Knowledge Graphs: Semantics, Machine Learning, and Languages
M. Acosta et al. (Eds.)

doi:10.3233/SSW230013

COBALT: A Content-Based Similarity Approach for Link Discovery over Geospatial Knowledge Graphs

Alexander BECKER [a], Abdullah AHMED [a] and Mohamed Ahmed SHERIF [a]
Axel-Cyrille NGONGA NGOMO [a]

[a] *DICE group, Department of Computer Science, Paderborn University*
ORCiD ID: Alexander Becker https://orcid.org/0009-0003-8212-4647, Abdullah Ahmed https://orcid.org/0000-0002-0668-0735, Mohamed Ahmed Sherif https://orcid.org/0000-0002-9927-2203, Axel-Cyrille Ngonga Ngomo https://orcid.org/0000-0001-7112-3516

Abstract. *Purpose:* Data integration and applications across knowledge graphs (KGs) rely heavily on the discovery of links between resources within these KGs. Geospatial link discovery algorithms have to deal with millions of point sets containing billions of points. *Methodology:* To speed up the discovery of geospatial links, we propose COBALT. COBALT combines the content measures with R-tree indexing. The content measures are based on the area, diagonal and distance of the minimum bounding boxes of the polygons which speeds up the process but is not perfectly accurate. We thus propose two polygon splitting approaches for improving the accuracy of COBALT. *Findings:* Our experiments on real-world datasets show that COBALT is able to speed up the topological relation discovery over geospatial KGs by up to 1.47×10^4 times over state-of-the-art linking algorithms while maintaining an F-Measure between 0.7 and 0.9 depending on the relation. Furthermore, we were able to achieve an F-Measure of up to 0.99 by applying our polygon splitting approaches before applying the content measures. *Value:* The process of discovering links between geospatial resources can be significantly faster by sacrificing the optimality of the results. This is especially important for real-time data-driven applications such as emergency response, location-based services and traffic management. In future work, additional measures, like the location of polygons or the name of the entity represented by the polygon, could be integrated to further improve the accuracy of the results.

Keywords. Knowledge graphs, Data Integration, Linked Data, Geospatial Knowledge graphs, Content Measure Similarity, Topological Relations

1. Introduction

The necessity for highly scalable methods for finding links between geospatial resources has arisen as a result of the rapid proliferation of linked geospatial data. Only 7.1% of the relationships between resources relate geographical elements, as was noted in earlier publications [1]. There are two basic causes for this: I) The vast quantity of geospatially represented resources on Linked Open Data (LOD) necessitates scalable techniques for

computing linkages between geospatial resources. For example LINKEDGEODATA [2] has over 20 billion triples describing millions of geographical things. II) The computation of certain relations, such as distance and topological links between geospatial resources, have to deal with the vector representation of geospatial data. For instance, for identifying the nearby points of interest within a certain radius.

Discovering links among KGs in RDF is crucial for many data-driven applications, according to the Linked Data principles [3]. Nowadays dealing with geospatial resources is a fundamental in many real-time applications [4], such as emergency response, location-based services and real-time traffic management. However, generating links among such real-time geospatial KGs is challenging task in order to enable real-time decision making. Thus, both the efficiency and scalability of the link discovery process becomes more challenging.

Recently, algorithms such as RADON [5], RADON2 [6], GIA.NT [7], and DORIC [8] have been developed. These algorithms compute topological relations between geographical resources quickly and effectively. In all of them, the *Dimensionally Extended Nine-Intersection Model* (DE-9IM) [9] is used. The DE-9IM defines topological relations between two-dimensional geometries by calculating the dimensions of the intersections between the interior, boundary and exterior of two geometries. The relations defined by the DE-9IM are the ones commonly used in natural language [10]: `Equals`, `Disjoint`, `Intersects`, `Touches`, `Crosses`, `Within`, `Contains`, `Overlaps`, `Covers` and `Covered By`. In an attempt to speedup the computation of geospatial relations, Ahmed et al. [11] have studied the effect of simplifying the resources' geometries on the runtime and F-Measure of link discovery approaches. However, computing the DE-9IM is still very expensive in terms of runtime.

In this paper, we propose COBALT, an approach based on the content measures combined with R-tree indexing to discover the topological relations defined in [9] and [10]. To the best of our knowledge, this is the first work that uses content-based measures integrated with R-tree indexing for discovering links among RDF geospatial resources. We summarize our contribution as follows:

1. We present and formalize the problem of topological relation discovery for geospatial resources based on content measures and R-tree indexing.
2. We study the effect of using different R-tree building algorithms, node capacities and the impact of indexing both datasets.
3. We study the impact of using the content-based measures for topological relations discovery on both runtime and accuracy.
4. In order to increase the accuracy of our approach, we propose two polygon splitting strategies and analyze the effect of them on both runtime and accuracy.

The rest of the paper is structured as follows. We begin by introducing the link discovery problem over RDF KG in Section 2, where we also formally define the topological relations based on content measures. Then, we describe our approach in Section 3. In Section 4, we present our evaluation and results. We then discuss the state-of-the-art related work in Section 5. Finally, we conclude our paper and present some future work in Section 6. Our implementation of COBALT is open source and implemented into the LIMES framework.[1]

[1] https://github.com/dice-group/LIMES

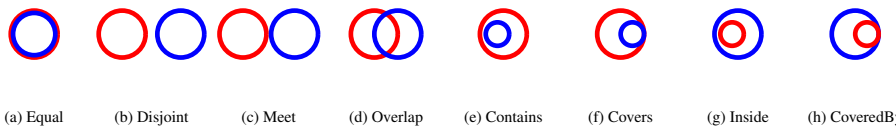


Figure 1. The content measure relations.

2. Preliminaries

Knowledge Graph. A Knowledge Graph (KG) G is a set of triples $(s,p,o) \in (\mathscr{R} \cup \mathscr{B}) \times \mathscr{P} \times (\mathscr{R} \cup \mathscr{L} \cup \mathscr{B})$, where $\mathscr{R}$ is the set of all resources, $\mathscr{B}$ is the set of all blank nodes, $\mathscr{P}$ is the set of all predicates, and $\mathscr{L}$ the set of all literals.

Link Discovery. Given a source knowledge graph G_s and a target knowledge graph G_t (for example, two KGs containing the geometric representation of national borders) and a relation r (e.g., `:touches`), the goal of the link discovery problem is to find all pairs $(s,t) \in G_s \times G_t$ such that $r(s,t)$ holds. The result is produced as a set of links called a *mapping*: $M = \{(s,r,t) | s \in G_s, t \in G_t\}$.

