Geometry on the Sphere: Google's S2 Library

Octavian Procopiuc
tavi@google.com

Library Overview

- C++ library, open source (Apache License 2)
- Designed and written by Eric Veach

- Basic representations of lat/lng points and 3d vectors
- Shapes on the unit sphere:
  - caps,
  - lat/lng rectangles,
  - polygons, polygonal lines.

- Hierarchical decomposition of the sphere into "cells".
- Ability to approximate regions using cells.

S2 Cell Hierarchy

- Hierarchical division of the sphere.
- Goals:
  - Enough resolution for indexing geographic features
  - Compact representation of each cell
  - Fast methods for querying with arbitrary regions
  - All cells at a given level should have similar area.

- One solution: Quad-tree.

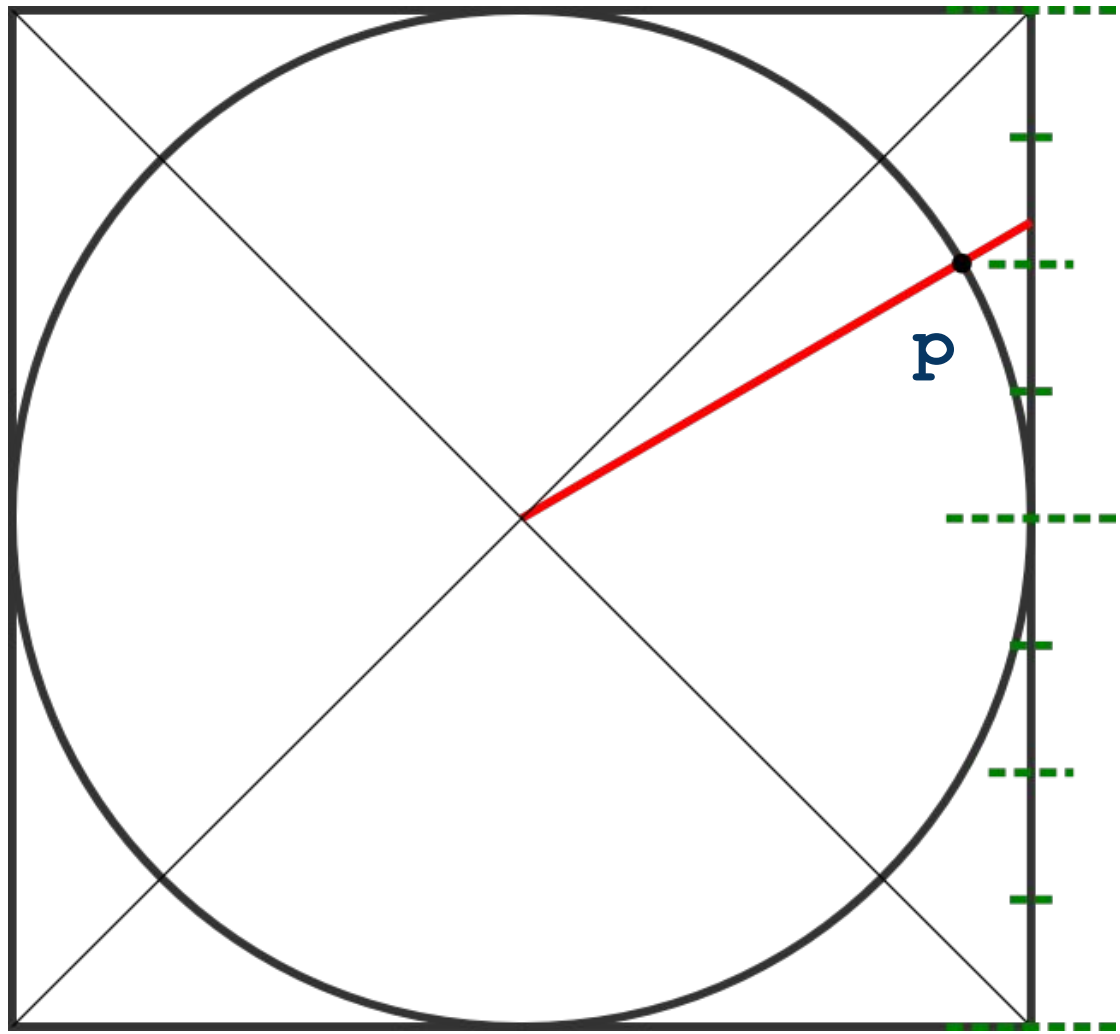
S2 Cell Hierarchy - Construction

Overview:

- Enclose sphere in cube [-1,1] x [-1,1] x [-1,1]
- Project `p` on the cube
- Build a quad-tree on each cube face
- Find quad-tree cell that contains the projection of `p`

