RADIG: A Resolution-adaptive Grid Reference Framework for Persistent Spatial Index

Sheng Zhou[*1] and Jonathan Simmons [†1]

[1]GDS Data Science, Ordnance Survey

GISRUK 2025

Summary

RADIG is a library for generating bit or character strings to create persistent and interoperable indexes for geospatial features, especially in large-scale distributed/parallel computing environments. The initial containment and intersection test in a spatial join may be performed by a string prefixing operation. A synchronised traversal algorithm is developed to perform efficient prefixing test on multi-way tried built from two string sets.

KEYWORDS: Spatial Index, Digit Interleaving, Databricks, Trie, Parallel Computing

1. Introduction

A major challenge in processing geospatial data on large-scale distributed/parallel platforms such as Azure Databricks is to partition and index data spatially for efficient spatial operations, and (ideally) make the spatial index persistent and interoperable. (Apache Sedona) introduced customised spatial data partitioner and proper R-tree based spatial index, but it is difficult to use and persist, and not interoperable. Subsequently Databricks introduced its own geospatial library (Mosaic) using H3 for spatial indexing.

H3 is a hierarchical grid system (HGS) which consists of several grids at different levels of granularity or resolution. Different grid systems may use different grid cell shapes. A unique bit/character string is assigned to each grid cell as a 1-dimensional reference for all locations in the cell. A geospatial feature may be indexed by collecting the references of all grid cells (at a single or multiple levels of grids) relating to the feature spatially. These references may be made persistent as an attribute of the original feature.

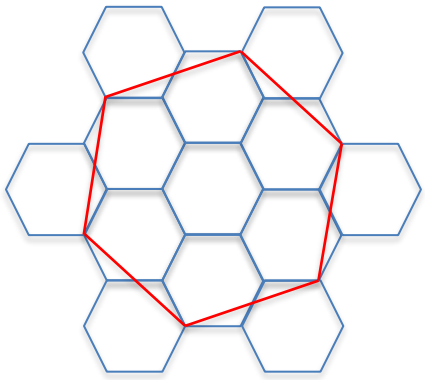

Figure 1 Two levels of Hexagons in H3 Index

Some HGS, e.g. HTM (Kunszt et al, 2001) and Geohash (Niemeyer, 2008), are sub-divisive, that is, a

[*] Sheng.Zhou@os.uk
[†] Jonathan.Simmons@os.uk

grid cell is spatially subdivided to generate child grid cells at next lower level of grid. H3 is not a strict subdivision grid system (**Figure 1**) but Mosaic also supports indexing in sub-divisive British National Grid (BNG).

Two grid cell references refer to the same grid cell if they are equal by string comparison. This enables spatial join on two dataframes with geometry columns using Mosaic generated index references in Databricks. This equality test implies that the references in two dataframes must be generated at the same grid resolution. Also, a large feature indexed at a small resolution will result in too many references/rows generated; alternatively, a small feature indexed at a large resolution will result in poor space utilisation (too many false positive intersections during joining).

Ideally, a feature may be indexed at a resolution adaptive to its size, or even at multiple resolutions. Also, the resolution change between two adjacent levels of grid should be relatively small in order to improve space utilisation.

2. Reference Prefixing

In some HGS a grid cell reference is a prefix of the references of some other grid cells at the lower level. In a sub-divisive HGS if the prefixing relation is between a grid cell and only those cells sub-divided from it, the cell containment relationship may also be determined by a string prefix testing (i.e. whether one grid reference is the prefix of the other reference). For example, the HTM grid cell with reference S2 is subdivided into four cells S20, S21, S22 and S23 at the lower level (**Figure 1**). Since S2 is the prefix of S20, S21, S22 and S23, we may determine that the grid cell S2 contains cells S20, S21, S22 and S23 without applying spatial predicates:

```
"S20".startsWith("S2") is true => cell("S2").contains(cell("S20")) is true
```

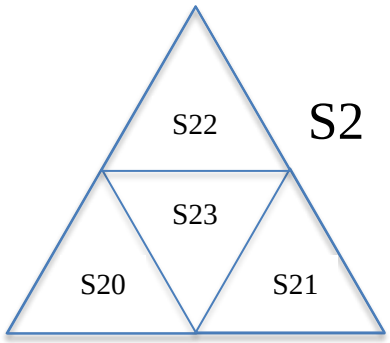

Figure 2 HTM Grid Cell Subdivision and Grid References

Here `startsWith()` is a string prefixing test method commonly available (maybe under different names) in database systems.