Content Measures for Topological Relations. We use the content measures as defined Godoy et al. [12] for deciding if the relation $r(s,t)$ exists. Godoy et al. have implemented three content measures to determine whether a topological relation between two polygons exists by comparing the area, the diagonal or the area and the diagonal of the polygon's minimum bounding boxes. The relations distinguished by the content measures are shown in Figure 1. *The Minimum Bounding Box (MBB)* of a polygon P with $n \geq 3$ points $((x_1,y_1),\ldots,(x_n,y_n))$ is defined as the smallest rectangle which fully contains the polygon's points. Formally, $\text{MBB}(P) = \big((X_{min},Y_{max}),(X_{max},Y_{min})\big)$, where $X_{min} = \min(x_1,\ldots,x_n)$, $X_{max} = \max(x_1,\ldots,x_n)$, $Y_{min} = \min(y_1,\ldots,y_n)$ and $Y_{max} = \max(y_1,\ldots,y_n)$. The *area* of a MBB M is defined as $area(M) = (X_{max} - X_{min}) \cdot (Y_{max} - Y_{min})$. The *diagonal* of a MBB M is formally defined as $diagonal(M) = \sqrt{(X_{max} - X_{min})^2 + (Y_{max} - Y_{min})^2}$. We further use the same definitions of *intersection*, *union* and *distance* among MBBs from [12].

3. Approach

We start our approach by indexing the source dataset polygons using R-tree then we apply content measure on the indexed polygons.

R-tree Indexing. R-trees [13] are an enhanced variant of binary trees, where an R-tree stores the MBBs of the polygons instead of the polygons themselves. In COBALT, we use *Guttman*'s R-tree [14] to index the source dataset in order to filter out as many disconnected polygon pairs as possible to reduce the runtime of the linking process. Every node's MBB contains all its children's MBBs, so in case an MBB of a parent node does not intersect a query rectangle (a query rectangle is a MBB from target data), none of its descendants can [14]. The bottom layer of an R-tree stores the MBBs of source dataset polygons, and all layers above it match the criterion applied to indexing the bottom layer. One example of a handcrafted R-tree is depicted in Figure 2.

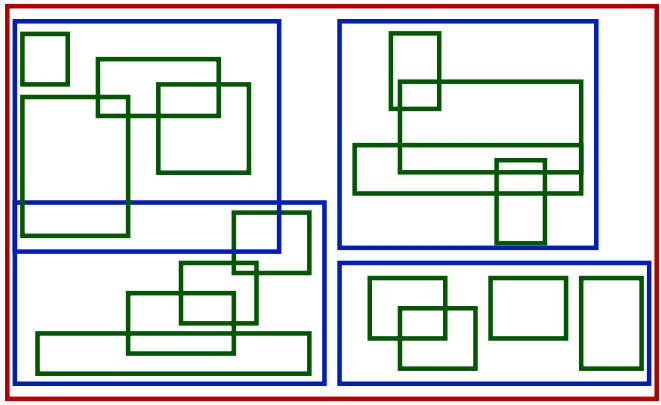


Figure 2. A handcrafted R-tree, green is the bottom layer, blue is the middle, and red is the top layer.

Querying R-tree. R-trees are easy to query recursively. Let q be the target MBB of the polygon for which we want to find the intersected MBBs of the source polygons. Since the nodes of an R-tree nodes are R-trees, we utilize the same algorithm for each node. If the current node of the R-tree is at the bottom layer, (i.e., it contains no other R-trees but polygons), we then verify if q intersects each of the MBBs of the source polygons saved in this node and add such source polygons to the query result. In case the current node of the R-tree is not on the bottom layer, we check if each child node's MBB has at least one common point with q and if that is the case, we recursively repeat the method for that node. In Figure 2 for instance, the MBBs of the left two blue nodes overlap. In case the query rectangle q lies in the area where two nodes' MBBs overlap, we have to check the children of both nodes. Therefore, we need a fast building approach that minimizes overlapping parent nodes.

Building R-tree. There are two main ways for constructing R-trees: (i) *Static building algorithms* work by getting all data as the input and then constructing the tree with all data at once. (ii) *Dynamic building algorithms* work by inserting data one by one into the tree. As our datasets are not changing frequently, we focus on static algorithms as dynamic algorithms will require more run time for reinserting data to keep the R-tree balanced. Because we only query the R-tree once for every target geometry, the build quality (overlap) is not as important for an overall fast execution. To test the impact of different building algorithms, we use four static R-tree building algorithms (i.e., *SmallestX*, *STR*, *OTM* and *PackedHilbertR-tree*) and one dynamic algorithm (i.e., R*-Tree). The first static algorithm is *SmallestX* [15], which sorts the MBBs by the smallest x coordinate. The *SortTileRecursive* [16] algorithm (STR) builds the R-tree bottom-up and divides the MBBs into slices sorted by the x coordinate and then sorts them by the y coordinate, then recursively combines the parent nodes of the bottom layer. The *OTM* algorithm [17] works similar to STR, but recursively sorts the MBBs by alternating x and y coordinate with a top-down bulk loading approach. The last static building algorithm we use is the *PackedHilbertR-tree* [18] algorithm, which sorts the MBBs by their position on the Hilbert curve. On the other hand, we use the dynamic R-tree building algorithm *R*-Tree* [19], which supports inserting of new elements after creation and tries to minimize the area occupied by nodes. In our experiments, we insert the polygons one

Relation	Disjoint	Meets	Overlap	Equals	Covers	CoveredBy	Contains	Inside
$F_a(A,B)$	(0,1)	(0,1)	(0,1)	1	1	(0,1)	1	(0,1)
$F_a(B,A)$	(0,1)	(0,1)	(0,1)	1	(0,1)	(0,1)	(0,1)	1
$F_a(A,B)+F_a(B,A)$	(0,1)	(0,1]	(0,2)	2	(1,2)	(1,2)	(1,2)	(1,2)

Table 1. Area based content measure relations based on values of F_a. [12]

by one into the R*-Tree and use the values $\{4,8,16,32,64,128,256\}$ for the capacities of each node.

Content Measures For Topological Relations. Given two MBBs A and B, the *area-based content measure* (F_a) is the first of the three content measures from [12]. F_a is the normalization of the area of each MBBs of both A and B by the area of the MBB of the union of A and B. Formally,

$$F_a(A,B) = \frac{\text{area(A)}}{\text{area(MBB(A} \cup \text{B))}}, \tag{1}$$

where $F_a(B,A)$ is defined analogously. The range of $F_a \in (0,1]$. In Table 1, we present the values of $F_a(A,B)$, $F_a(B,A)$ and $F_a(A,B)+F_a(B,A)$ for the different topological relations. F_a cannot distinguish the following pairs of relations: `(Meet, disjoint)`, `(covers, contains)` and `(covered by, inside)`. For instance, the union MBB will be the same as the MBB of the MBBs for `contains` and `covers`. However, this measure can accurately detect the `equal` relation because both the input MBBs have the same area as their union's MBB. The second content measure is the *diagonal-based content measure* (F_d) [12], formally defined as:

$$F_d(A,B) = \frac{\text{diagonal(A)}}{\text{diagonal(MBB(A} \cup \text{B))}}. \tag{2}$$