Step 1:

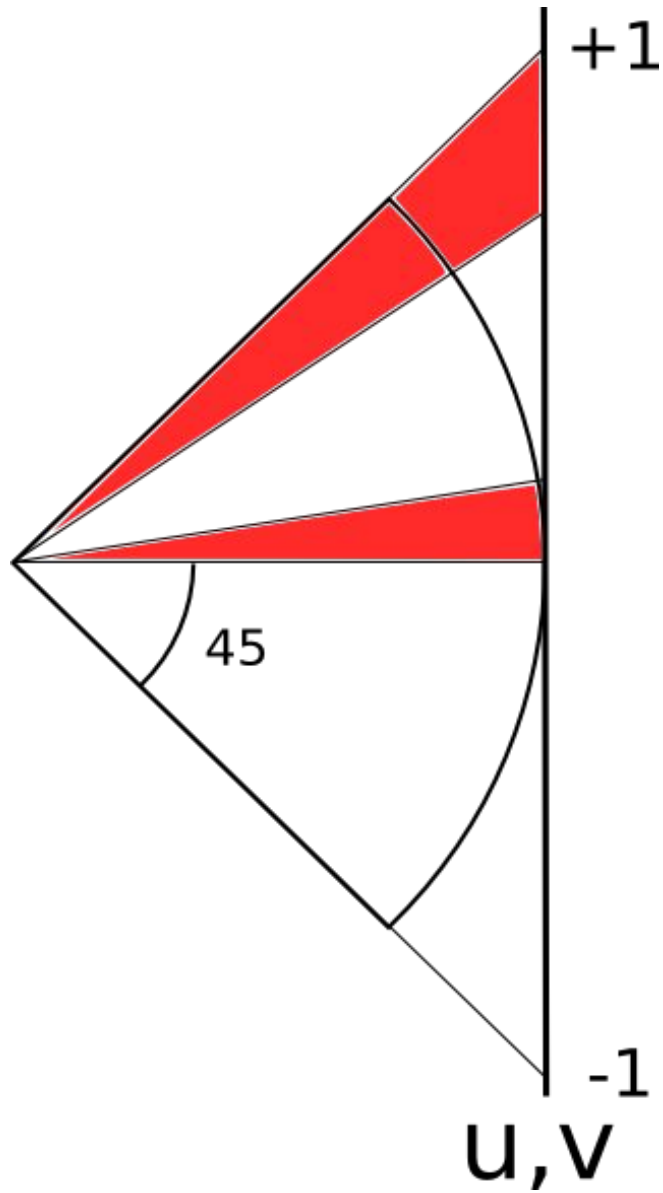
```
p=(lat,lng) => (x,y,z)
```



S2 Cell Hierarchy - Construction

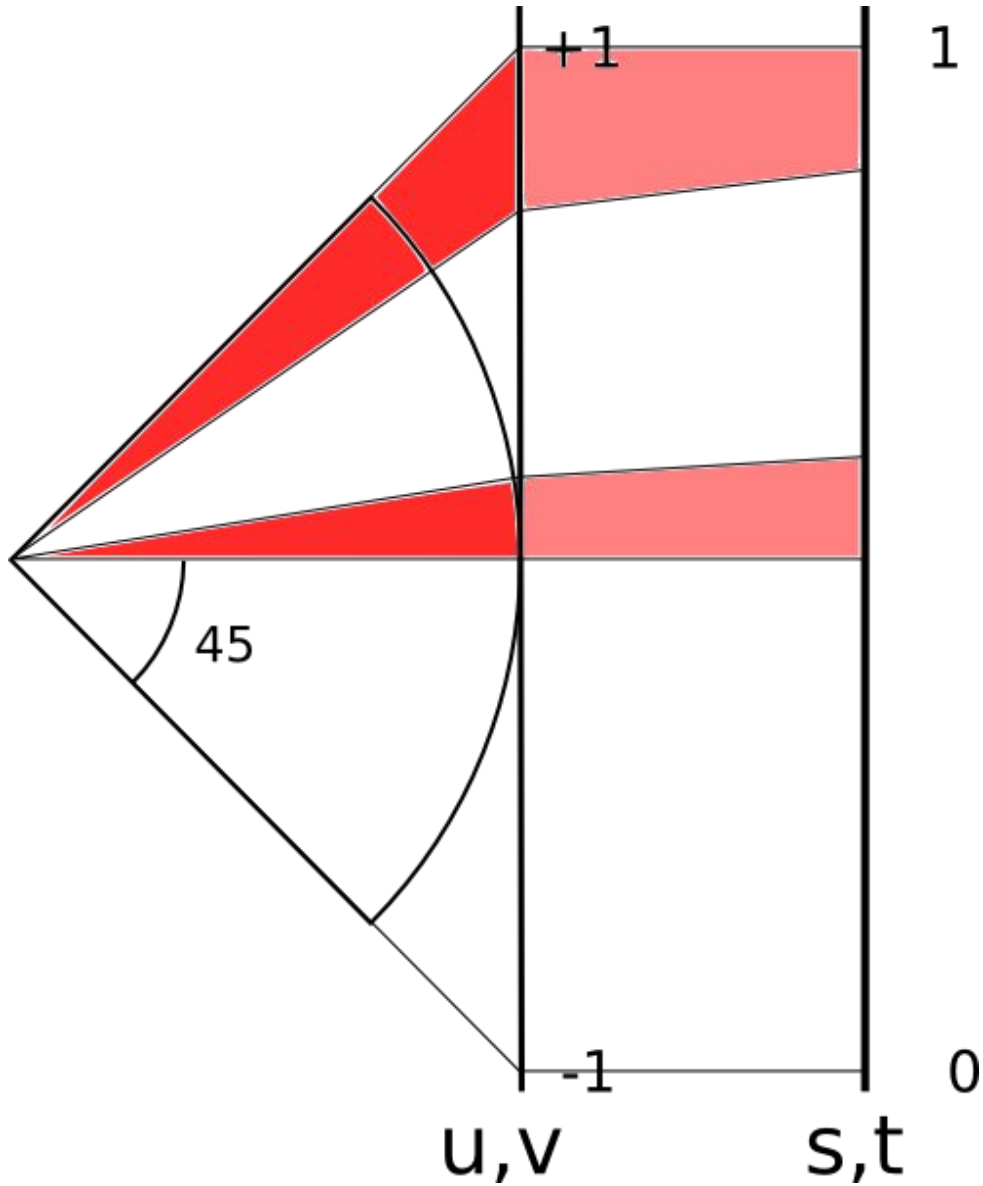
- Step 2:
  `(x,y,z) => (face,u,v)`

- Problem: same-area cells on the cube have different sizes on the sphere. Ratio of highest to lowest area: 5.2



S2 Cell Hierarchy - Construction

- Solution: non-linear transform
  `(face,u,v) => (face,s,t)`
- s,t in [0,1]



An Aside - Projection Trade-offs

