matches between spatial objects in different datasets ('assumptions') and check for their logical consistency with respect to other available information. The aim is to produce a maximally consistent subset of candidate matches which is as close as possible to the set of all true matches.

The input to MatchMaps is two sets of spatial objects $\mathcal{A}$ and $\mathcal{B}$ and two ontologies $\mathcal{T}_A$ and $\mathcal{T}_B$ defining concepts for describing them. (Note that crowd-sourced datasets do not have formal ontologies, but it is easy to generate simple inclusion hierarchies from their tags.) The output is a set S of $sameAs$ and $partOf$ matches between objects.

MatchMaps comprises seven main steps summarised below.

1. **Generate disjointness axioms** between concepts in $\mathcal{T}_A$ and $\mathcal{T}_B$. A disjointness axiom is a statement of the form $C_1 \sqcap C_2 = \emptyset$ (the sets of C_1 objects and C_2 objects are disjoint; for example, $Library \sqcap Pub = \emptyset$). The disjointness axioms are generated automatically by assuming the disjointness of sibling concepts in each ontology. We also manually generate a small set of axioms that prohibit objects of one type being $partOf$ objects of another type: $School \sqsubseteq \forall partOf.\neg Pub$ (if something is a School, then for all objects it is a part of, they are not Pubs). We use the description logic reasoner Pellet (Sirin et al. 2007) to check that adding a set of disjointness and '$partOf$-disjointness' axioms $\mathcal{D}_A$ to $\mathcal{T}_A$ does not result in incoherence (existence of provably unsatisfiable concepts), similarly for $\mathcal{D}_B \cup \mathcal{T}_B$. Axioms that cause incoherence are removed from $\mathcal{D}_A$ and $\mathcal{D}_B$, resulting in $\mathcal{D} = \mathcal{D}_A \cup \mathcal{D}_B$. This does not require human interaction. This is an auxiliary step that is needed to facilitate discovering problematic matches (such as a $sameAs$ match between a and b where a is a Library and b is a Pub).

2. **Generate terminology matches** between concepts in $\mathcal{T}_A$ and $\mathcal{T}_B$ of the form $C_1 \equiv C_2$ where C_1 is a concept in $\mathcal{T}_A$ and C_2 is a concept in $\mathcal{T}_B$. Currently we generate terminology matches automatically using a very simple heuristic based on similarity of concept names. For example,

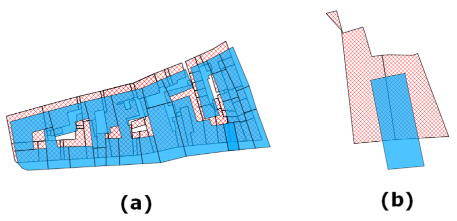

Figure 3: **a.** matching 'aggregated' geometries; **b.** matching single geometries

$OSGB : Shop \equiv OSM : Shop$. The set of terminology matches is $\mathcal{M}$. We check coherence of $\mathcal{T}_A \cup \mathcal{T}_B \cup \mathcal{D} \cup \mathcal{M}$ using Pellet. For every set of statements responsible for incoherence, we remove one of the statements in $\mathcal{D} \cup \mathcal{M}$. This step requires human interaction, because sometimes we need to decide whether to remove a terminology match or a disjointness axiom to restore coherence.

3. **Generate geometry matches** using aggregation and buffering. This is done using standard 2D spatial tools (Vivid Solutions, Inc. 2014) to aggregate, buffer and check for inclusions of their geometries. Rather than matching every single geometry in input datasets, we generate BEQ correspondences between 'aggregated' geometries, each of which is obtained by aggregating a non-empty collection of adjacent single geometries. The reason for doing the aggregation step is that sometimes matching every single geometry is impossible while there is a clear match between two aggregates. As shown in Fig.3(a), there is a clear correspondence between aggregated geometries from OSM (dark) and OSGB (light, dotted). However, for a single OSM geometry (dark), there can be more than one candidate from OSGB (dotted), as shown in Fig.3(b). We cannot decide which one is correct based only on the similarity of geometries. This type of problem often occurs when matching, for example, terraced houses or small shops in a shopping centre. This step does not require human interaction.