The range of $F_d \in (0,1]$ and it cannot distinguish `(covers, contains)` and `(covered by, inside)` for the same reason as in the case of F_a. The third content measure is the *mixed content measure* (F_m) [12]. F_m utilizes the area, diagonal and distance of the MBBs for finding the topological relations. Unlike the other two content measures, it is able to distinguish between `(contains, covers)` and `(inside, covered by)`. Formally,

$$F_m(A,B) = \frac{\text{area(A)} - 2 \cdot \text{area(MBB(A} \cap \text{B))}}{\text{area(A)}} + \frac{\text{distance(A, B)}}{\text{diagonal(A)}}. \tag{3}$$

Combining R-tree Indexing and Content Measure. Our R-tree indexing filters out `disjoint` polygon pairs based on their MBB. We only keep the indexed source dataset in memory, which reduces the space complexity as we then stream-process the target dataset. In the case of the `disjoint` relation, we first add all pairs of geometries that the indexing would filter out to the result set then we check the other relations on the rest of the geometries pairs. In Algorithm 1, we line out the steps for the area based content measure F_a. For the other measures, we replace F_a with F_d for the diagonal measure and F_m for the mixed measure (Lines 11-12). Additionally, the values of F_a need to be checked against the other measures' values from [12] (Line 14).

Algorithm 1 DiscoverLinksAreaBased(G_s, G_t, r)

Input: Source KG G_s, Target KG G_t, Topological relation r
Output: Mapping : $M = \{(s,r,t)|s \in G_s, t \in G_t\}$
$tree \leftarrow buildRtree(G_s)$
Initialise $M \leftarrow \{\}$
for each MBB(t) $t \in G_t$ **do**
    $I \leftarrow$ queryRtree(tree, t))
    if r is disjoint **then**
        Add all pairs$(s,r,t) \forall s \in (G_s \setminus I)$ to M
    end if
    for each MBB(s) $s \in I$ **do**
        $X \leftarrow \mathrm{F}_a(MBB(s), MBB(t))$
        $Y \leftarrow \mathrm{F}_a(MBB(t), MBB(s))$
        $Z \leftarrow X + Y$
        if X, Y, Z match the respective values of the relation r in in Table 1 **then**
            Add (s,r,t) to M
        end if
    end for
end for
return M

Algorithm 2 MatchTrees(sourceTree, targetTree)

Result $\leftarrow \{\}$
if area(sourceTree) < area(targetTree) **then**
    swap sourceTree and targetTree
end if
for each child of sourceTree **do**
    if child is leaf **then**
        Result = Result $\cup$ queryRtree(targetTree, MBB(child))
    else
        Result = Result $\cup$ MatchTrees(child, targetTree)
    end if
end for
return Result

Indexing both Datasets. In many cases swapping the source and target datasets results in different runtimes. In order to reduce the impact of dataset ordering on the runtime, we study the possibility of indexing both datasets instead of one. Instead of querying the R-tree for each target geometry, we use Algorithm 2 to match two R-trees to each other and recursively find all pairs that intersect. This approach removes the need to choose which dataset to index but comes with the price of increasing the memory footprint of our approach as we have to keep both datasets in memory.

Splitting Polygons to Gain Accuracy. We are able to improve the F-Measure of COBALT for some relations by splitting the geometries into multiple pieces before using the content measure functions to determine the relation. In particular, we split polygons recursively t times into four pieces using two different strategies: **1) Equal split**, where

(a) Splitting the polygon into equal size parts two times

(b) Splitting the polygon into fitting parts two times

Figure 3. Different options to split polygons. The green triangle is the original polygon, the dark red lines are the splitting lines for the first split iteration, the light red lines are the splitting lines for the second split iteration. The blue rectangles indicate the MBB used for determining the second iteration splitting lines of the top left corner.

we are splitting the original polygon into equally sized parts. The resulting polygon parts are the intersection of a grid pattern over the original polygon and the original polygon itself. In some cases, this leads to some splits not achieving any additional information as their parts of the grid are empty. For instance in Figure 3a, any further splits of the top left corner cell (the blue highlighted cell) would not increase the accuracy of COBALT as the original polygon does not have any points within this cell. **2) Fitting split**, where we divide the polygon into equally sized parts but using the MBB of the current polygon part for further splitting. Splitting the top left corner cell of the same polygon of the previous example using this strategy will result in the splitting presented in Figure 3b, where further splitting of the blue highlighted cell results in more detailed splits that fit better to the shape of the polygon. After splitting the polygons we compute the MBBs for all parts. Now as we have multiple polygon parts, we change the way the relation of the original polygon is determined. Let t be the number of splits into four parts. Let $A_{(i,j)}$ be the split part of geometry A at column i and row j and $B_{(k,l)}$ be the split part of geometry B at column k and row l for $\{(i,j,k,l) \in \mathbb{N}^4 | 0 \leq i,j,k,l < 2^t\}$. The newly defined relations can be found in Table 2. In particular, every grid pattern $A_{(i,j)}$ must be equal to $B_{(k,l)}$ for the `equals` relation to hold. For the `intersects` relation, at least one $A_{(i,j)}$ has to intersect with at least one $B_{(k,l)}$. For the `within` relation, all $A_{(i,j)}$ have to be contained in the MBB of the union of all $B_{(k,l)}$ it intersects. For the `contains` relation, we swap A and B then compute `within` relation instead. For the `overlaps` relation, we three conditions must hold: 1) at least one $A_{(i,j)}$ is not within B, 2) at least one $B_{(k,l)}$ is not within A, and 3) at least one $A_{(i,j)}$ intersects with at least one $B_{(k,l)}$. For the *touches* relation, at least one $A_{(i,j)}$ must touch any $B_{(k,l)}$ and every $B_{(i,j)}$ is related to every $B_{(k,l)}$ by either `touches` or `disjoint` relation.

