Business Requirement

Local Authorities to survey Footways for:

1. Pavement Condition Index to determine quality of Footpath

2. Potential “Trips and Falls”
OSi Prime 2

- National Coverage

- Multitude of ground feature classes

- Ground feature objects in point, line and polygon formats
Why Open Source?

- Functions not available - Symmetrical diff, buffers (dissolved/non-dissolved)
- Multiple separate steps
- Speed - **8 hours** per Local Authority

- Parallelsql ran on PostgreSQL
- Simplification of algorithm
- Availability of spatial functions available
- **2 mins for Dún Laoghaire Rathdown CoCo**
- **50 mins for National dataset**
Prime 2 – Footways

Footways generated from Prime 2 data indirectly

• Buffered Road Centrelines
  • Combination of WAY_POLY and WAY_LINE dataset
  • WAY_POLY used to calculate road widths

• Average Width used
  • All WAY_LINE segments buffered by distance based on average width the individual segment
  • E.g. if average width of WAY_POLY segment is 10.8m, buffer corresponding WAY_LINE segment by 5.4m.
Prime 2 – Footways

1. Round Buffer (Dissolved/Non-dissolved)
2. Flat Buffers (Dissolved/Non-dissolved)
3. Symmetrical Difference (cul de sacs)
4. Calculate all neighbouring round buffers and dissolve (T junction widths)
5. Dissolve no. 4 and outline is used
DROP TABLE IF EXISTS footways_parts;

CREATE UNLOGGED TABLE footways_parts
(
    id serial,
    footways_edge_id integer NOT NULL,
    local_authority_id integer NOT NULL,
    geom2157 geometry,
    CONSTRAINT footways_parts_pkey PRIMARY KEY (id),
    CONSTRAINT enforce_srid_geom2157 CHECK (st_srid(geom2157) = 2157)
) WITH (OIDS=FALSE);

ALTER TABLE footways_parts
    OWNER TO postgres;

CREATE INDEX idx_footways_parts_footways_edge_id
    ON footways_parts
    USING btree
    (footways_edge_id);

-- Index: idx_footways_parts_local_authority_id
-- DROP INDEX idx_footways_parts_local_authority_id;

CREATE INDEX idx_footways_parts_local_authority_id
    ON footways_parts
    USING btree
    (local_authority_id);

-- Index: sidx_footways_parts
-- DROP INDEX sidx_footways_parts;

CREATE INDEX sidx_footways_parts
    ON footways_parts
    USING gist
    (geom2157);
TRUNCATE TABLE footways_parts;

-- takes 1:11 secs
SELECT parallelsql
('dbname=footways user=postgres password='HuufDorf13!''', --connection string
 'round_buffers_for_intersections', --table
 'rbi.footways_edge_id', --variable to partition by processes
 'WITH footways_p AS (SELECT rbi.footways_edge_id,
 rbi.local_authority_id,
 CASE WHEN de.geom2157 IS NULL THEN rbi.geom2157 ELSE ST_LineMerge(ST_Difference(rbi.geom2157,
 ST_Buffer(de.geom2157, 0.00001, ''endcap=round join=round''))) END as geom2157
 FROM round_buffers_intersections rbi LEFT OUTER JOIN dead_ends de ON rbi.footways_edge_id =
 de.footways_edge_id
 WHERE 1=1
 )
 SELECT footways_edge_id,
 local_authority_id,
 (ST_Dump(geom2157)).geom AS geom2157
 FROM footways_p
 );', --the statement as string to be executed in parallel
 'footways_parts (footways_edge_id, local_authority_id, geom2157)', --result table, has to be created
 first
 'rbi', --table alias used for split column
 8, --number of cores
 '1=1', --replace string in the query
 10000 --block size
);

ANALYZE footways_parts;