4. **Generate object matches** ($sameAs$ and $partOf$ matches between objects). For each pair of matched geometries from the previous step, we consider associated objects and check them for similarity of labels using a string similarity measure. In straightforward cases, when there are two objects a and b with similar geometries and similar labels in different datasets, we add $sameAs(a, b)$ to a set of candidate matches S; or if there is a set of objects $\{a_1, \ldots, a_n\}$ where the union of their geometries is similar to a geometry of a single object b in another dataset, we add $partOf(a_i, b)$ to S for every a_i. A difficult case is when there is a match between two aggregated geometries which contain objects $\{a_1, \ldots, a_n\}$ in one dataset and objects $\{b_1, \ldots, b_k\}$ in another dataset (many-to-many matching case). When we cannot decide the exact matches automatically using labels and types of objects, we generate all matches which are possibly correct between the objects in the two sets: for each pair a_i, b_j with similar labels, we generate $sameAs(a_i, b_j)$, $partOf(a_i, b_j)$, $partOf(b_j, a_i)$. The output of this step is the set S of candidate matches. This step does not require human interaction.

5. **Validate matches using LBPT**. Check S for consistency using a qualitative spatial logic, Logic of ParT and whole for Buffered geometries (LBPT), which is explained in the next section. If an inconsistency is found, we retract $sameAs$ or $partOf$ matches from S to restore consistency. This step is implemented using a dedicated LBPT reasoner and an Assumption-Based Truth Maintenance System (ATMS). This step may require human interaction to decide which matches to remove.

6. **Validate matches using UNA/NPH**. Check S for consistency with respect to UNA or NPH. UNA refers to Unique Name Assumption: for each dataset, $sameAs(a_1, a_2)$ does not hold for a_1 and a_2 with different IDs (each object is represented exactly once). NPH (No PartOf Hierarchy) is a stronger assumption that states that there are no objects b_1, b_2 in the same set such that $partOf(b_1, b_2)$ holds. UNA and NPH hold for the OSGB data. However UNA or NPH can be violated in the OSM data. Therefore this check is 'soft': if in a crowd-sourced dataset some object is represented twice, or there is a genuine $partOf$ relationship determined by human checking, we skip this 'error' and do not retract any assumptions. This step is required since even after consistency checks in the previous steps, there may be 'too many' matches in S. It is implemented using Pellet, and requires human interaction. This is an optional step, which could be skipped if UNA or NPT is violated frequently in at least one input dataset.

7. **Validate matches using classification**. Check for consistency of S together with $\mathcal{T}_A \cup \mathcal{T}_B \cup \mathcal{D} \cup \mathcal{M}$. Restore consistency and return the resulting set S. This step requires human interaction since either a match in S or a disjointness axiom may be wrong.

In addition to Pellet, the LBPT reasoner and the ATMS, the implementation of MatchMaps builds on a number of existing tools. The JTS Topology Suite (Vivid Solutions, Inc. 2014) is used to process two dimensional geometries, and the graphical user interface is implemented using the OpenJUMP libraries (JPP 2014).

Qualitative Spatial Logic

In this section we outline the qualitative spatial logic LBPT used to detect errors in matches, and briefly give the intuitions behind the logic. The logic itself is presented in (Du and Alechina 2014).

The language of $LBPT$ contains a set of individual names (for individual geometries), binary predicates $NEAR$, FAR and BPT, and logical connectives, $\neg$ (not), $\wedge$ (and) and $\rightarrow$ (implies). As introduced before, BPT and BEQ stand for 'buffered partOf' and 'buffered equal' respectively. $BEQ(a, b)$ is defined as $BPT(a, b) \wedge BPT(b, a)$. The relations $NEAR$ and FAR are also defined using a level of tolerance σ. Two geometries a and b are $NEAR$ if there exist a point p_a in a and a point p_b in b such that the distance between p_a and p_b is at most 2σ. The intuition is that if a and b are shifted towards each other by σ they will definitely be touching. Two geometries a and b are FAR if for any two points p_a in a and p_b in b, their distance is strictly greater than 4σ. $NEAR$ and FAR are disjoint but not mutually exhaustive relations. For two points which are at the distance of 3σ, they are neither $NEAR$ nor FAR.

