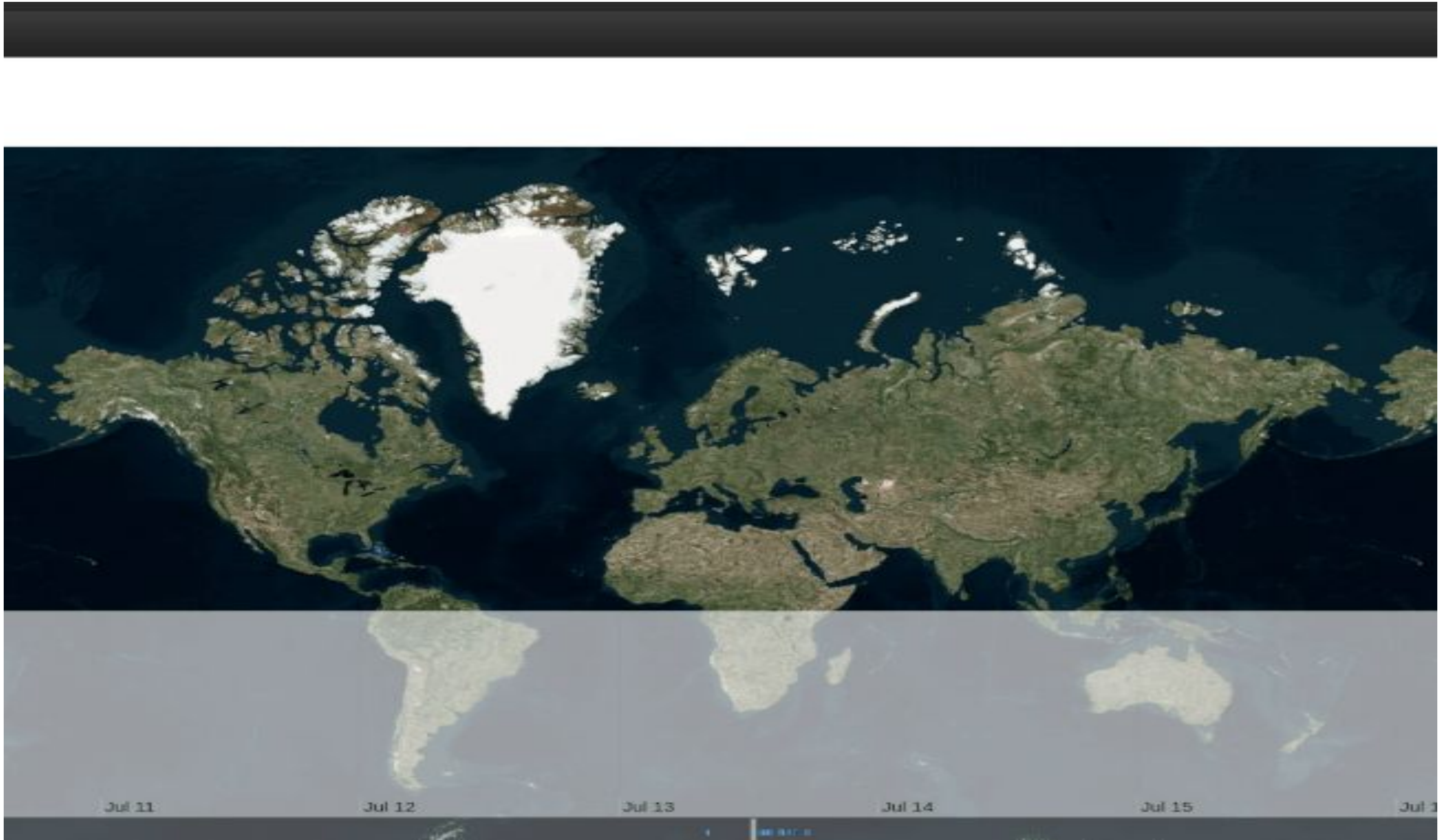
Interlinking geospatial RDF data

Geospatial Interlinking in action



Geospatial Interlinking

Input:

- A topological relation ***R***
- A source dataset of geometries ***S***
- A target dataset of geometries ***T***

  Types of Geometries:

  - LineStrings
  - Polygons

Output:

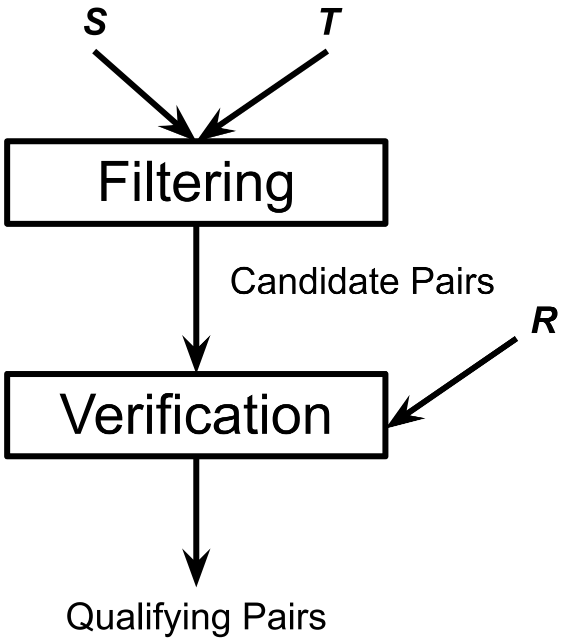
- All pairs (s,t) ∈ ***S*** x ***T*** such that ***R(s,t) = true***