4. Evaluation & Results

Datasets. We use two real-world datasets for evaluating COBALT: 1) The *NUTS*[2] dataset from the *Eurostat* group describes the territory of countries in the European Union, (potential) candidate countries and countries belonging to the European Free

[2] https://ec.europa.eu/eurostat/de/web/gisco/geodata/reference-data/administrative-units-statistical-units/NUTS, accessed on 01.09.2022

`equals`	$\forall \{(i,j) \in \mathbb{N}^2 \mid 0 \leq i,j < 2^t\}: A_{(i,j)}$ `equals` $B_{(i,j)}$
`intersects`	$\exists \{(i,j,k,l) \in \mathbb{N}^4 \mid 0 \leq i,j,k,l < 2^t\}: A_{(i,j)}$ `intersects` $B_{(k,l)}$
`disjoint`	$\forall \{(i,j,k,l) \in \mathbb{N}^4 \mid 0 \leq i,j,k,l < 2^t\}: A_{(i,j)}$ `disjoint` $B_{(k,l)}$
`within`	$\forall \{(i,j) \in \mathbb{N}^2 \mid 0 \leq i,j < 2^t\}: A_{(i,j)}$ `within` MBB$(\{B_{(k,l)} \mid \forall \{(k,l) \in \mathbb{N}^2 \mid 0 \leq k,l < 2^t\}:$ $A_{(i,j)}$ `intersects` $B_{(k,l)}\})$
`contains`	swap A and B then compute `within`
`overlaps`	$(\exists \{(i,j,k,l) \in \mathbb{N}^4 \mid 0 \leq i,j,k,l < 2^t\}: A_{(i,j)}$ `equals` $B_{(k,l)}) \vee A_{(i,j)}$ `within` $B_{(k,l)}) \vee$ $A_{(i,j)}$ `contains` $B_{(k,l)}) \vee A_{(i,j)}$ `overlaps` $B_{(k,l)}) \wedge$ $(\exists \{(i,j) \in \mathbb{N}^2 \mid 0 \leq i,j < 2^t\}: A_{(i,j)} \neg$ `within` MBB $(\{B_{(k,l)} \mid \forall \{(k,l) \in \mathbb{N}^2 \mid 0 \leq k,l < 2^t\}:$ $A_{(i,j)}$ `intersects` $B_{(k,l)}\})) \wedge$ $(\exists \{(i,j) \in \mathbb{N}^2 \mid 0 \leq i,j < 2^t\}: B_{(i,j)} \neg$`within` MBB$(\{A_{(k,l)} \mid \forall \{(k,l) \in \mathbb{N}^2 \mid 0 \leq k,l < 2^t\}:$ $B_{(i,j)}$ `intersects` $A_{(k,l)}\}))$
`touches`	$(\exists \{(i,j,k,l) \in \mathbb{N}^4 \mid 0 \leq i,j,k,l < 2^t\}: A_{(i,j)}$ `touches` $B_{(k,l)}) \wedge$ $\neg(\exists \{(i,j,k,l) \in \mathbb{N}^4 \mid 0 \leq i,j,k,l < 2^t\}: A_{(i,j)}$ `equals` $B_{(k,l)}) \vee A_{(i,j)}$ `within` $B_{(k,l)}) \vee$ $A_{(i,j)}$ `contains` $B_{(k,l)}) \vee A_{(i,j)}$ `overlaps` $B_{(k,l)})$

Table 2. Topological relations based on multiple splits of polygons.

Trade Association. 2) The *Corine Land Cover (CLC)*[3] [20] created by the *European Environment Agency*. CLC contains information about the land use of the 39 EEA39 countries[4].

Hardware & Software. All experiments were conducted on the NOCTUA1[5] cluster of the *Paderborn university*. NOCTUA1 consists of 256 compute nodes, each having two Intel *Xeon Gold* "Skylake" 6148 processors, which comes to a total of 40 cores with 2.4 GHz and 192 GiB main memory. All used algorithms were implemented in *Java*, and the compute nodes ran on *OpenJDK* version 11.0.2. For an accurate runtime measurement, all experiments were started with all datasets already loaded into the main memory. Additionally, for linking each dataset pair, we ran the algorithms on the same compute node. All experiments were conducted with a memory limit of 30 GB. Unless otherwise stated, we use only one core for all the experiments.

Experiments Settings. We use COBALT$_{area}$, COBALT$_{diagonal}$ and COBALT$_{mixed}$ to dub the *area*, *diagonal* and the *mixed* measures of COBALT, respectively. We use the following four baselines: i) RADON [5], ii) RADON with only the MBBs of the original polygons (dubbed RADON$_{\text{MBB}}$), iii) GIA.NT [7] and iv) GIA.NT with only the MBBs of the original polygons (dubbed GIA.NT$_{\text{MBB}}$). For a fair runtime comparison, we use a version of GIA.NT that computes only one relation at a time. We also implemented a space-indexing-based version of COBALT, where we optimized the content-based measures based on the space tiling indexing of RADON [5]. We use COBALT$_{area(R)}$, COBALT$_{diagonal(R)}$ and COBALT$_{mixed(R)}$ to dub the *area*, *diagonal* and the *mixed* measures of the space-tiling-based indexing measures of COBALT, respectively. We also used the *Douglas-Peucker* polygon simplification algorithm [21]. The simplification is ap-

[3] https://land.copernicus.eu/pan-european/corine-land-cover, accessed on 01.09.2022
[4] https://land.copernicus.eu/portal_vocabularies/geotags/eea39
[5] https://pc2.uni-paderborn.de/hpc-services/available-systems/noctua1

plied to the dataset using simplification thresholds {0.05, 0.1, 0.2}, then the relations are computed using RADON. They were labeled as $\text{RADON}_{\text{simp}(0.05)}$, $\text{RADON}_{\text{simp}(0.1)}$ and $\text{RADON}_{\text{simp}(0.2)}$. Within all experiments, we computed the topological relations `{equals, intersects, contains, within, touches, overlaps}`.

Research questions. We aim to answer the following research questions:

- Q_1. What is the effect of indexing the input datasets on the runtime of COBALT?
- Q_2. How much efficiency (i.e., less runtime) we gain by using content measure for topological relation discovery?
- Q_3. How much accuracy we lose (i.e., less F-Measure) by using content measure for topological relation discovery?
- Q_4. In case we use a simplified version of the original polygons, will we have a better trade-off between accuracy and efficiency than using COBALT?
- Q_5. Will COBALT benefit from parallelization for big KGs such as CLC?
- Q_6. What is the trade-off between accuracy and efficiency when we integrate our polygon splitting strategies into COBALT?

Research question Q_1**.** The aim of our *first set of experiments* was to evaluate different R-tree indexing options for COBALT. To measure the difference in runtime between indexing only one dataset vs. indexing both the source and target datasets, we linked NUTS to CLC (see Table 3) and CLC to NUTS (see Table 4). First, we compared the algorithms that index both datasets to the algorithms that only index one dataset. Our results showed that when linking NUTS to CLC most of the algorithms that only index one dataset are faster than the matching algorithms that index both of them. On the CLC to NUTS experiment (CLC×NUTS), however, the matching algorithms that index both datasets are faster than the algorithms that only index one dataset. This shows that the choice of the source dataset makes a difference regarding the runtime and the smaller dataset should be indexed instead of the bigger one. Because of the higher memory need for indexing both datasets, we decided to index only one dataset in our further experiments. In addition, our results showed that computing the *Hilbert curve*, or inserting entries one by one with the *R*-Tree*, takes much more time than the other algorithms. OMT and STR have the best runtime of the algorithms as their computations are not expensive and produce high quality R-trees. We conclude that the choice of the R-tree building algorithm as well as the capacity of the R-tree are highly dependent on the datasets used for benchmarking. It is important to find a balance between a fast building algorithm and an algorithm that allows efficient queries. Sorting entries by both x and y coordinate like OMT and STR is a good way to achieve this. This answers our first research question Q_1.

