Efficient K-nearest neighbour search in PostgreSQL

Oleg Bartunov, Teodor Sigaev

Knn-search: The Problem

```
knn=# select id, date, event from events
      order by date <-> '1957-10-04'::date asc limit 10;
   id   |    date    |                                 event
--------+------------+-----------------------------------------------------------------
  58137 | 1957-10-04 | U.S.S.R. launches Sputnik I, 1st artificial Earth satellite
  58136 | 1957-10-04 | "Leave It to Beaver," debuts on CBS
 117062 | 1957-10-04 | Gregory T Linteris, Demarest, New Jersey, astronaut, sk: STS 83
 117061 | 1957-10-04 | Christina Smith, born in Miami, Florida, playmate, Mar, 1978
 102670 | 1957-10-05 | Larry Saumell, jockey
  31456 | 1957-10-03 | Willy Brandt elected mayor of West Berlin
  58291 | 1957-10-05 | 12th Ryder Cup: Britain-Ireland, 7 -4 at Lindrick GC, England
  58290 | 1957-10-05 | 11th NHL All-Star Game: All-Stars beat Montreal 5-3 at Montreal
  58292 | 1957-10-05 | Yugoslav dissident Milovan Djilos sentenced to 7 years
 102669 | 1957-10-05 | Jeanne Evert, tennis player, Chris' sister
(10 rows)

Time: 115.548 ms
```

- Very inefficient:
  - Full table scan, btree index on date won't help.
  - Sort full table

Knn-search: Existing solutions

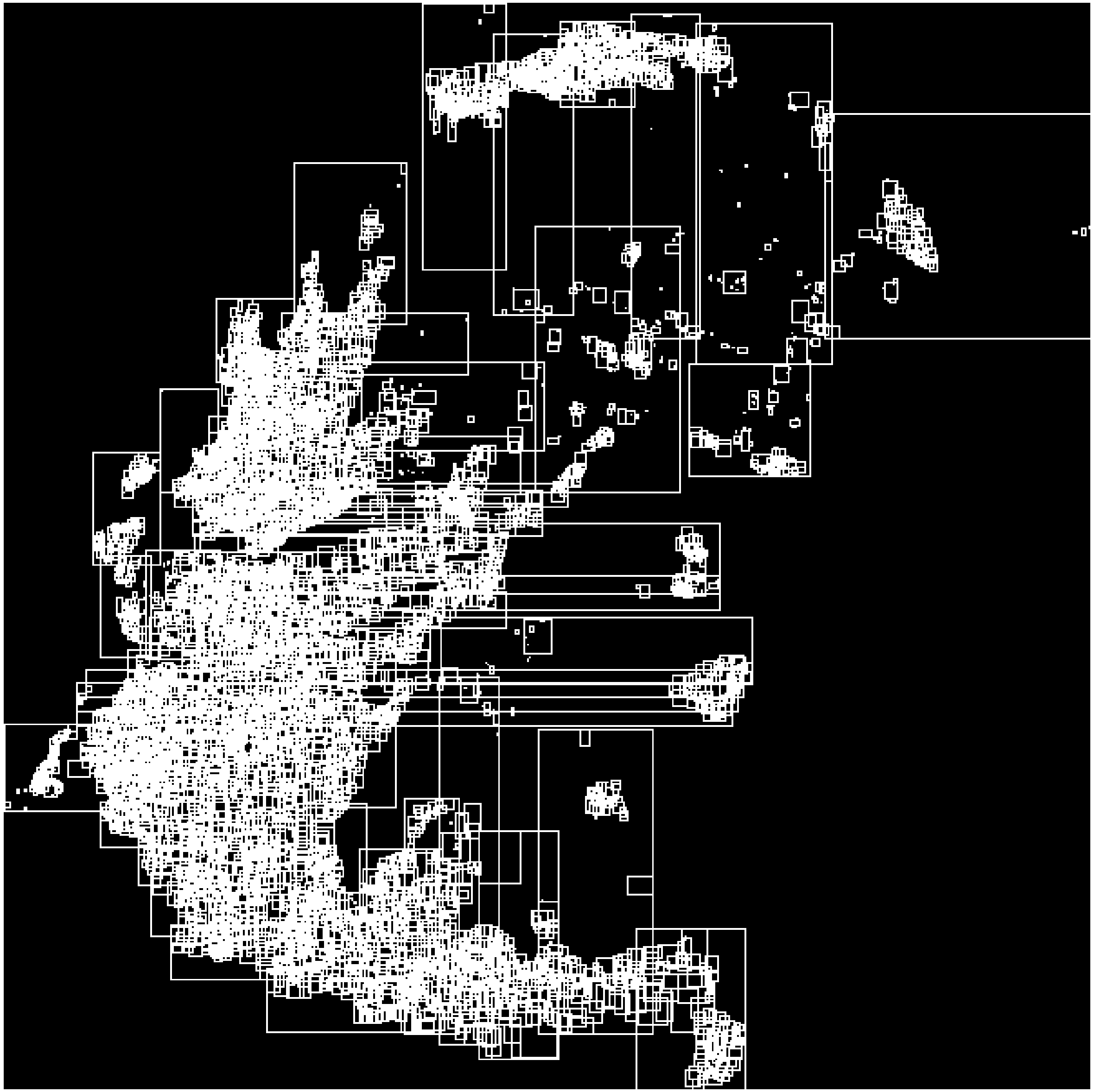
- Traditional way to speedup query
  - Use indexes - very inefficient (no search query !)
    - Scan full index
    - Full table scan, but in random order !
    - Sort full table
    - **Better not to use index at all !**
  - Constrain data space (range search)
    - Incremental search → to many queries
    - Need to know in advance the size of neighbourhood, how ? 1 Km is ok for Paris, but too small for Siberia
    - Maintain 'density map' ?