Challenges:

- quadratic time complexity, **O(n^2)**
- time-consuming topological relations over complex geometries

Filtering – Verification Framework

Two-step procedure to reduce the quadratic time complexity:



Filtering, a.k.a. Space Tiling

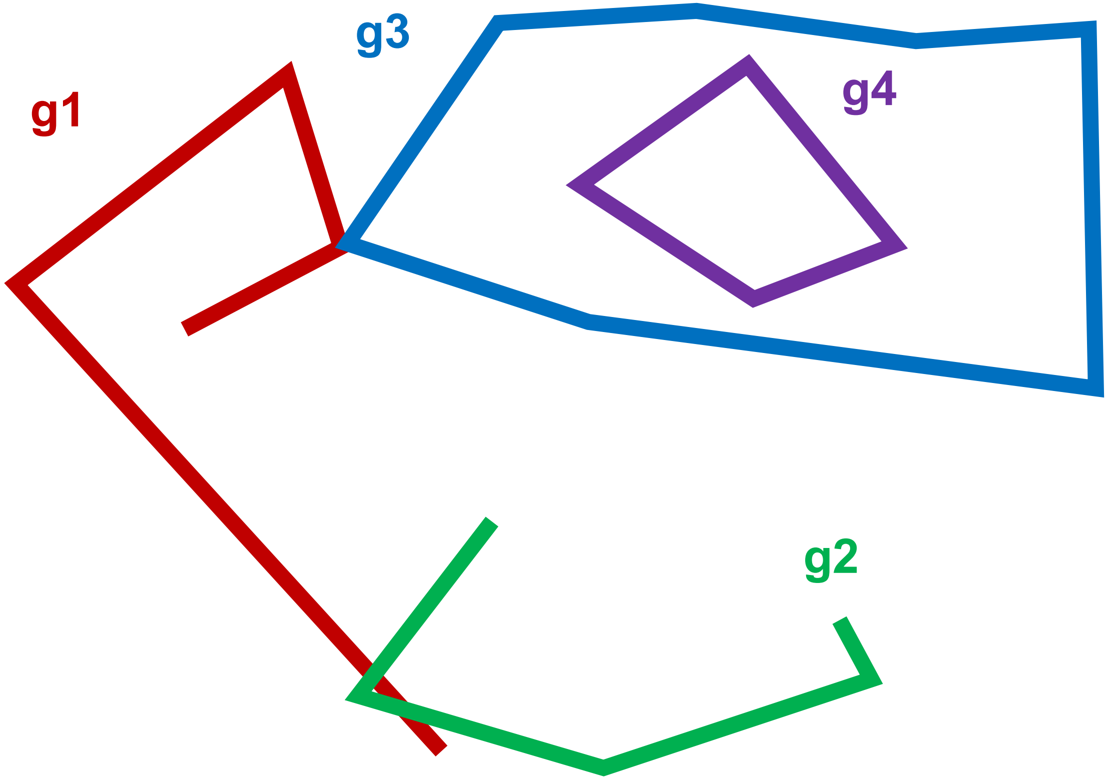
Involves three steps:

1. We define an ***Equigrid*** on Earth's surface
2. We index geometries according to their ***Minimum Bounding Rectangle***
3. We define as ***candidate pairs*** only the geometries that share at least one tile