For the following experiments we use the STR building algorithm with a capacity of 4, but to respect the downside of only indexing one dataset, we also use the longer taking dataset combination for runtime values.

To answer Q_2, Q_3 and Q_4, we conducted our *second set of experiments* where we evaluate the performance of COBALT vs. all the baselines in terms of runtime and F-Measure. In particular, we aim to find the topological relations within the NUTS dataset against itself (i.e., NUTS×NUTS) and CLC×NUTS using each of the aforementioned algorithms. For linking NUTS×NUTS, the total required runtimes to compute the topological relations are shown in Figure 4a. The F-Measure for each relation can be seen in Table 5. For linking CLC×NUTS, the total required runtimes to compute the same six relations are shown in Figure 4b. The F-Measure of each relation can be seen in Table 6.

Table 3. Runtime in milliseconds for linking NUTS to CLC using different R-tree building algorithms and capacities combined with the mixed content measure

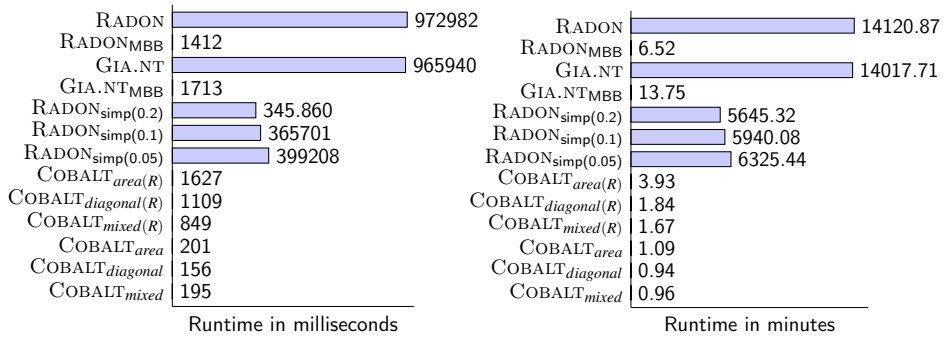
Algorithm	4	8	16	32	64	128	256
R*TREE	55745	49757	61810	77957	137733	202192	208227
HILBERT	258193	245062	233901	236357	363893	270579	273802
SMALLESTX	204226	95004	58875	64362	65360	62508	65948
OMT	35055	35778	36415	44063	71217	82937	103397
STR	36986	37088	38366	44042	48191	61899	77262
MATCHHILBERT	398502	345562	328975	320710	327772	357677	410224
MATCHSMALLESTX	85300	71626	62810	59780	46495	53344	66957
MATCHOMT	72359	56792	55213	50404	44134	48084	54525
MATCHSTR	45716	42130	41216	41769	41859	42278	43248

Table 4. Runtime in milliseconds for linking CLC to NUTS using different R-tree building algorithms and capacities combined with the mixed content measure

Algorithm	4	8	16	32	64	128	256
R*TREE	135037	139044	189439	372823	1067386	3527572	1067386
HILBERT	801618	866145	621211	793548	556130	540250	556130
SMALLESTX	263744	245692	422600	195218	93560	134091	324985
OMT	77928	62193	60354	55639	48598	54950	66138
STR	51448	47783	46708	47289	47359	48344	47359
MATCHHILBERT	401815	357914	363288	358353	344742	370740	417821
MATCHSMALLESTX	138362	106530	73954	70796	57164	68377	107256
MATCHOMT	72590	57643	56258	51175	45958	50478	56514
MATCHSTR	45245	42892	42077	42613	43233	43218	43849

Research question Q_2**.** From Figure 4a, we can see that all the content-based measures implemented in COBALT (i.e., COBALT_{area}, $\text{COBALT}_{diagonal}$ and COBALT_{mixed}) with R-tree indexing are 4 to 8 times faster than their counterparts (i.e., $\text{COBALT}_{area(R)}$, $\text{COBALT}_{diagonal(R)}$ and $\text{COBALT}_{mixed(R)}$) deployed based on the RADON's space tiling indexing. For instance, the total runtime of COBALT_{mixed} is 195 milliseconds while the total runtime in $\text{COBALT}_{mixed(R)}$ is 849 milliseconds, which means that COBALT_{mixed} is 4.3 times faster than $\text{COBALT}_{mixed(R)}$. The slowest content-based measure of COBALT (i.e., the COBALT_{area}) is *on average* 4840 times faster than RADON. COBALT_{mixed} is up to 1.47×10^4 times faster than RADON, which is the best speedup COBALT has in comparison to all other algorithms. This shows clearly how efficient are the the content-based measures when it comes to the runtime, which clearly answers our second research question Q_2.

Research question Q_3**.** Based on the results of Table 5, we analysed the impact of using the content-based measures on the F-Measure. For discovering the `equals` relation based on MBBs of the original polygons, all algorithms achieved an F-Measure of 0.996. For the `intersects`, `contains` and `within` relations, the F-Measures were 0.852, 0.853 and 0.853, respectively. The `overlaps` relation was the most affected relation by using the MBBs. In the case of the NUTS×NUTS experiment for instance, using MBBs



(a) Total runtime of linking NUTS×NUTS.

(b) Total runtime of linking CLC×NUTS.

Figure 4. Runtime results of KG linking experiments.

instead of the original polygons as the input for discovering the `overlaps` relation resulted in 586 true positives (out of 790 or 74.47%), 26884 false positives, 4012426 true negatives (out of 4039310 or 99.33%) and 204 false negatives. In total, using MBBs classified 45 times more polygon pairs falsely as being overlapped than the true number of overlapping pairs. The high number of pairs that was correctly identified as not overlapping is caused by the indexing algorithm, which filters out a high percentage of non-intersecting pairs. The only relation where the content-based measures produce better F-Measures than both RADON$_{\text{MBB}}$ and GIA.NT$_{\text{MBB}}$ was the `touches` relation. Both RADON$_{\text{MBB}}$ and GIA.NT$_{\text{MBB}}$ were not able to detect the `touches` relation correctly in most cases as the intersection matrix of the MBBs depends heavily on the polygon shape. For instance, using both RADON$_{\text{MBB}}$ and GIA.NT$_{\text{MBB}}$ for discovering the `touches` relation for the NUTS×NUTS Experiment (again see Table 5) resulted in an F-Measure of 0.001, while COBALT$_{area}$ and COBALT$_{diagonal}$ achieved an F-Measure of 0.678 and 0.779, respectively. Both the area and diagonal measures benefited from the fact that there are 20150 pairs that `touch` each other but only 790 pairs that `overlap`. To summarise, by using the content-based measures we lose *on average* 32% of the F-Measure compared to the F-Measure of 1.0 produced by RADON or GIA.NT. This answers our third research question Q_3.