- Choices for the `(u,v) => (s,t)` projection.
  - Linear: fast, but cell sizes vary widely
  - Tangent: uses atan() to make sizes more uniform; slow
  - **Quadratic**: much faster and almost as good as tangent.

	Area Ratio	Cell -> Point	Point -> Cell
Linear	5.20	0.087	0.085
Tangent	1.41	0.299	0.258
Quadratic	2.08	0.096	0.108

microseconds

S2 Cell Hierarchy - Construction

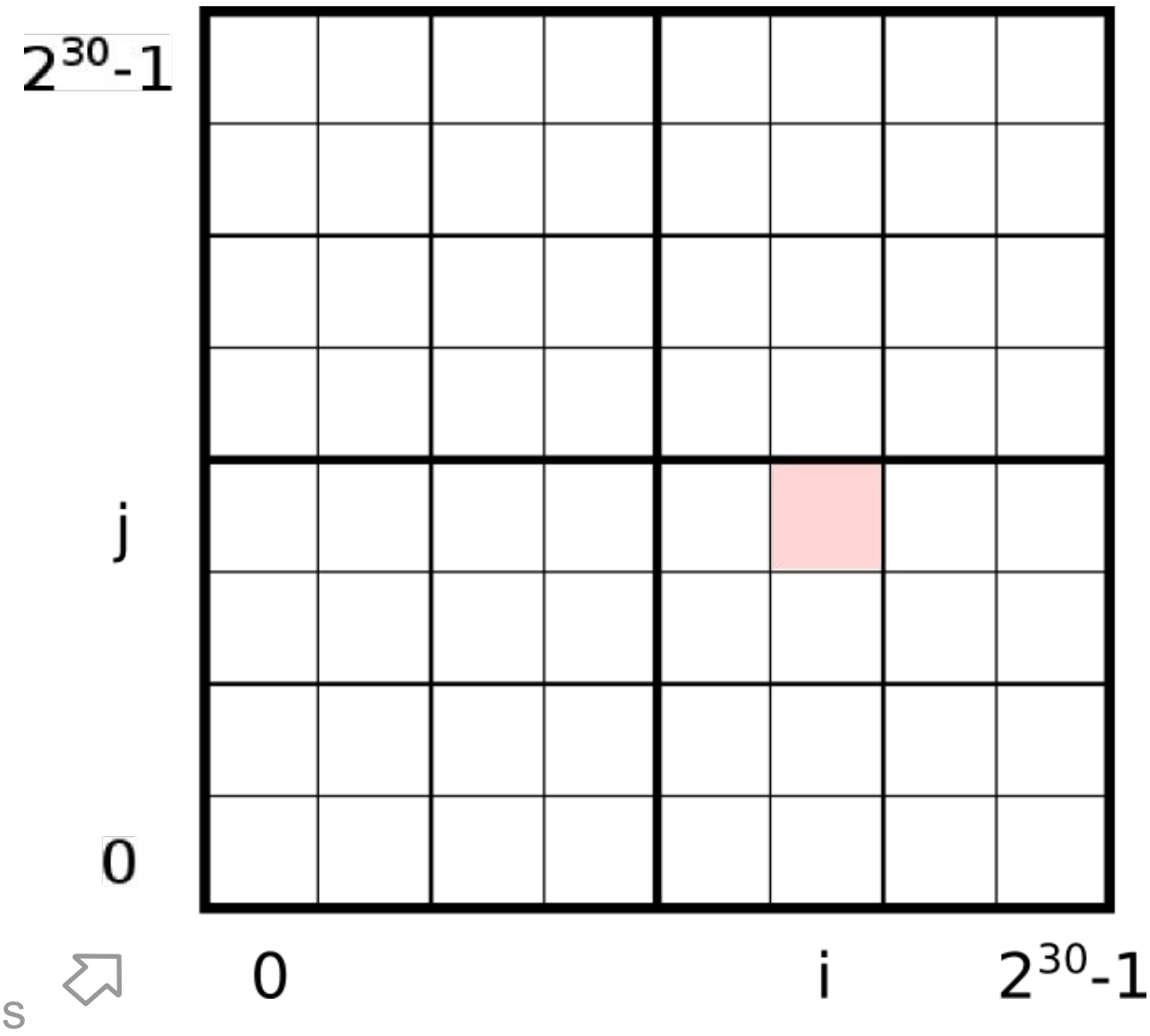
Story so far:

- `(lat,lng)`
- `(x,y,z)`
- `(face,u,v)`
- `(face,s,t)`

Discretize `(s,t)`

- `(face,i,j)`

Quad-tree cell: most significant bits of `i` and `j`



S2 Cell Hierarchy - Construction

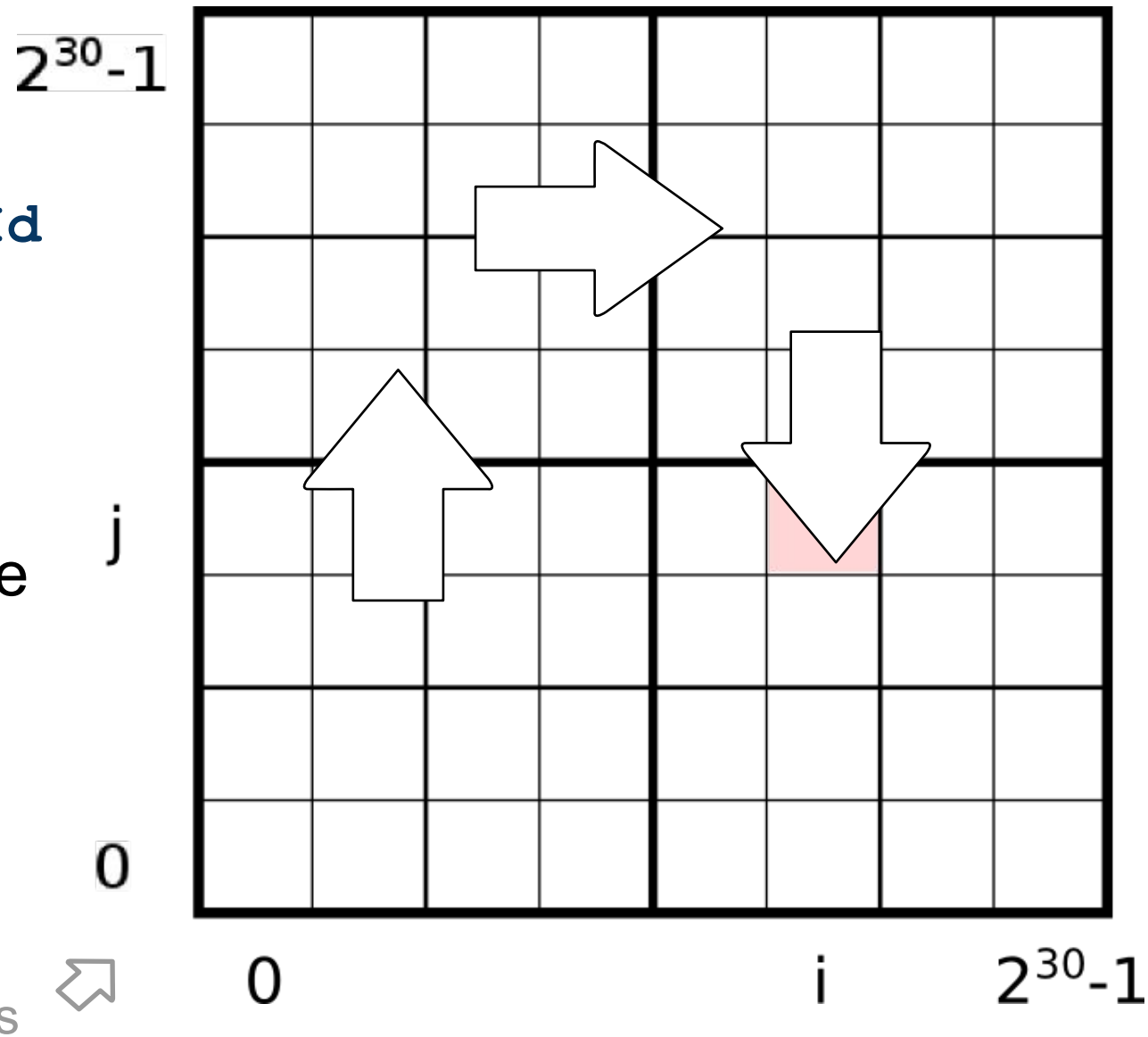
Last step:

`(face,i,j)=> S2CellId`

[ID is a 64-bit integer]

Enumerate cells along a Hilbert space-filling curve

- fast to encode and decode (bit flipping)
- preserves spatial locality



S2 Cell Hierarchy - Construction

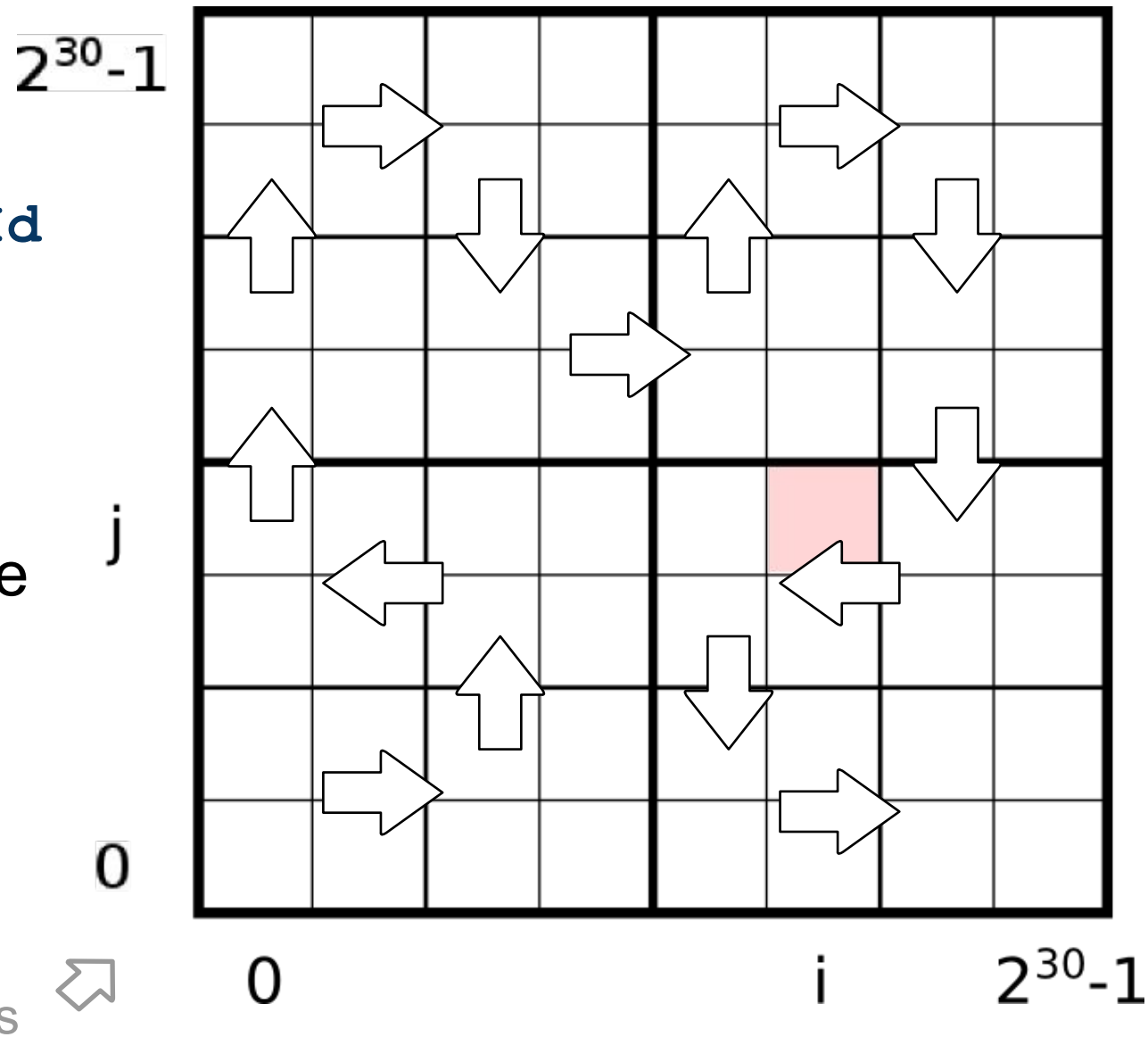
Last step:

```
(face,i,j)=> S2CellId
```