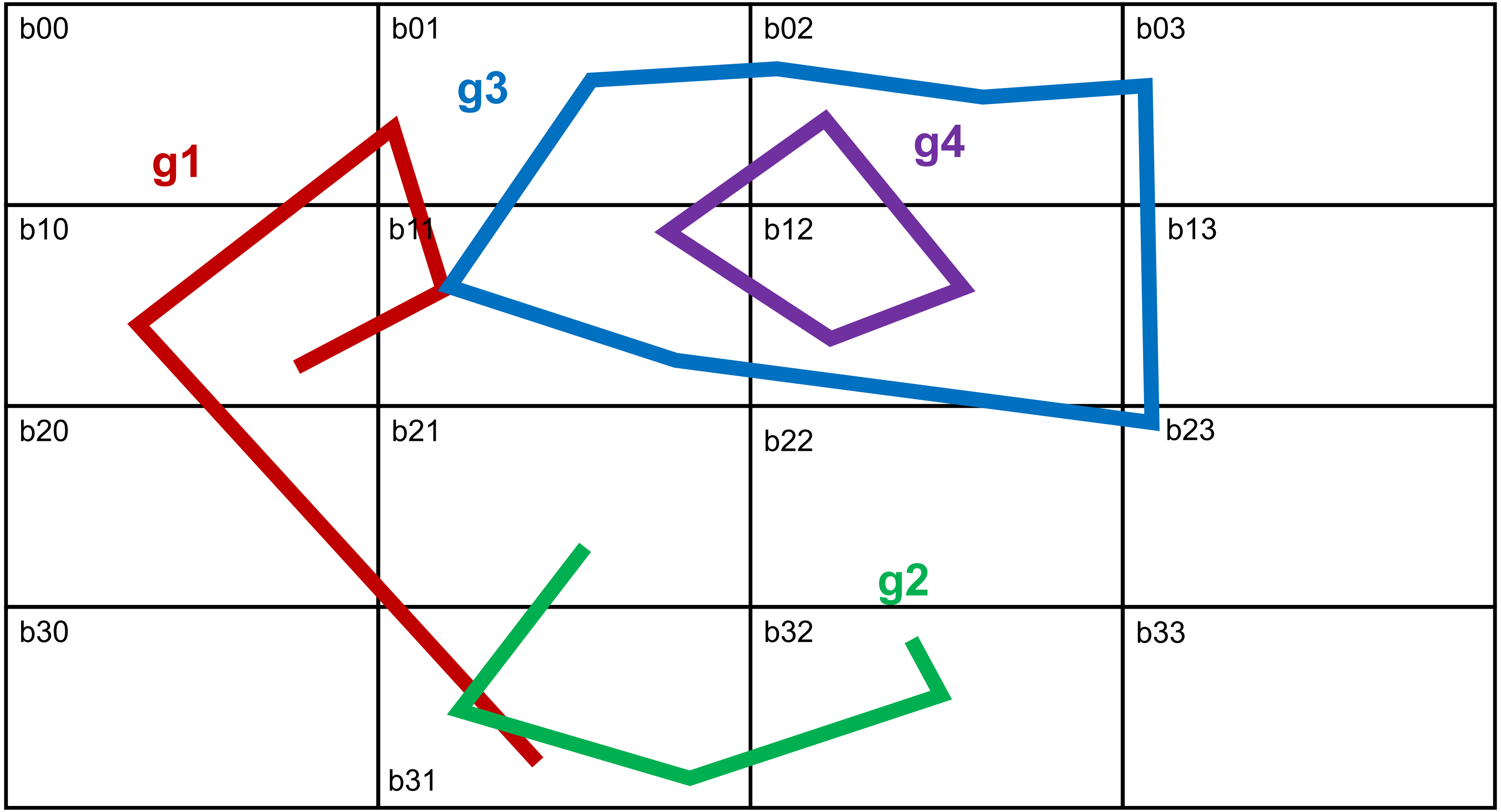
Advantages:

- Exact process
- Linear time complexity O(n)
- Significant gains in efficiency

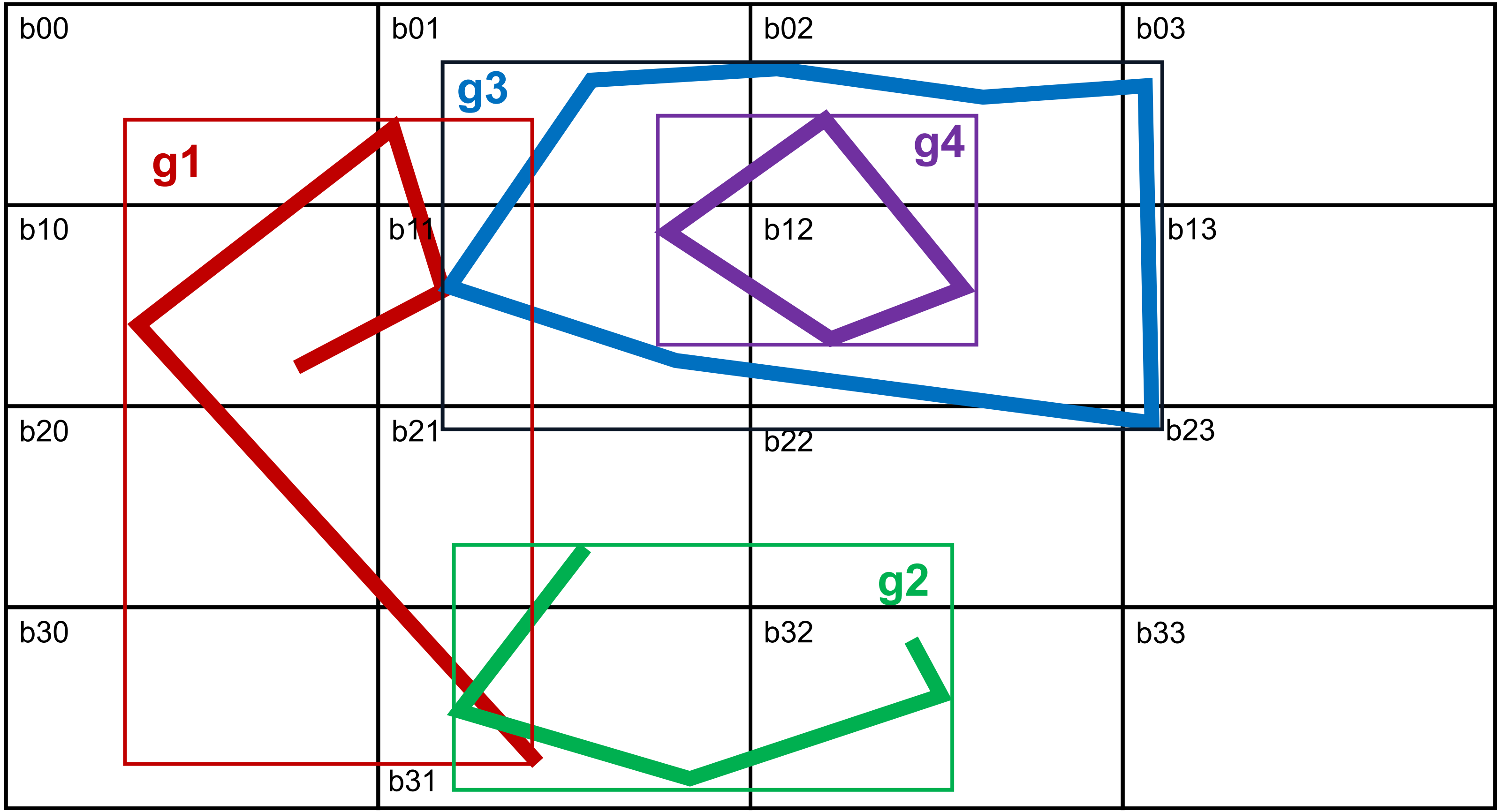
Space Tiling Example



Space Tiling Example - Equigrid



Space Tiling Example - MBR indexing

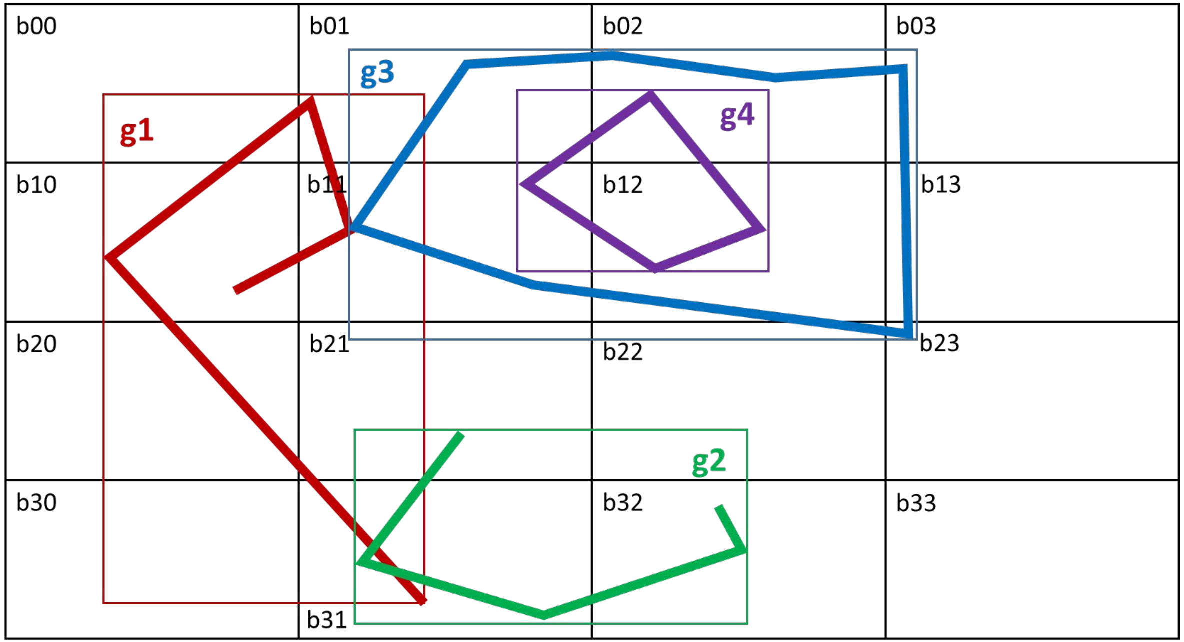


Space Tiling Example – Candidate Pairs

Just **3** pairs:

$g_1 - g_2$
$g_1 - g_3$
$g_3 - g_4$

50% lower than the **6** pairs of the brute-force approach.

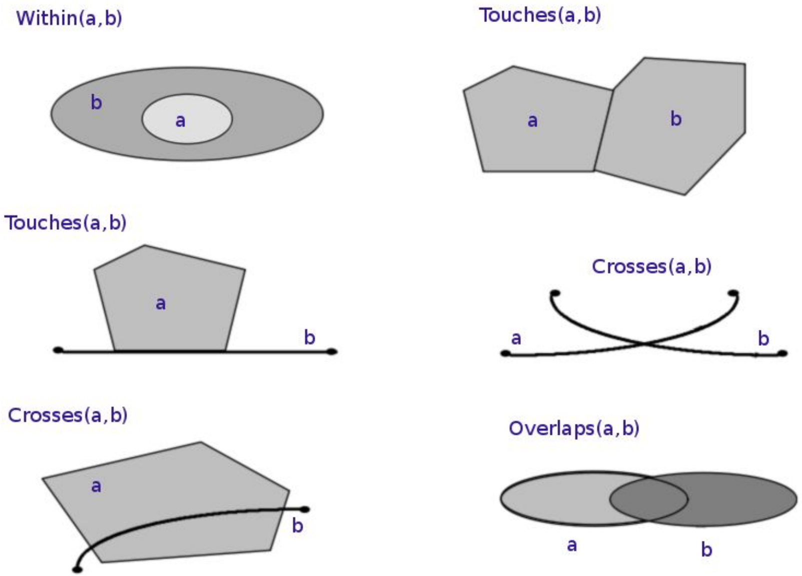


Verification

Two different types:

1. Proximity relations (such as `dbp:near`) with a distance threshold
   - e.g., find all cities from ***S*** that are less than 1km away from any river in ***T***
2. Topological relations according to the Dimensionally Extended 9-Intersection Model (**DE9IM**)
   - Equals
   - Touches
   - Contains
   - Covers
   - Intersects
   - Within
   - CoveredBy
   - Crosses
   - Overlaps

   - Disjoint



ORCHID

Filtering:

- **Static** space tiling
- Granularity for width and height = **θ / R / a**
  - a = 1

Verification:

- Hausdorff distance $hd(s,t) = \max_{s_i \in S}\{\min_{t_i \in T}\{\delta(s_i,t_j)\}\} <= \theta$
- Optimizations for efficient computation:
  - Bounding circles
  - Cauchy-Swarz Inequality for Distance Approximation

Open-source implementation (https://github.com/dice-group/LIMES)

Axel-Cyrille Ngonga Ngomo: ORCHID - Reduction-Ratio-Optimal Computation of Geo-spatial Distances for Link Discovery. International Semantic Web Conference, 2013: 395-410

Silk-spatial

Filtering:

- **Static** space tiling
- Granularity for width and height = **$1/a^{o\,2}$**
    - a = 10

Verification:

- DE9IM topological relations – single relation per run
- Massive parallelization (Apache Hadoop)

Open-source implementation (https://github.com/silk-framework/silk)

Panayiotis Smeros, Manolis Koubarakis: Discovering Spatial and Temporal Links among RDF Data. LDOW@WWW 2016

RADON

Filering:

- Swapping strategy
- **Dynamic** space tiling
  - Width = ½ · ($\text{average}_{s \in S}$(s.width) + $\text{average}_{t \in T}$(t.width))
  - Length = ½ · ($\text{average}_{s \in S}$(s.length) + $\text{average}_{t \in T}$(t.length))

Verification:

- DE9IM topological relations – single relation per run
  - Relation-based optimizations
  - Hash-based redundancy elimination
- Multi-core parallelization

Open-source implementation (https://github.com/dice-group/LIMES)

Mohamed Ahmed Sherif, Kevin Dreßler, Panayiotis Smeros, Axel-Cyrille Ngonga Ngomo: Radon - Rapid Discovery of Topological Relations. AAAI 2017: 175-181

stLD

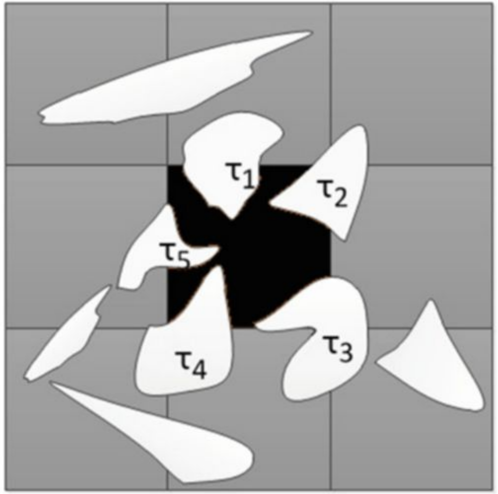
Filtering:

- **Static** Index
- Variety of approaches (e.g., R-Trees, Equigrid, Hierarchical Grid)
- Indexes exclusively the source dataset ***S***
- MaskLink algorithm

Verification:

- Both topological and proximity relations – single relation per run
- Massive parallelization (Apache Flink)
- Suitable for streams

Implementation not available.



Georgios M. Santipantakis, Apostolos Glenis, Christos Doulkeridis, Akrivi Vlachou, George A. Vouros: stLD: towards a spatio-temporal link discovery framework. SBD@SIGMOD 2019: 4:1-4:6
Georgios M. Santipantakis, Christos Doulkeridis, George A. Vouros, Akrivi Vlachou: MaskLink: Efficient Link Discovery for Spatial Relations via Masking Areas. CoRR abs/1803.01135 (2018)

GIA.nt: Geospatial Interlinking At large – Part A

Improving RADON's **Filtering**:

- **Dynamic** space tiling, based exclusively on the source dataset **S**
  - Width = $\text{average}_{s \in S}$(s.width)
  - Length = $\text{average}_{s \in S}$(s.length)
- No dataset swapping

Lower running time

- Target dataset (=largest input dataset) is read one by one from the **disk**
- Inherent removal of redundant (i.e., repeated) geometry pairs

Memory requirements lower by >50%

  - Easily parallelizable in MapReduce, due to its geometry-centric functionality

George Papadakis, Georgios M. Mandilaras, Nikos Mamoulis, Manolis Koubarakis. Progressive, Holistic Geospatial Interlinking. WWW 2021: 833-844