Research question Q_4**.** State-of-the-art approaches tend to use polygons simplification in order to speed up the link discovery of topological relations [22]. As part of our second set of experiments, we studied the trade-off between accuracy and efficiency by using content-based measures on the polygons' MBBs vs. using a simplified version of the original polygons. Based on the results of Table 5, the F-Measures of RADON$_{\text{simp(0.05)}}$, RADON$_{\text{simp(0.1)}}$ and RADON$_{\text{simp(0.2)}}$ for the relations `contains` and `within` were worse than all the results produced using MBBs of polygons. For instance, RADON$_{\text{simp(0.05)}}$, RADON$_{\text{simp(0.1)}}$, and RADON$_{\text{simp(0.2)}}$ achieved the F-Measures 0.7, 0.72 and 0.733, respectively. While using COBALT on the MBBs of the original polygons achieved an F-Measure of 0.853 for the `contains` and `within` relations. When using the polygon simplification algorithms, the F-Measure for the `equals` relation is 1.0 for RADON with simplified polygons when linking NUTS×NUTS. The content measures are able to achieve an F-Measure of 0.996 for this relation. From the aforementioned results, we can conclude that using content-based measures on the polygons' MBBs result in a better trade-

Table 5. F-Measure for linking NUTS×NUTS (all values rounded to three decimal places). The results of COBALT combined with RADON indexing are omitted, because the indexing does not change the accuracy.

Algorithm	equals	intersects	contains	within	touches	overlaps
RADON	1.000	1.000	1.000	1.000	1.000	1.000
$\text{RADON}_{\text{MBB}}$	0.996	0.852	0.853	0.853	0.001	0.041
GIA.NT	1.000	1.000	1.000	1.000	1.000	1.000
$\text{GIA.NT}_{\text{MBB}}$	0.996	0.852	0.853	0.853	0.001	0.041
$\text{RADON}_{\text{simp(0.2)}}$	1.000	0.916	0.733	0.733	0.177	0.068
$\text{RADON}_{\text{simp(0.1)}}$	1.000	0.953	0.721	0.721	0.199	0.064
$\text{RADON}_{\text{simp(0.05)}}$	1.000	0.980	0.700	0.700	0.209	0.061
COBALT_{area}	0.996	0.852	0.853	0.853	0.678	0.041
$\text{COBALT}_{diagonal}$	0.996	0.852	0.853	0.853	0.779	0.041
COBALT_{mixed}	0.996	0.852	0.853	0.853	0.001	0.041

Table 6. F-Measure for linking CLC×NUTS. All values rounded to three decimal places and - indicate the total absence of the relation in the result set. The results of COBALT combined with RADON indexing are omitted, because the indexing does not change the accuracy.

Algorithm	equals	intersects	contains	within	touches	overlaps
RADON	-	1.000	1.000	-	-	1.000
$\text{RADON}_{\text{MBB}}$	-	0.709	0.689	-	-	0.066
GIA.NT	-	1.000	1.000	-	-	1.000
$\text{GIA.NT}_{\text{MBB}}$	-	0.709	0.689	-	-	0.066
$\text{RADON}_{\text{simp(0.2)}}$	-	0.938	0.931	-	-	0.332
$\text{RADON}_{\text{simp(0.1)}}$	-	0.963	0.958	-	-	0.419
$\text{RADON}_{\text{simp(0.05)}}$	-	0.980	0.975	-	-	0.540
COBALT_{area}	-	0.709	0.689	-	-	0.066
$\text{COBALT}_{diagonal}$	-	0.709	0.689	-	-	0.066
COBALT_{mixed}	-	0.709	0.689	-	-	0.066

off between efficiency and accuracy than using a simplified version of polygons. We can see the same behavior also for our second linking task, i.e., CLC×NUTS, see results in Table 6. This clearly answers our fourth research question Q_4.

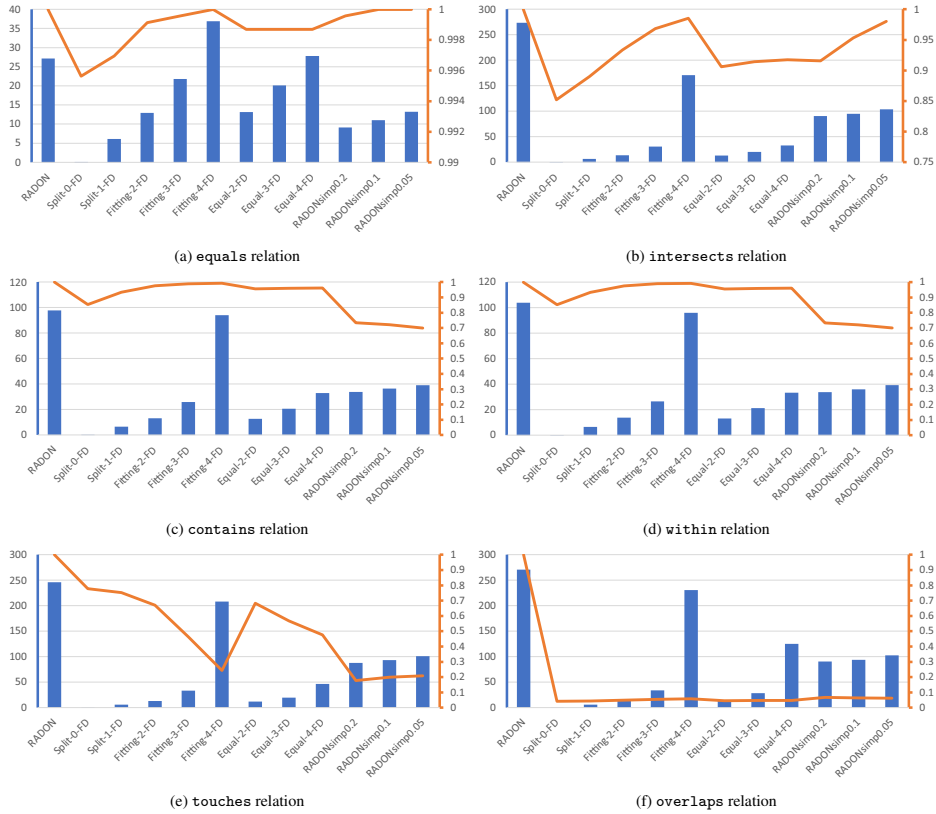
Research question Q_5 To answer Q_5, we conducted our *third set of experiments* by linking CLC against itself (i.e., CLC×CLC). For this experiment we implemented a parallelised version of COBALT, where we used $\{1,3,4,8\}$ thread(s). As shown in Table 7, all MBB based algorithms did not benefit from using multiple threads. Because the MBB-based algorithms are so fast, to the extent that the time needed for threads coordination is the same as the time saved by allocating the work to other threads. All of the MBB-based algorithms were able to finish linking CLC to itself in less than one hour. This is clearly answer our research question Q_5. On the other hand, multiple threads decreased the runtime of RADON and GIA.NT, because they use the intersection matrix that requires expensive computing which can take advantage of employing more threads. In particular, RADON is 6.31 times faster with 8 threads than with only one thread. All

Table 7. Runtime for linking CLC×CLC using a different number of threads. All runtimes are recorded in hours, where all values are rounded to three decimal places.