[ID is a 64-bit integer]

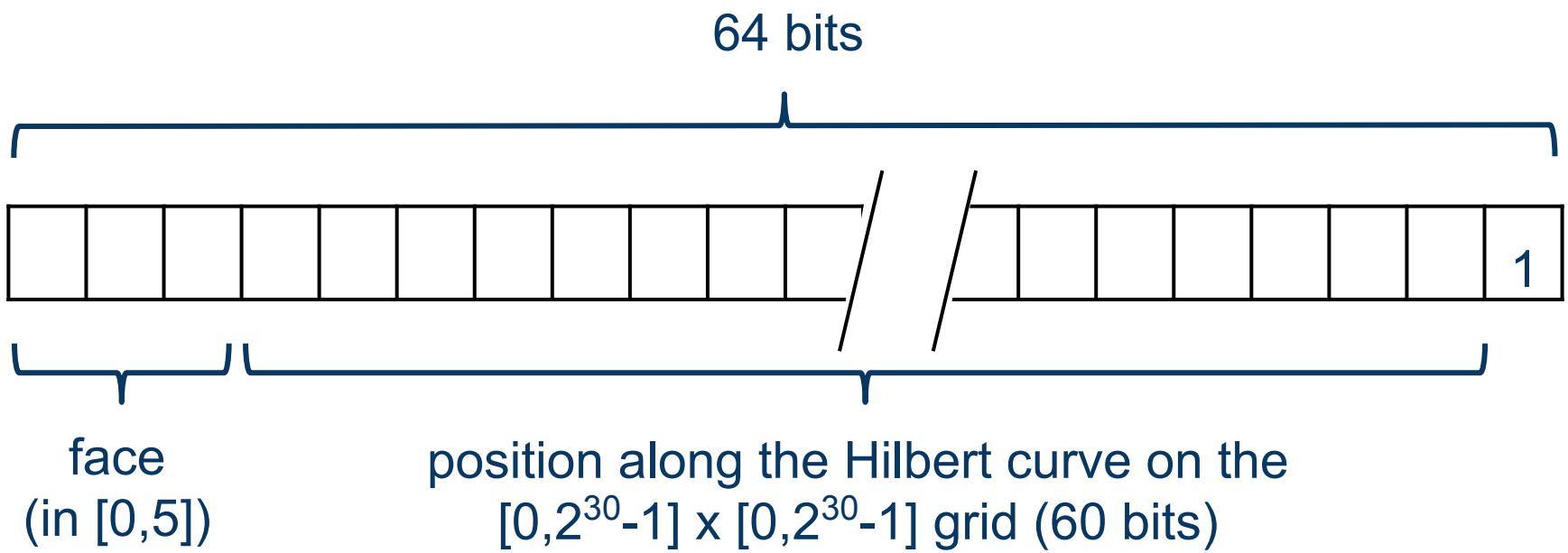
Enumerate cells along a Hilbert space-filling curve