GIA.nt: Geospatial Interlinking At large – Part B

Improving RADON's **Verification**:

- **Holistic** Geospatial Interlinking:

  Simultaneous estimation of all DE9IM topological relations → **Intersection Matrix**

Run-time lower by **>80%**

$$\mathrm{DE9IM}(a,b) = \begin{bmatrix} \dim(I(a)\cap I(b)) & \dim(I(a)\cap B(b)) & \dim(I(a)\cap E(b)) \\ \dim(B(a)\cap I(b)) & \dim(B(a)\cap B(b)) & \dim(B(a)\cap E(b)) \\ \dim(E(a)\cap I(b)) & \dim(E(a)\cap B(b)) & \dim(E(a)\cap E(b)) \end{bmatrix}$$

Examples:

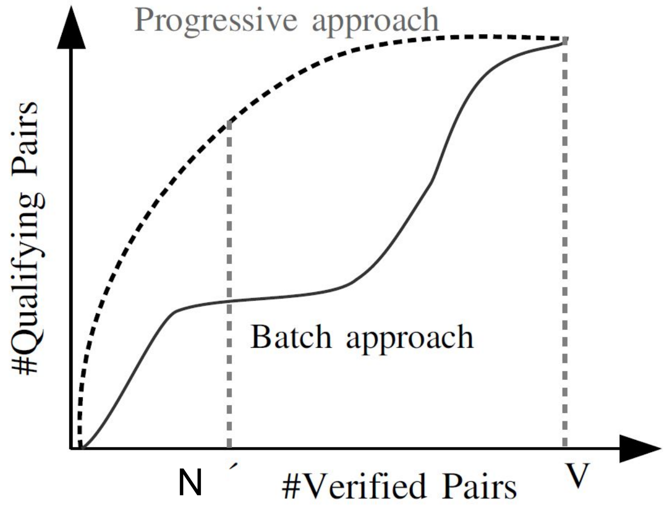
Equals	$\begin{bmatrix} T & * & F \\ * & * & F \\ F & F & * \end{bmatrix}$
	T*F**FFF*

Intersects	$\begin{bmatrix} T & * & * \\ * & * & * \\ * & * & * \end{bmatrix}$	$\begin{bmatrix} * & T & * \\ * & * & * \\ * & * & * \end{bmatrix}$	$\begin{bmatrix} * & * & * \\ T & * & * \\ * & * & * \end{bmatrix}$	$\begin{bmatrix} * & * & * \\ * & T & * \\ * & * & * \end{bmatrix}$
	T********	*T*******	***T*****	****T****

Progressive Geospatial Interlinking

Ideal for applications with limited resources:

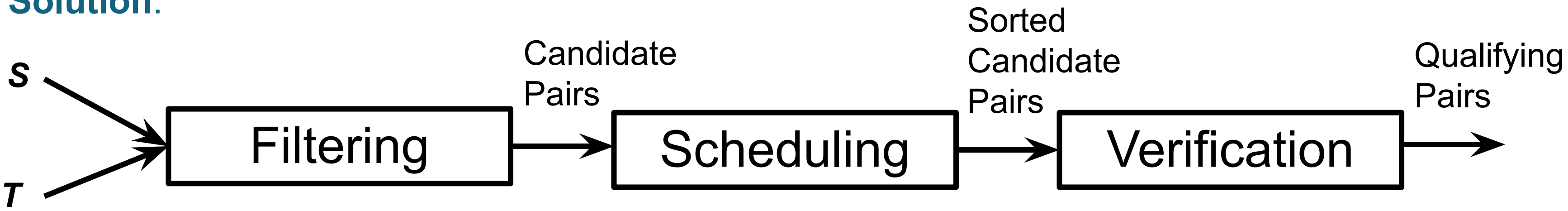
- Temporal or computational (e.g., Amazon Lambda functions)



Requirements with respect to batch approaches [1]:

1. Same Eventual Quality
2. Improved Early Quality
   - Measured through Progressive Geometry Recall (PGR)

Solution:



[1] *Steven Euijong Whang, David Marmaros, Hector Garcia-Molina: Pay-As-You-Go Entity Resolution. IEEE Trans. Knowl. Data Eng. 25(5): 1111-1124 (2013)*

Progressive GIA.nt

Input:

- Budget B + source dataset + target dataset

Filtering:

- Same as batch GIA.nt

Scheduling:

- Priority queue with top-B weighted candidate pairs based on either of the following functions:
  - Co-occurrence Frequency (CF): #common tiles
  - Jaccard Similarity (JS): normalized CF
  - Pearson's χ^2 test (χ^2): degree to which ***s*** and ***t*** occur independently in tiles

higher scores → more likely to satisfy at least one topological relation

Verification:

- Processes the pairs of the priority queue in decreasing weight

George Papadakis, Georgios M. Mandilaras, Nikos Mamoulis, Manolis Koubarakis. Progressive, Holistic Geospatial Interlinking. WWW 2021: 833-844

Dynamic Progressive Geospatial Interlinking

Improved Static Progressive Geospatial Interlinking in three ways:

1. New weighting schemes based on the complexity of geometries.
   - Normalized MBR overlap → higher effectiveness

$$MBR(s,t) = \frac{MBR(s \cap t)}{MBR(s \cup t)} = \frac{MBR(s \cap t)}{MBR(s) + MBR(t) - MBR(s \cap t)}$$

   - Inverse sum of points → higher time efficiency

$ISP(s,t) = \frac{1}{p(s)+p(t)}$, where p(g) denotes the number of boundary points

2. Composite weighting schemes → higher effectiveness, more deterministic behavior
   - the primary one is used for scheduling all pairs
   - the secondary one is used for resolving the ties

3. Dynamic Progressive GIA.nt

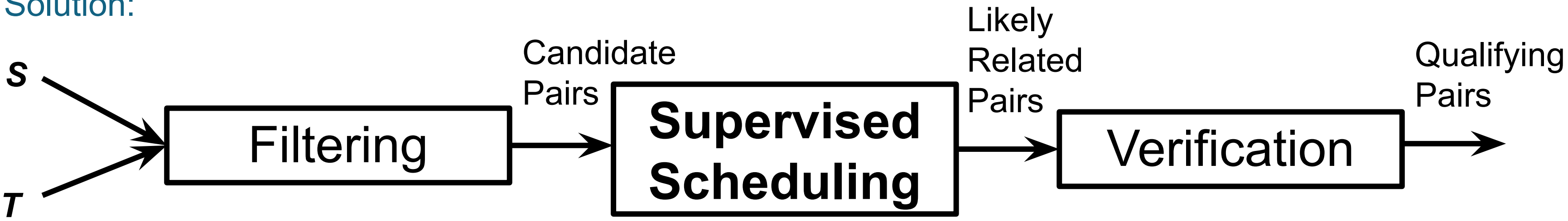
George Papadakis, Georgios M. Mandilaras, Nikos Mamoulis, Manolis Koubarakis. Static and Dynamic Progressive Geospatial Interlinking. ACM TSAS (to appear)

Supervised Progressive Geospatial Interlinking

Drawbacks of Progressive Geospatial Interlinking:

- Store the top-BU weighting pairs in main memory
- Might be hard to fine-tune BU
- Considers at most two sources of evidence, i.e., composite weighting schemes

Solution:



1. Filtering → as in (Batch & Progressive) GIA.nt
2. Supervised Filtering
   - Classify candidate pairs into “**`likely related pairs`**” & “**`unlikely related pairs`**” using a feature vector
3. Verification → as in Batch GIA.nt

Supervised Filtering

Challenges:

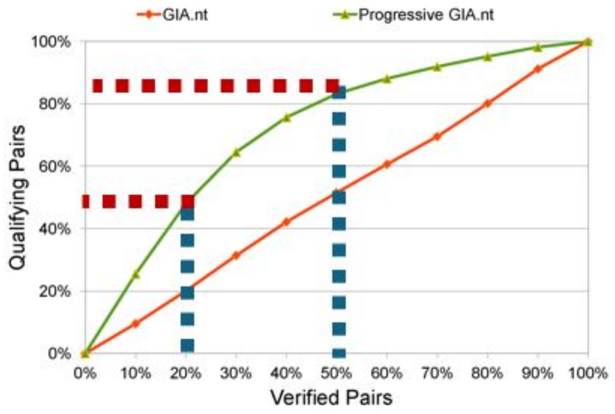
- Define generic, effective & efficient features
- Avoid any human intervention
- Address class imbalance
- Minimize the feature and the training set → simple & efficient classification models

Approach outline:

- **Self-supervised learning based on undersampling**
- 4 categories of features
    1. Area-based (source/target/intersection MBR area)
    2. Boundary-based (source/target #boundary points and boundary length)
    3. Grid-based (#common tiles, #tiles intersecting the target MBR)
    4. Candidate-based (total/distinct/real candidates per source/target geometry)
- 2 sub-categories in each case:
    - Atomic features
    - Composite features

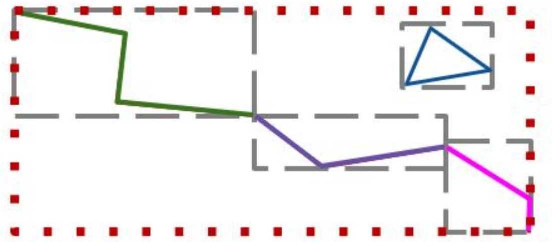
Future directions

- Proactive Geospatial Interlinking
  - Terminate Geospatial Interlinking automatically as soon as recall exceeds a desired level → minimize the time required for processing voluminous datasets



- Generalize to 3-dimensional data
  - Silk-spatial: 3rd dimension = time
  - stLD: 3rd dimension = height (e.g,. aviation data)