Knn-search: What do we want !

- We want to avoid full table scan – read only **right** tuples
  - So, we need index
- We want to avoid sorting – read **right** tuples in **right** order
  - So, we need special strategy to traverse index
- We want to support tuples visibility
  - So, we should be able to resume index traverse

R-tree index

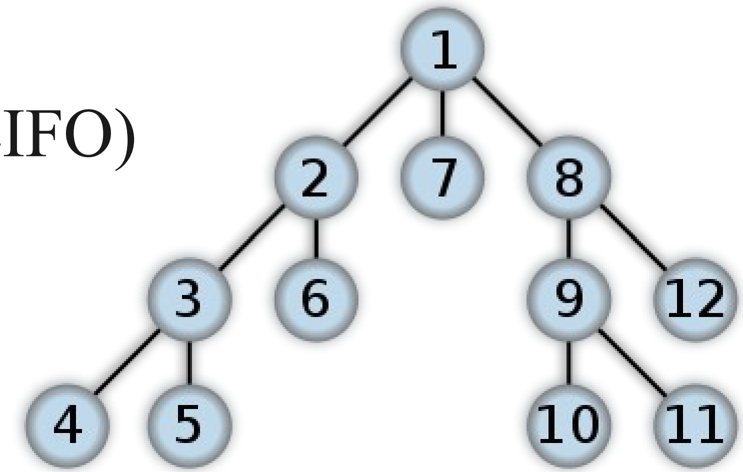
- Visualization of R-tree index using Gevel
- Greece
  (data from rtreeportal.org)



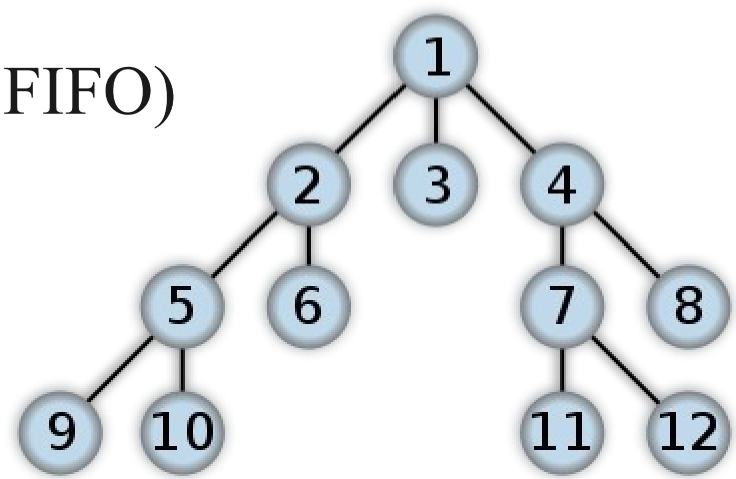
Knn-search: Index traverse

- Depth First Search (stack, LIFO)

  R-tree search



- Breadth First Search (queue, FIFO)



- Both strategies are not good for us – full index scan

Knn-search: Index traverse

- Best First Search (PQ, priority queue). Maintain order of items in PQ according their distance from given point
  - Distance to MBR (rectangle for Rtree) for internal pages – minimum distance of all items in that MBR
  - Distance = 0 for MBR with given point
  - Distance to point for leaf pages
- Each time we extract point from PQ we output it – it is next closest point ! If we extract rectangle, we expand it by pushing their children (rectangles and points) into the queue.
- We traverse index by visiting only interesting nodes !