- fast to encode and decode (bit flipping)
- preserves spatial locality



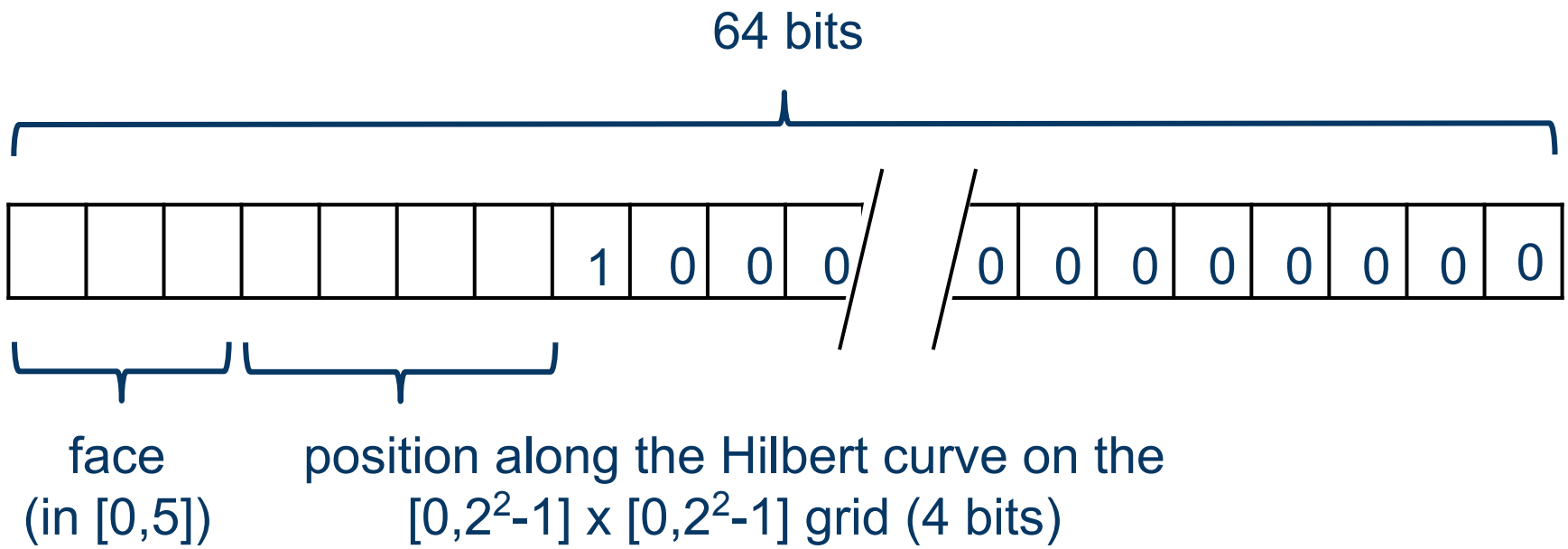
S2 Cell Hierarchy - Construction

S2 Cell ID of a **leaf** cell (level 30):