A sound and complete axiomatisation of LBPT consists of the following axioms:

A0 All tautologies of classical propositional logic

A1 $BPT(a, a)$;

A2 $NEAR(a, b) \rightarrow NEAR(b, a)$;

A3 $FAR(a, b) \rightarrow FAR(b, a)$;

A4 $BPT(a, b) \wedge BPT(b, c) \rightarrow NEAR(c, a)$;

A5 $BPT(b, a) \wedge BPT(b, c) \rightarrow NEAR(c, a)$;

A6 $BPT(b, a) \wedge NEAR(b, c) \wedge BPT(c, d) \rightarrow \neg FAR(d, a)$;

A7 $NEAR(a, b) \wedge BPT(b, c) \wedge BPT(c, d) \rightarrow \neg FAR(d, a)$;

MP Modus ponens: $\phi,\ \phi \rightarrow \psi\ \vdash\ \psi$.

Initially, LBPT reasoning was implemented using Pellet. LBPT reasoning rules were encoded as constraints on object properties. However, not all LBPT reasoning rules are expressible in this way, and we also encountered some bugs (for example, non-minimal explanations) in the explanations generated using reasoning with role chain inclusions. We therefore implemented a dedicated LBPT reasoner integrated with an assumption-based truth maintenance system (ATMS) (de Kleer 1986). The LBPT reasoner derives consequences by applying inference rules to previously derived sentences, and the ATMS maintains dependencies between derived consequences and a set of distinguished data known as *assumptions*. In particular, it maintains all minimal sets of assumptions responsible for the derivation of $\perp$ (false), referred to as *nogoods* in ATMS terminology.

The reasoner reasons using inference rules which encode the LBPT axioms. It repeatedly checks whether any of its inference rules are applicable to the assumptions and/or any previously derived sentences, and if so, it sends to the ATMS the consequence of applying the rule together with a *justification* consisting of the name of the rule and the antecedents of the rule used to derive the consequence. The newly derived consequence can now be used as an input to the reasoner and the cycle repeats until no new consequences can be derived.

The ATMS computes dependency (or derivability) of data from sets of assumptions. In our case statements of matches between objects are treated as assumptions and NEAR and FAR statements as premises (non-retractable statements). To compute derivability, the ATMS builds and maintains a dependency or justification graph. A *node* in the graph represents a unique datum (i.e., a formula). Inconsistency is represented by a distinguished node $n_\perp$. *Justifications* form the edges of the graph and record the fact that a datum node (the consequent) can be derived from a set of other datum nodes (the antecedents). A node may be the consequent of more than one justification (recording the different ways in which it can be derived), and be an antecedent in other justifications (recording the inferences that can be made using it). Each node has a *label* that records the minimal sets of assumptions from which it can be derived.

Whenever a new justification is received from the problem-solver, the labels of nodes in the graph are updated to maintain four key properties:

- **soundness** if a set of assumptions is in the label of a formula, then the reasoner has found a derivation of this formula which only uses those assumptions;
- **completeness** all ways of deriving the formula discovered by the reasoner so far are included in the label;
- **minimality** the formula (so far) has not been discovered to be derivable from a strict subset of any set of assumptions in the label;
- **consistency** if a set of assumptions is discovered to be inconsistent, then it is removed from the labels of all nodes (except $n_\perp$).

The basic task of the ATMS is to compute sound, complete, minimal and consistent labels for datum nodes which correctly reflect the sequence of justifications generated by the reasoner. From the label of a node we can therefore determine the minimal sets of assumptions needed to (consistently) derive the corresponding datum discovered by the reasoner. From the label of the false node $n_\perp$ we can discover the reasons for any inconsistencies in the ontology (i.e., the ATMS nogoods).