In conclusion, in a sub-divisive HGS with reference prefixing, spatial join on mixed resolution references may be performed using string operation only.

3. RADIG – A Resolution Adaptive Digit-Interleaving HGS

3.1. Griding by Digits

Given a spatial extent and a coordinate system to represent locations in the space, each digit place in an ordinate value may be viewed as a partition to the extent in the corresponding dimension. Consequently, a hierarchical partition is formed going from the uppermost digit to the lowermost digit. For the example of a 2D planar space extent of X = [0, 10^7) and Y = [0, 10^7) in decimal(base$_{10}$) number

system with 3 decimal places, a location will be represented as ($x_6x_5x_4x_3x_2x_1x_0.x_{-1}x_{-2}x_{-3}$ $y_6y_5y_4y_3y_2y_1y_0.y_{-1}y_{-2}y_{-3}$) with 7 integral digits. The pair of uppermost digits (x_6, y_6) form a 10x10 grid partition on the entire extents with cell size of 10^6x10^6. Each cell is represented by a pair of digit values, e.g. [2, 3] for the cell with corners (2000000 3000000) and (3000000 4000000). Similarly, digits (x_6x_5, y_6y_5) form a further 10x10 partition on each (x_6, y_6) cell, and so on. This is obviously a sub-divisive process.

3.2. Digit-Interleaving for Cell Reference

A method to pack a digit pair into a 1-dimensional reference string is via bit/digit interleaving (Samet, 1990):

$$(x_mx_{m-1}\ldots x_1, y_my_{m-1}\ldots y_1) \Rightarrow x_my_mx_{m-1}y_{m-1\ldots}x_1y_1$$

For example, (23, 45) is mapped to reference 2435 (padding with '0' may be required to make X and Y digits aligned at equal length).

For relative large number base, the resolution change between two grid levels may be too higher for the sake of space utilisation, e.g.,10 times change for decimal coordinates. A dual-divisor system is used to address this issue:

```
base = first_divisor x second_divisor
```

For example, for a base 10 = 2x5 setting, a cell is first divided into 2x2= 4 cells and then each of the 4 cells is divided into 5x5=25 cells. As a result, a digit will be represented by two digits in the generated reference string.

A single divisor digit-interleaved coordinate string representing a grid cell (X, Y) with *m* integral digits and *p* decimal digits has the following form:

$$Gd_1d_2d_3d_4x_my_mx_{m-1}y_{m-1}\ldots x_1y_1x_{-1}y_{-1}\ldots x_{-p}y_{-p}$$

Here $Gd_1d_2d_3d_4$ is the reference header served as a grid descriptor. Users may customise these parameters to create their own specific grids.

- G: either X or Y. X for encoding x ordinate first
- d_1: base – 1 (number base, currently permitted values: 1-9 and A-F, supporting granularity between 1:4 ($base_2$) and 1:256 ($base_{16}$))
- d_2d_3: number of integer digits (in base d_1) used in encoded coordinate string. Permitted value: "01" to "99".
- d_4: first-divisor – 1 (permitted values: 1- 9, A-F). d_4 equals to d_1 indicates a single-divisor grid.