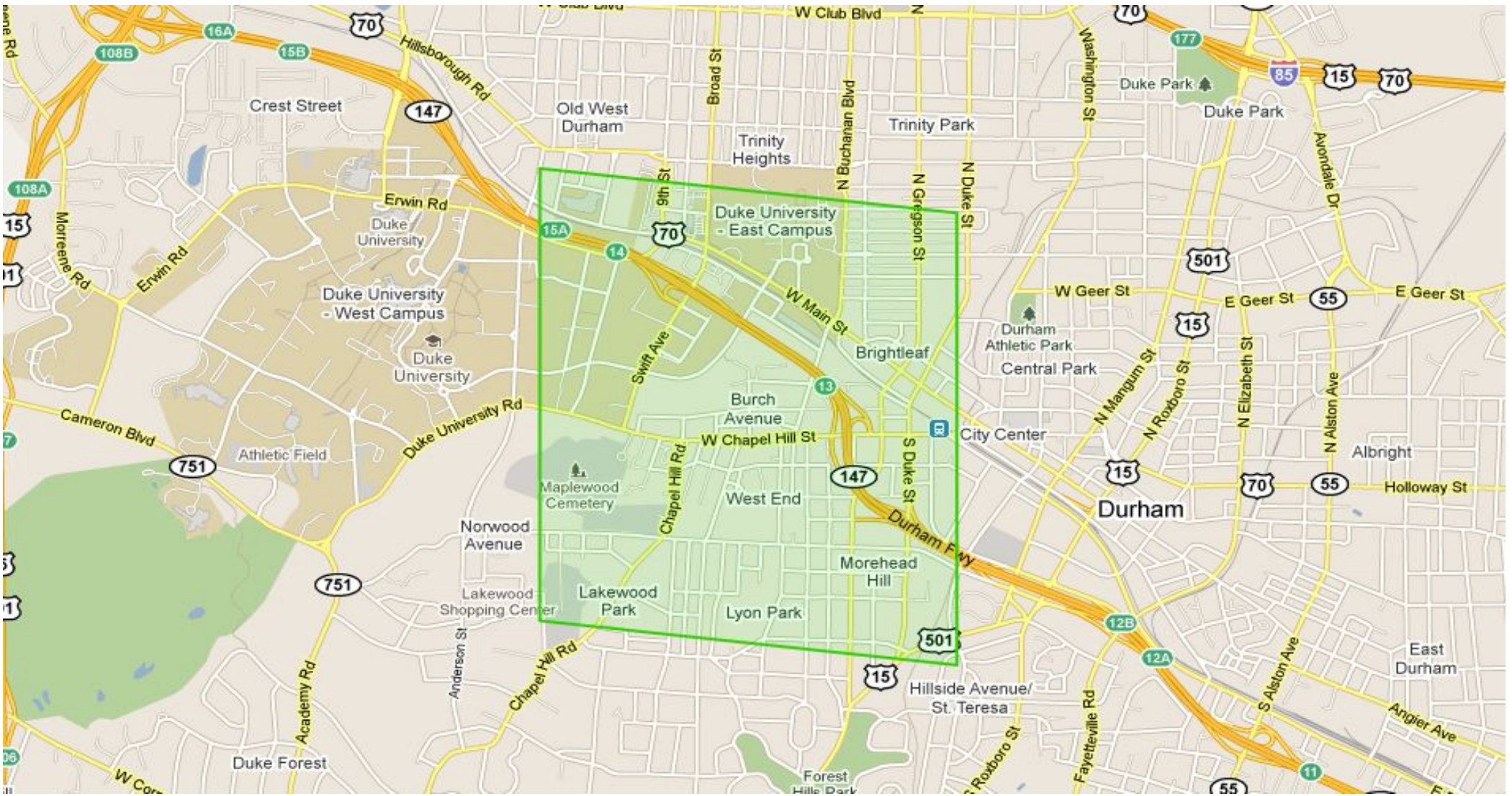
S2 Cell Hierarchy - Construction

S2 Cell ID of a **level-2** cell:



One S2 Cell

Id: 0x89ace41000000000 (0b1000100110101100111001000001000...), Level: 12



S2 Cells - Stats

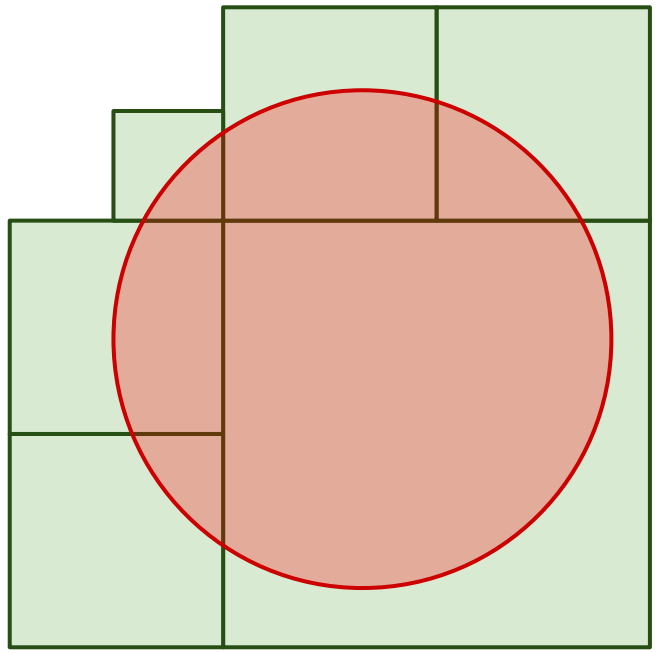
Level	Min Area	Max Area
0	85,011,012 km^2	85,011,012 km^2
1	21,252,753 km^2	21,252,753 km^2
12	3.31 km^2	6.38 km^2
30	0.48 cm^2	0.93 cm^2

smallest cell

Every cm^2 on Earth can be represented using a 64-bit integer.

Approximating Regions Using S2 Cells

Given a region, find a (small) set of cells that cover it.



Parameters:

- max number of cells
- max cell level
- min cell level

Max # cells	Median ratio (covering area / region area)	Worst ratio
4	3.31	15.83
8	1.98	4.03
20	1.42	1.94
100	1.11	1.19

⇦ default

What Else Is In the Library

- CCW: Given three points on the sphere, are they counter-clockwise?
  - Multiple implementations, with various tradeoffs.
- Polygons
  - Containment, intersection, union, difference, simplification, centroid computation, etc.
  - Serialization.
- Polygonal lines, Spherical caps.
- Extensive tests and micro-benchmarks.

Other Similar Libraries

- Hierarchical Triangular Mesh (http://skyserver.org/HTM). The lat/lng <-> triangle id conversion is ~100 slower than the lat/lng <-> s2 cell id conversion.
- HEALPix (http://healpix.jpl.nasa.gov). Cell boundaries are not geodesics; structure is more complicated.
- COBE Quadrilateralized Spherical Cube (http://lambda.gsfc.nasa.gov/product/cobe/skymap_info_new.cfm). Similar decomposition of sphere. But does not use space-filling curve, edges are not geodesics, and projection is more complicated.

The Code

http://code.google.com/p/s2-geometry-library/