Algorithm	1 Thread	2 Threads	4 Threads	8 Threads
RADON	1179.489	686.007	340.827	186.913
$\text{RADON}_{\text{MBB}}$	0.703	0.729	0.597	0.586
GIA.NT	1179.463	675.589	334.928	179.415
$\text{GIA.NT}_{\text{MBB}}$	0.635	0.458	0.346	0.329
$\text{COBALT}_{area(R)}$	0.710	0.670	0.583	0.573
$\text{COBALT}_{diagonal(R)}$	0.539	0.564	0.513	0.505
$\text{COBALT}_{mixed(R)}$	0.494	0.550	0.509	0.503
COBALT_{area}	0.209	0.215	0.186	0.192
$\text{COBALT}_{diagonal}$	0.190	0.195	0.175	0.183
COBALT_{mixed}	**0.179**	**0.190**	**0.171**	**0.180**

of the content measures with R-tree indexing are at least three times faster than RADON with MBBs.

Research question Q_6**.** To study the trade-off between accuracy and efficiency when we apply our polygons splitting strategies (i.e., the equal split and the fitting split strategies) before applying the content measures, we conducted our *last set of experiments*. In particular, we are interested in comparing COBALT with the two splitting strategies to other approximation algorithms (i.e., polygon simplification). For this experiment we compute the topological relations for NUTS×NUTS. We benchmarked both split strategies defined in Section 3 against RADON and the combination of RADON and polygon simplification as we did in the previous experiments. The splitting algorithms are combined with the diagonal-based content measure. We dubbed our first splitting strategy (depicted in Figure 3a) as EQUAL-t-FD and the second splitting strategy as FITTING-t-FD (depicted in Figure 3b) with t being the number of recursive splits (We used 0 to 4 recursive splits) and FD being the diagonal-based content measure. As both split strategies produce the same result for 0 and 1 recursive splits, we only each of them once as SPLIT-0-FD and SPLIT-1-FD. For the `equals` relation (see Figure 5a), the diagonal content measure function achieved an F-Measure of 0.996 before applying the splitting algorithm on the polygons. The fitting split strategy with 3 and 4 recursive splits (i.e., FITTING-3-FD and FITTING-4-FD) achieved the best accurate results. On the other hand, the polygon simplification algorithms were all able to achieve perfect results in less time than both FITTING-3-FD and FITTING-4-FD. The diagonal content measure however achieved a high F-Measure of 0.996 while being over 100 times faster than the simplification algorithms. For the `intersects` relation (see Figure 5b), SPLIT-0-FD achieved an F-Measure of 0.852 without splitting polygons. The *fitting-split strategy* has a better accuracy for the `intersects` relation than the *equal-split strategy* for each t but with increased runtime. When compared FITTING-3-FD to EQUAL-4-FD we also notice that FITTING-3-FD is both faster and more accurate than EQUAL-4-FD. FITTING-3-FD is three times faster than the simplification algorithms and is only slightly worse in accuracy than the $\text{RADON}_{\text{simp}(0.05)}$ algorithm, but better than $\text{RADON}_{\text{simp}(0.1)}$ and $\text{RADON}_{\text{simp}(0.2)}$. For the `contains` and `within` relation (see Figure 5c and 5d), the content measures without splitting were able to achieve an F-Measure of 0.853, which was already better



(a) `equals` relation (b) `intersects` relation

(c) `contains` relation (d) `within` relation

(e) `touches` relation (f) `overlaps` relation

Figure 5. Runtime in seconds (blue) and F-Measure (orange) results for linking NUTS × NUTS.

than the simplification algorithms. By splitting the polygons, we were able to achieve a higher F-Measure.In particular, the FITTING-3-FD was 99% accurate while it used only 26.4% of the runtime RADON needs to compute the `contains` relation. This indicated that FITTING-3-FD was the strategy with the best runtime/accuracy trade-off. The `overlaps` relation (see Figure 5f) is a relation that cannot be accurately detected by the content measures or the polygon simplification algorithms. In this case splitting the polygons has a positive effect on the accuracy, but it is still too low to be usable with all F-Measures being smaller than 0.07. For the `touches` relation (see Figure 5e), the diagonal content measure was able to achieve a higher F-Measure (0.779) without splitting. This happens because the diagonal content measure focuses on recall rather than precision and by splitting the polygons there are more cases where parts from the two polygons are overlapping. Therefore, the splitting reduces accuracy and the normal diagonal content measure function should be used for the `touches` relation. Overall, splitting polygons is good way to improve the accuracy of COBALT for the spatial relations `intersects`, `contains`, and `within`. Our experiments show that the FITTING-T-FD achieves a higher F-Measure than the EQUAL-T-FD algorithms for each respective T but also have a higher runtime. By using our splitting technique, we could guarantee to finish a linking task in a predetermined amount of time while also fully utilizing the time to maximize the accuracy of the result. This answers our research question Q_6.

5. Related Work

In last years many algorithms have been proposed to address both the efficiency and accuracy of link discovery in general and link discovery over geospatial RDF KGs in particular. For instance, *Ngonga Ngomo* [1] computes the distance between geographical items using the Hausdorff distance. *Sherif et al.* [23] provide a review of 10 point-set distance metrics for link discovery. SILK [24] computes topological relations in accordance with the DE-9IM standard based on the MultiBlock. To compute topological relations between geographical resources quickly and accurately, RADON [5] offers an indexing technique coupled with space tiling approach. While RADON computes the intersection matrix for each relation between a pair of geometries repeatedly, RADON2 [6] caches the computed intersection matrix and reuses it whenever it is possible. In GIA.NT [7], *Papadakis et al.* have adapted RADON's indexing. In particular, instead of calculating the estimated total *hypervolume* to decide which dataset to index, the authors simply index the first dataset using a grid approach. In DORIC [8], *Jin et al.* the relation computation problem is optimized by using existing links to infer new links. For instance, in case A equals B and A equals C, DORIC infers that B equal C. *Ahmed et al.* [11] studied the effect of simplifying the resources' geometries on the runtime and F-Measure of link discovery approaches over geospatial KGs. However, our approach computes topological relations using content measures as defined in [12] instead of computing the DE-9IM intersection matrix. Accuracy has received a considerable attention from the research society of link discovery. For instance, algorithms such as RADON [5], RADON2 [6], GIA.NT [7], and DORIC [8] achieve an F-Measure of 1, while algorithm such as the ones in [11] and our presented algorithm here (COBALT) scarify the accuracy in favour of efficiency.