Give a decimal grid cell (x_6, y_6) and dual divisor $div_1 = 2$, $div_2 = 5$, the encoded dual-divisor string is:

$$X9071x_{6\_1}y_{6\_1}x_{6\_2}y_{6\_2}$$

Where: $x_{6\_1} = [x_6 / div_2]$, $y_{6\_1} = [y_6 / div_2]$, $x_{6\_2} = x_6 \% div_2$, $y_{6\_2} = y_6 \% div_2$

Here […] is the integral part of a number and % is the modular operation. For example, (1,7) will be encoded into "0112".

Grid cell references generated by digit-interleaving is inherently prefixing. Therefore, this HGS will support spatial join on two datasets with mixed-resolution references. It is worth noting that although BNG is sub-divisive, BNG grid references are generated by digit concatenation rather than interleaving so they are not prefixing.

3.3. References At Multiple Grid Levels

RADIG support generation of grid cell references for features at pre-defined resolutions, resolutions adaptive to feature size, and at multiple grid levels controlled by the following mechanism (**Figure 3**).

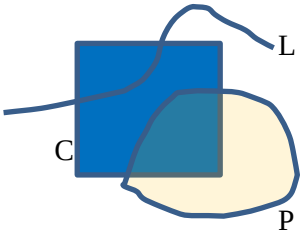

Figure 3 A grid cell C with an intersecting linestring L and polygon P

If the ratios EA=Area(Intersection(C, P))/Area(C) or EL=Length(Intersection(L, C))/Length(side_of_C) is lower than pre-defined thresholds, the process will sub-divide C and generate references at a finer resolution.

Figure 4 is an example of grid subdivision result. **Figure 5** illustrates the reference generation process.