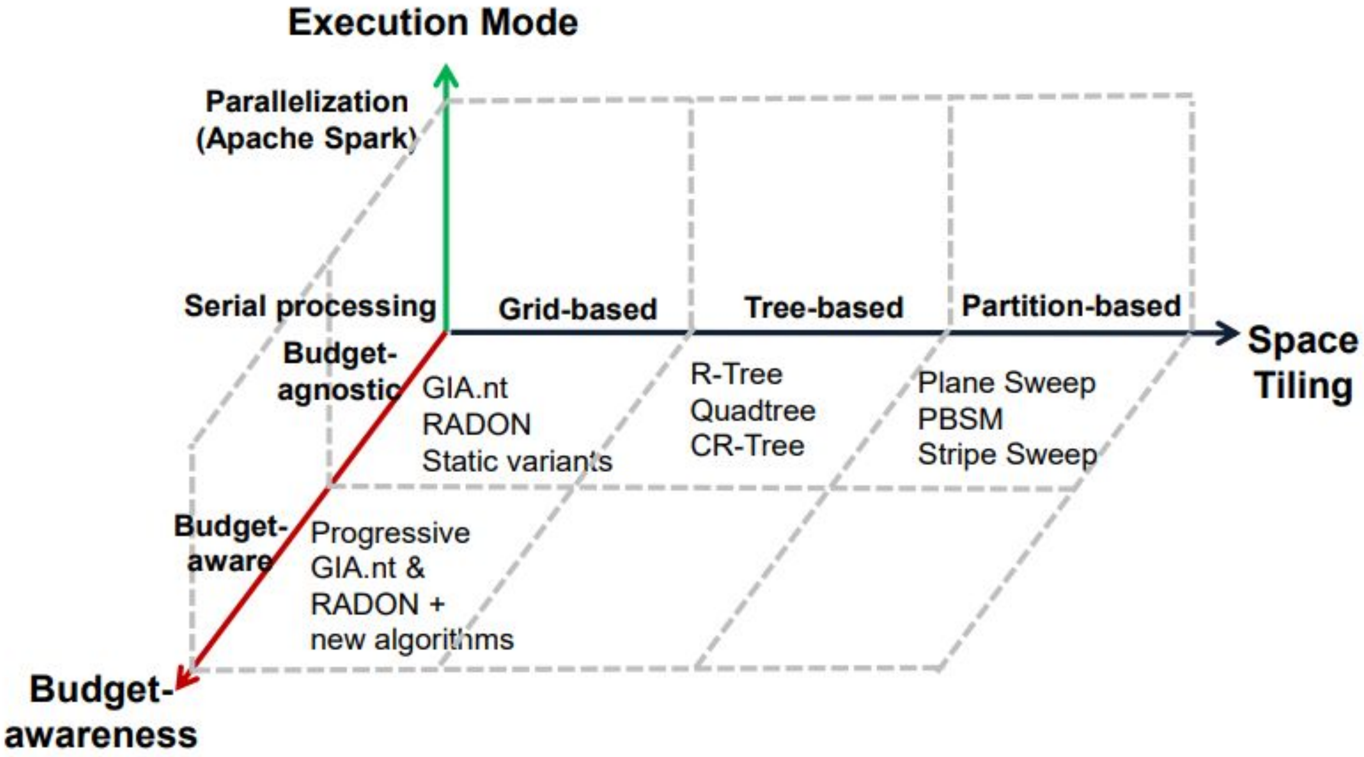
- Improve Intersection Matrix computation
  - O(n · logn) [1]
  - Fine-grained MBR



[1] *Edward P. F. Chan, Jimmy N. H. Ng: A General and Efficient Implementation of Geometric Operators and Predicates. SSD 1997: 69-93*

JedAI-spatial demonstration

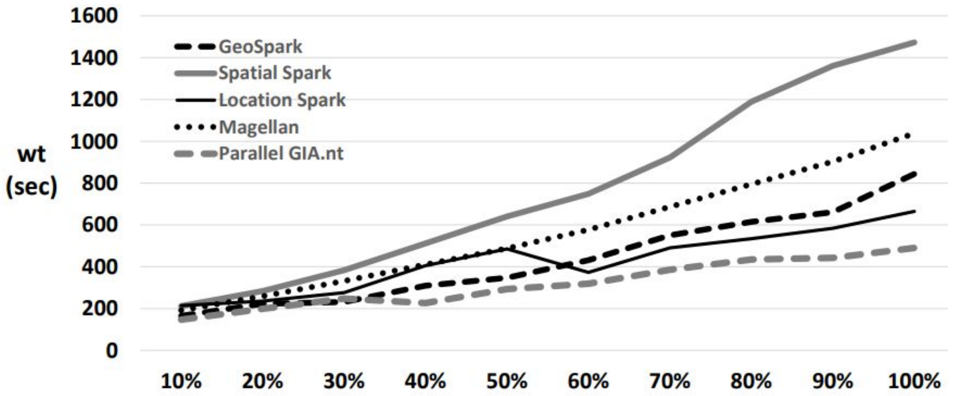
JedAI-spatial



Preliminary implementation available at: https://github.com/AI-team-UoA/JedAI-spatial

Parallel Algorithms

- Common three-stage pipeline for the state-of-the-art parallel joins:
  - GeoSpark, i.e., Apache Sedona
  - Spatial Spark
  - Magellan
  - Location Spark
  - Parallel GIA.nt



Scalability Analysis over D1
(|S|=2.3M, |T|=5.8M, |C|=6.3M)