The LBPT reasoner with an ATMS is implemented in Pop-11 (http://www.cs.bham.ac.uk/research/projects/poplog/freepoplog.html). The LBPT inference rules are implemented using Poprulebase, a Pop-11 forward chaining rule interpreter.

Experiments and Evaluation

In this section, we report the use of MatchMaps to match OSM data (building layer) (OpenStreetMap 2014) to OSGB MasterMap data (Address Layer and Topology Layer) (Ordnance Survey 2012). The studied areas are in city centres of Nottingham UK and Southampton UK. The Nottingham data was obtained in 2012, and the Southampton data in 2013. The numbers of spatial objects in the case studies are shown in Table 1. The number of OSM objects is smaller in each case, because OSM data often describes a collection of OSGB objects as a whole, for example, OSGB shops as a shopping centre in OSM.

Table 1: Data used for evaluation

	OSM spatial objects	OSGB spatial objects
Nottingham	281	13204
Southampton	2130	7678

We chose these two datasets for evaluation because they have a reasonable representation in OSM (city centres usually attract more attention from OSM contributors, and a variety of buildings and places are represented there) and are of reasonable size. In both cases, we set the value of σ used in geometry matching to be 20 m. The experiments were

Table 2: Matching OSM spatial objects to OSGB

	TP	FP	TN	FN	Precision	Recall
Nottingham	177	19	64	21	0.90	0.84
Southampton	1997	21	71	41	0.98	0.97

performed on an Intel(R) Core(TM)2 Duo CPU E8400 @ 3.00 GHz, 4.00 GB RAM desktop computer. Times are in seconds, averaged over 5 runs.

The main objective of evaluation was to establish the precision and recall of MatchMaps. Given the size of the case studies, it was infeasible for domain experts to produce a complete set of ground truth matches manually. Instead, we computed the ground truth as follows. For each OSM object a, we check all matches which involve a (either a single $sameAs(a, b)$ match with some b in the OSGB dataset, or several $partOf$ matches involving a) produced by MatchMaps. If the match or matches were determined by a human expert to be correct, a was classified as 'Correctly Matched' (True Positive or TP), otherwise it was classified as 'Incorrectly Matched' (False Positive or FP). For $a \in FP$, a check was made whether a correct match for a existed; if yes, a was labelled FP_{sbm}. If a was not involved in any matches, a check was made whether a correct match for it existed. If there was no correct match, then a was placed in 'Correctly Not-matched' (True Negative or TN), otherwise in 'Incorrectly Not-matched' (False Negative or FN). Straightforward matches were checked by a non-expert using guidelines developed in conjunction with a subject matter expert from the Nottingham Geospatial Institute. A subject matter expert at Ordnance Survey (Great Britain's National Mapping Authority) classified non-straightforward cases (approximately 10% of the total output of the system for the given datasets). Note that the size of each group is the number of OSM spatial objects in it. For example, for the Victoria Centre in OSM, though there are hundreds of 'partOf' matches involving it, it is only counted as one element in 'Correctly Matched'. Precision was computed as the ratio of $|TP|$ to $|TP| + |FP|$, and recall as the ratio of $|TP|$ to $|TP| + |FN| + |FP_{sbm}|$. As shown in Table 2, for both Nottingham and Southampton cases, precision is $\geq 90\%$ and recall $\geq 84\%$.