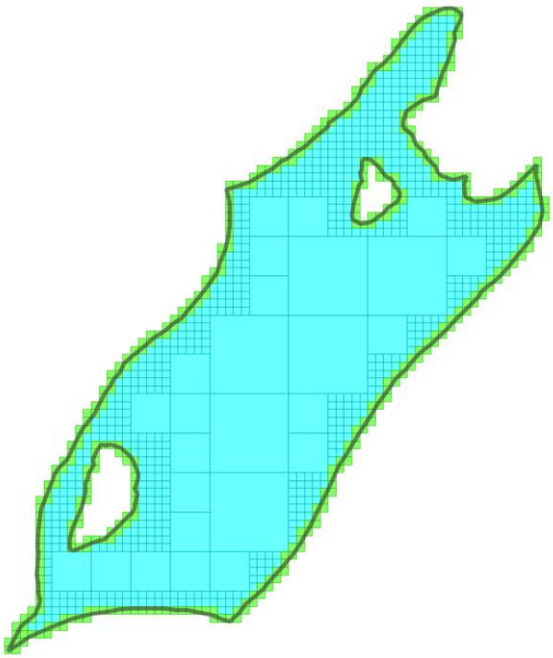

Figure 4 Boundary and interior grid cells at multiple levels for a polygon

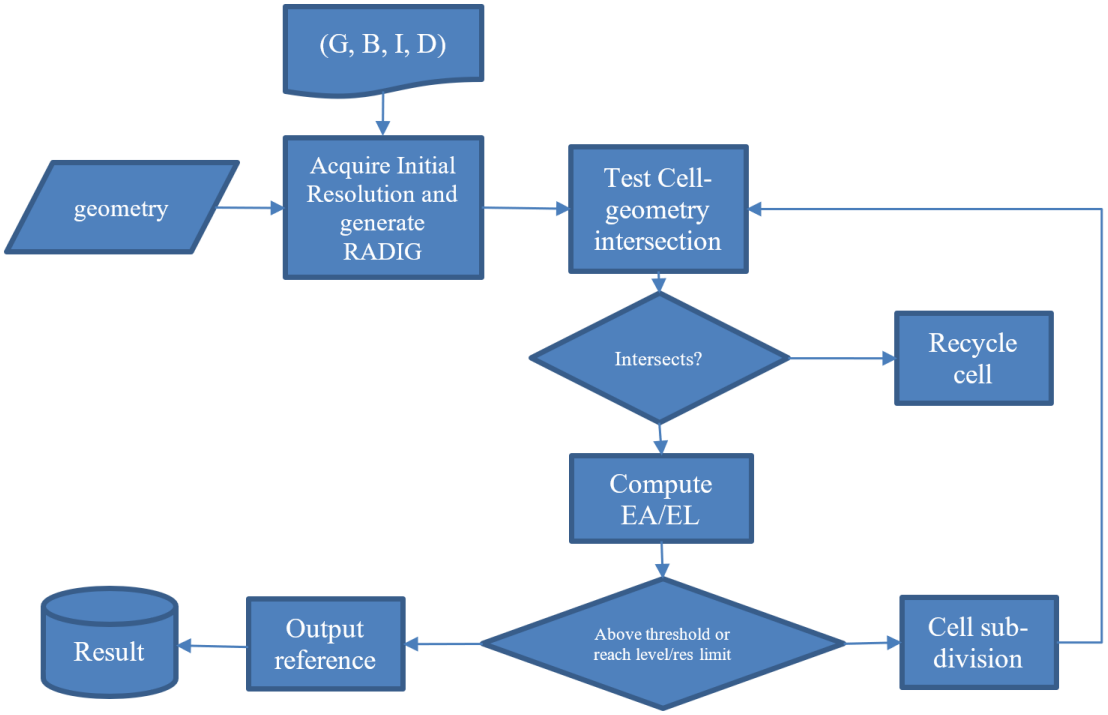

Figure 5 RADIG reference generation process flow

4. RADIG Match Trie for Efficient Prefix Matching

For two dataframes with RADIG reference columns, spatial join by containment or intersection may be performed as:

```
containDF = dfFeature.join(dfQuery, dfFeature.RadigRef.startsWith(dfQuery.RadigRef))

intersectDF = df1.join(df2, df1.RadigRef.startsWith(df2.RadigRef)|df2.RadigRef.startsWith(df1.RadigRef))
```

Since `startsWith` method is not yet optimised on many platforms for large scale query, the above queries are often translated into inefficient full cross-joins. To address this problem, we developed a "synchronised traversal" algorithm for prefix test between two sets of strings. We first build two multi-way tries (a.k.a. Prefix tree) for reference strings in two datasets to be joined (**Figure 6**) and then traverse on one trie while visiting the corresponding node on the other (**Figure 7**). The time complexity for this algorithm is *O*(**a*****m*****N**) where **a** is the average key length, **m** the alphabet size and **N** the maximum Trie size.

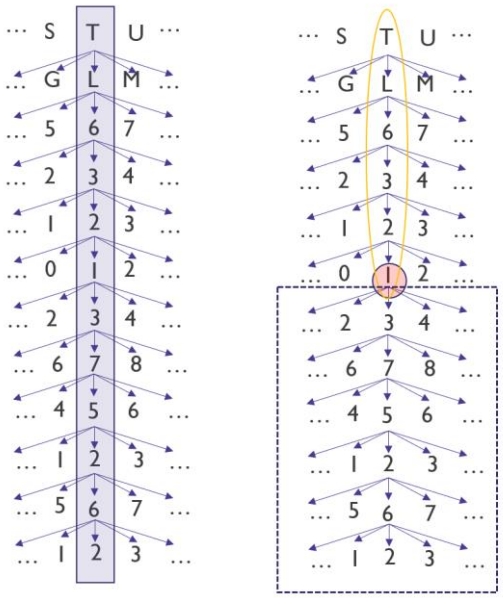

Figure 6 A Trie containing String TL6321375262 (left) and a query with String TL6321 (right)