6. Conclusion & Future Work

In this paper, we propose COBALT, an approach for topological relation discovery. COBALT combines the R-tree indexing with the content-based measures in order to scale up the topological relations discovery process. Based on our experiments, COBALT is able to achieve a speed up of up to 1.47×10^4 over state-of-the-art algorithms. On the other side, we also study the impact of applying our proposed approach on the accuracy of the generated links. In order to optimize COBALT, we propose two polygon splitting strategies. Without applying our splitting strategies, COBALT achieves an F-Measure between 70% and 90%. By applying our proposed splitting strategies, the F-Measure of COBALT is improved to up to 99%. In future work, we aim to improve the accuracy of COBALT by incorporating more non-spacial information into the linking process. In particular, we will consider information regarding the type, location, description, and name of the resources represented by the polygons in the linking process.

Acknowledgements

This work has been supported by the Ministry of Culture and Science of North Rhine-Westphalia (MKW NRW) within the project SAIL (grant no NW21-059D) and the European Union's Horizon Europe research and innovation program (grant no 101070305). The authors gratefully acknowledge the funding of this project by computing time provided by the Paderborn Center for Parallel Computing (PC2).

References

[1] Ngonga Ngomo AC. ORCHID - Reduction-Ratio-Optimal Computation of Geo-Spatial Distances for Link Discovery. In: Proceedings of ISWC 2013; 2013. p. 395-410.

[2] Auer S, Lehmann J, Hellmann S. LinkedGeoData: Adding a Spatial Dimension to the Web of Data. In: Bernstein A, Karger DR, Heath T, Feigenbaum L, Maynard D, Motta E, et al., editors. The Semantic Web - ISWC 2009. Berlin, Heidelberg: Springer Berlin Heidelberg; 2009. p. 731-46.

[3] Berners-Lee T. Linked-data design issues; 2009. W3C design issue document. Available from: https://www.w3.org/DesignIssues/LinkedData.html.

[4] Zhang C, Li W. The roles of web feature and web map services in real-time geospatial data sharing for time-critical applications. Cartography and Geographic Information Science. 2005;32(4):269-83.

[5] Sherif MA, Dreßler K, Smeros P, Ngomo ACN. Radon–rapid discovery of topological relations. In: Thirty-First AAAI Conference on Artificial Intelligence; 2017. .

[6] Ahmed AF, Sherif MA, Ngomo ACN. Radon2: a buffered-intersection matrix computing approach to accelerate link discovery over geo-spatial rdf knowledge bases. In: Ontology Matching: OM-2018: Proceedings of the ISWC Workshop; 2018. p. 197.

[7] Papadakis G, Mandilaras G, Mamoulis N, Koubarakis M. Progressive, Holistic Geospatial Interlinking. In: Proceedings of the Web Conference 2021. WWW '21. New York, NY, USA: Association for Computing Machinery; 2021. p. 833–844. Available from: https://doi.org/10.1145/3442381.3449850.

[8] Jin X, Eom S, Shin S, Lee KH, Hong C. DORIC: discovering topological relations based on spatial link composition. Knowledge and Information Systems. 2021 Oct;63(10):2645-69. Available from: https://doi.org/10.1007/s10115-021-01603-2.

[9] Clementini E, Di Felice P, van Oosterom P. A small set of formal topological relationships suitable for end-user interaction. In: Abel D, Chin Ooi B, editors. Advances in Spatial Databases. Berlin, Heidelberg: Springer Berlin Heidelberg; 1993. p. 277-95.

[10] Egenhofer MJ, Mark DM, Herring J. The 9-Intersection: Formalism and Its Use for Natural-Language Spatial Predicates (94-1). UC Santa Barbara: National Center for Geographic Information and Analysis; 1994.

[11] Ahmed AF, Sherif MA, Ngomo ACN. On the Effect of Geometries Simplification on Geo-spatial Link Discovery. Procedia Computer Science. 2018;137:139-50. Proceedings of the 14th International Conference on Semantic Systems 10th – 13th of September 2018 Vienna, Austria. Available from: https://www.sciencedirect.com/science/article/pii/S1877050918316193.

[12] Godoy F, Rodríguez A. Defining and Comparing Content Measures of Topological Relations. Geoinformatica. 2004 dec;8(4):347–371. Available from: https://doi.org/10.1023/B:GEIN.0000040831.81391.1d.

[13] Balasubramanian L, Sugumaran M. A State-of-Art in R-Tree Variants for Spatial Indexing. International Journal of Computer Applications. 2012 03;42:35-41.

[14] Guttman A. R-Trees: A Dynamic Index Structure for Spatial Searching. SIGMOD Rec. 1984 jun;14(2):47–57. Available from: https://doi.org/10.1145/971697.602266.

[15] Kamel I, Faloutsos C. On Packing R-Trees. In: Proceedings of the Second International Conference on Information and Knowledge Management. CIKM '93. New York, NY, USA: Association for Computing Machinery; 1993. p. 490–499. Available from: https://doi.org/10.1145/170088.170403.

[16] Leutenegger S, Lopez M, Edgington J. STR: A Simple and Efficient Algorithm for R-Tree Packing; 1997. p. 497-506.

[17] Lee T, Lee S. OMT: Overlap Minimizing Top-down Bulk Loading Algorithm for R-tree. In: CAISE Short paper proceedings. vol. 74; 2003. p. 69-72.

[18] Kamel I, Faloutsos C. Hilbert R-Tree: An Improved R-Tree Using Fractals. In: Proceedings of the 20th International Conference on Very Large Data Bases. VLDB '94. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.; 1994. p. 500–509.

[19] Beckmann N, Kriegel HP, Schneider R, Seeger B. The R*-tree: an efficient and robust access method for points and rectangles. In: SIGMOD '90; 1990. .

[20] Büttner G, Kosztra B. CLC2018 Technical Guidelines. European Environment Agency; 2017. Available from: https://land.copernicus.eu/user-corner/technical-library/clc2018technicalguidelines_final.pdf.

[21] Douglas DH, Peucker TK. ALGORITHMS FOR THE REDUCTION OF THE NUMBER OF POINTS REQUIRED TO REPRESENT A DIGITIZED LINE OR ITS CARICATURE. Cartographica: The International Journal for Geographic Information and Geovisualization. 1973;10:112-22.

[22] Ahmed AF, Sherif MA, Ngomo ACN. On the Effect of Geometries Simplification on Geo-spatial Link Discovery. In: SEMANTiCS 2018 - Research Track. SEMANTiCS '18; 2018. Available from: http://svn.aksw.org/papers/2018/SEMANTICS_GeoSimp/paper/public.pdf.

[23] Sherif MA, Ngomo ACN. A systematic survey of point set distance measures for link discovery. Semantic Web Journal(Cited on page 18). 2015.

[24] Smeros P, Koubarakis M. Discovering Spatial and Temporal Links among RDF Data. In: LDOW@ WWW; 2016. .