We also performed experiments to evaluate the performance of the qualitative spatial reasoning step (step 5). In that step, we are primarily interested in consistency of matches generated in step 4 for the many-to-many case, that is, $sameAs$ and $partOf$ matches between two sets of objects $\{a_1, \dots, a_n\}$ and $\{b_1, \dots, b_k\}$ which belong to two buffered equal aggregated geometries. To check consistency of those matches with respect to LBPT, we translate matches to LBPT. For every $sameAs(a_i, b_j)$, we generate an assumption $BEQ(g(a_i), g(b_j))$ and for every $partOf(a_i, b_j)$, we generate an assumption $BPT(g(a_i), g(b_j))$, where $g(a_i)$ and $g(b_j)$ are geometries of a_i and b_j, respectively. We also generate $NEAR$ and FAR facts between all objects in $\{a_1, \dots, a_n\}$ and between all objects in $\{b_1, \dots, b_k\}$, from distances between geometries within the same dataset, using $\sigma = 20m$. If a BEQ or BPT assumption is involved in a derivation of inconsistency and is retracted, then the corresponding $sameAs$ or $partOf$ matches are also retracted.

Table 3: LBPT reasoning

	nogoods	retracted BEQ/BPT	interactions
Nottingham	172	31	3
Southampton	268	114	7

Table 3 shows the number of nogoods generated by the LBPT reasoner with an ATMS. The number of interactions is the number of times users are asked to take actions or use strategies to resolve problems (a strategy is a heuristic which allows users to retract all 'similar' statements at a time, for example, all statements of the form $partOf(x, o)$ for some small object o). As a result of LBPT reasoning and removal of BEQ and BPT assumptions, we also withdraw 1325 $sameAs/partOf$ assumptions for Nottingham and 488 $sameAs/partOf$ assumptions for Southampton.

Using LBPT reasoning to remove these assumptions makes a difference to the running time and to the number of human interactions required to restore consistency in step 6. If we skip the LBPT reasoning step, and check the matching with respect to UNA and NPH assumptions directly, then it would require much more time to calculate explanations for inconsistencies (see Table 4).

Table 4: Time for calculating explanations in UNA and NPH step by Pellet

	with LBPT step	without LBPT step
Nottingham	16s	812s
Southampton	3s	46s

More importantly, more explanations for inconsistency would be generated by Pellet, since approximately 1800 wrong matches are removed in the LBPT step. The exact number of additional human interactions is not known since an attempt to resolve all inconsistencies was abandoned after 4 hours, but it is significantly greater than the number of interactions required with the LBPT step (see Table 5).

Table 5: UNA and NPH reasoning by Pellet

	retracted $sameAs/partOf$	interactions
Nottingham	151	11
Southampton	199	12

Related Work

Other tools for ontology matching or data interlinking exist, for example LogMap (Jiménez-Ruiz and Grau 2011) and KnoFuss (Nikolov, Uren, and Motta 2007). In (Du et al. 2013a), their performance on geospatial datasets is compared to that of the preliminary version of MatchMaps (Du et al. 2013b). The precision and recall of MatchMaps (on smaller datasets with manually computed ground truth) are much higher than that of LogMap and KnoFuss, mainly because they do not make explicit use of spatial information. Another feature of MatchMaps which explains its higher precision, is that when an inconsistent set of matching statements and axioms is found, MatchMaps does not automatically withdraw one of the statements but instead asks a human expert to make a decision. This approach was adopted because it was discovered empirically that none of the commonly-used heuristics for choosing the statement to withdraw works well in all cases.

In geospatial information science, several methods have been developed for matching geometries or integrating geospatial data. Kundu (2006) proposed a method for conflating two polygonal lines, based on a distance measure. Fu and Wu (2008) proposed a method for matching entities in vector spatial data, based on interior intersection for polygons and buffer division for lines. Tong et al (2009) proposed a spatial feature matching method in map conflation, based on the calculation of weighted average of positional, shape, directional, and topological measures. However, there is no consensus approach to matching geometries, and the computational cost of most methods is high. Crowd-sourced geospatial information introduces additional challenges, since the geometries are typically less accurate and may contain errors.

For the application described in this paper, standard spatial formalisms (see, for example, (Aiello, Pratt-Hartmann, and Benthem 2007)) are not easily applicable since they presuppose accurate geometries or regions with sharp boundaries. Formalisms which consider indeterminate regions, such as (Roy and Stell 2001), presuppose the existence of a core part of a region and a more vague part, while in our application, the real location of the object can be anywhere within the buffer with the same degree of (un)certainty.