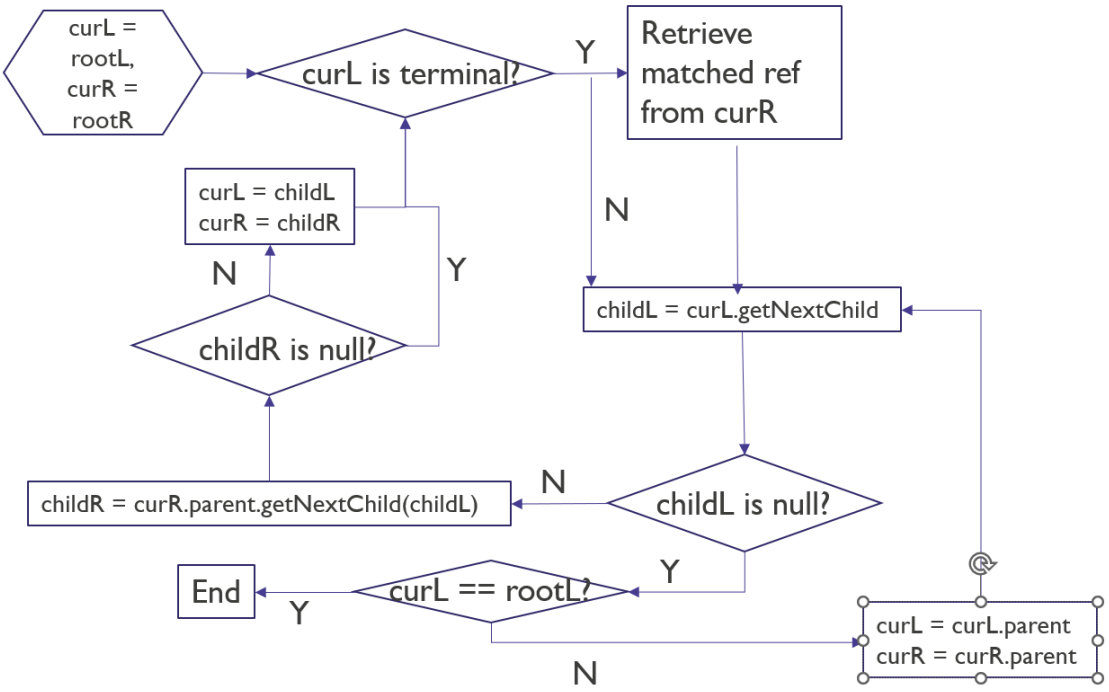

Figure 7 Synchronised Traversal of two multiway tries

5. Implementation and Discussion

RADIG is implemented in Java with Scala and Python wrappers for Databricks. A repo will be made public at:

https://github.com/OrdnanceSurvey/Radig2

Our experiments on joining 40M address points with 14.8M building polygons using synchronised trie traversal yields performance slower but at the same magnitude as the join based on string equality. We consider it quite acceptable for the benefits brought by RADIG references: persistent and interoperable single reference column for joining with other datasets indexed at different/mixed

resolutions.

References

Apache Sedona. https://sedona.apache.org/latest/. Accessed 13 Dec. 2024

H3. https://h3geo.org/docs/highlights/indexing/. Accessed 13 Dec. 2024

Kunszt P Z, Szalay A S, Alexander S and Thakar A R (2001). The Hierarchical Triangular Mesh. In Banday A J, Zaroubi S and Barteimann M (eds.). *Mining the Sky*. Springer. Pp631-637

Mosaic. https://databrickslabs.github.io/mosaic/. Accessed 13 Dec. 2024

Niemeyer G (2008). Geohash.org is public! https://blog.labix.org/2008/02/26/geohashorg-is-public. Accessed 15 Dec. 2024

Samet H (1990). *The design and analysis of spatial data structures*. Addison-Wesley, New York.

Biographies

Sheng Zhou is a Senior Data Scientist at Ordnance Survey. He obtained his PhD in GIS/Urban Planning from Cardiff University and participated in research projects on multiscale spatial database and multi-agent system for active maps at Glamorgan and Cardiff universities. His research interests include machine learning algorithms and applications on spatial data, computational geometry, spatial databases and map generalisation.

Jonathan Simmons is the Head of Data Science and Management at Ordnance Survey.