Conclusion

Crowd-sourced geospatial datasets contain a wealth of information. However this information is not entirely reliable and needs to be validated. One way of validating it is to match the objects described in a crowd-sourced dataset to an authoritative geospatial data source. We describe an application, MatchMaps, that generates such matches and checks them for consistency with respect to ontological classification of the objects and a qualitative spatial logic. We do this by extending existing artificial intelligence techniques for matching ontologies to the more specialised case of a geospatial ontology (containing location and geometry information), and using a novel qualitative spatial logic to validate the matches. An evaluation of MatchMaps shows a high degree of precision and recall for two case studies. One direction of future research is to conduct a user evaluation study with OSGB to determine the amount of human effort required to perform a matching task.

References

Aiello, M.; Pratt-Hartmann, I. E.; and Benthem, J. F. v. 2007. *Handbook of Spatial Logics*. Springer.

de Kleer, J. 1986. An assumption-based TMS. *Artificial Intelligence* 28(2):127–162.

Du, H., and Alechina, N. 2014. A Logic of Part and Whole for Buffered Geometries. In *the 7th European Starting AI Researcher Symposium (STAIRS)*, 91–100.

Du, H.; Alechina, N.; Jackson, M.; and Hart, G. 2013a. Matching Geospatial Instances. In *Proceedings of the 8th International Workshop on Ontology Matching*, volume 1111 of *CEUR Workshop Proceedings*, 239–240. CEUR-WS.org.

Du, H.; Alechina, N.; Jackson, M.; and Hart, G. 2013b. Matching Formal and Informal Geospatial Ontologies. In *Geographic Information Science at the Heart of Europe*, Lecture Notes in Geoinformation and Cartography. Springer. 155–171.

Fu, Z., and Wu, J. 2008. Entity matching in vector spatial data. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* 37(B4):1467 – 72.

Haklay, M. 2010. How good is volunteered geographical information? A comparative study of OpenStreetMap and Ordnance Survey datasets. *Environment and Planning B: Planning and Design* 37(4):682–703.

ISO Technical Committe 211. 2003. ISO 19107:2003 Geographic information – Spatial schema. Technical report, International Organization for Standardization (TC 211).

Jackson, M. J.; Rahemtulla, H.; and Morley, J. 2010. The Synergistic Use of Authenticated and Crowd-Sourced Data for Emergency Response. In *2nd International Workshop on Validation of Geo-Information Products for Crisis Management (VALgEO)*, 91–99.

Jiménez-Ruiz, E., and Grau, B. C. 2011. LogMap: Logic-Based and Scalable Ontology Matching. In *Proceedings of the10th International Semantic Web Conference*, volume 7031 of *LNCS*, 273–288. Springer.

JPP. 2014. openjump. http://www.openjump.org.

Kundu, S. 2006. Conflating two polygonal lines. *Pattern Recognition* 39(3):363–372.

Nikolov, A.; Uren, V.; and Motta, E. 2007. KnoFuss: a Comprehensive Architecture for Knowledge Fusion. In *Proceedings of the 4th International Conference on Knowledge Capture*, 185–186.

OpenStreetMap. 2014. The Free Wiki World Map. http://www.openstreetmap.org.

Ordnance Survey. 2012. Ordnance Survey. http://www.ordnancesurvey.co.uk.

Roy, A., and Stell, J. 2001. Spatial Relations between Indeterminate Regions. *International Journal of Approximate Reasoning* 27(3):205 – 234.

Sirin, E.; Parsia, B.; Grau, B. C.; Kalyanpur, A.; and Katz, Y. 2007. Pellet: a Practical OWL-DL Reasoner. *Web Semantics: Science, Services and Agents on the World Wide Web* 5:51–53.

Tong, X.; Shi, W.; and Deng, S. 2009. A probability-based multi-measure feature matching method in map conflation. *International Journal of Remote Sensing* 30(20):5453–5472.

Vivid Solutions, Inc. 2014. JTS Topology Suite. http://www.vividsolutions.com/jts.