Knn-search: Performance

- SEQ (no index) – base performance
  - Sequentually read full table + Sort full table (can be very bad, sort_mem !)
- DFS – very bad !
  - Full index scan + Random read full table + Sort full table
- BFS – the best for small k !
  - Partial index scan + Random read k-records
    T(index scan) ~ Height of Search tree ~ log(n)
  - Performance win BFS/SEQ ~ Nrelpages/k, for small k.
    The more rows, the more benefit !
  - Can still win even for k=n (for large tables) - no sort !

Knn-search: What do we want !

- + We want to avoid full table scan – read only **right** tuples
  - So, we need index
- + We want to avoid sorting – read **right** tuples in **right** order
  - So, we need special strategy to traverse index
- + We want to support tuples visibility
  - So, we should be able to resume index traverse
- We want to support many data types
  - So, we need to modify GiST

Knn-search: modify GiST

- GiST – Generalized Search Tree, provides
  - API to build custom disk-based search trees (any tree, where key of internal page is a Union of keys on children pages)
  - Recovery and Concurrency
  - Data type and query extendability
- GiST is widely used in GIS (PostGIS), text search, astro,bio,...
- Current strategy of search tree traverse is DFS
  - Change to BFS (Best First Search) strategy
  - Retain API compatibility

GiST: Technical details

Depth First Search

```
push Stack, Root;
While Stack {
   If p is heap {
             output p;
   else {
             children = get_children(p);
             push Stack, children;
        }
}
```

Best First Search

```
push PQ, Root;
While PQ {
   If p is heap {
             output p;
   else {
             Children = get_children(p);
             push PQ, children;
        }
}
```

- For non-knn search all distances are zero, so PQ => Stack and BFS => DFS
- We can use only one strategy (BFS) for both – normal search and knn-search !

Knn-search: What do we want !

- + We want to avoid full table scan – read only <right> tuples
  - So, we need index
- + We want to avoid sorting – read <right> tuples in <right> order
  - So, we need special strategy to traverse index
- + We want to support tuples visibility
  - So, we should be able to resume index traverse
- + We want to support many data types
  - So, we need to modify GiST

Knn-search: syntax

- Knn-query uses ORDER BY clause

```
SELECT … FROM … WHERE …
ORDER BY p <-> '(0.,0.)'::point
LIMIT k;
```

<-> - distance operator, should be provided for data type

Knn-search: Examples

- Synthetic data – 1,000,000 randomly distributed points

```
create table qq ( id serial, p point, s int4);

insert into qq (p,s)  select point( p.lat, p.long),
(random()*1000)::int
from ( select  (0.5-random())*180 as lat, random()*360 as
long
       from ( select generate_series(1,1000000) ) as t
) as p;
create index qq_p_s_idx on qq using gist(p);
analyze qq;
```

- Query – find k-closest points to (0,0)

```
set enable_indexscan=on|off;
explain (analyze on, buffers on)
  select * from qq order by (p <-> '(0,0)') asc limit 10;
```

Knn-search: Examples

- postgresql.conf:

  shared_buffers = 512MB #32MB
  work_mem = 32MB #1MB
  maintenance_work_mem = 256MB #16MB
  checkpoint_segments = 16
  effective_cache_size = 1GB #128MB

- Index statistics (n=1000,000)

```
Number of levels:          3
Number of pages:           8787
Number of leaf pages:      8704
Number of tuples:          1008786
Number of invalid tuples:  0
Number of leaf tuples:     1000000
Total size of tuples:      44492028 bytes
Total size of leaf tuples: 44104448 bytes
Total size of index:       71983104 bytes
```

Knn-search: Examples

k=1, n=1,000,000

```
 Limit  (cost=0.00..0.08 rows=1 width=24) (actual time=0.104..0.104
rows=1 loops=1)
   Buffers: shared hit=4
   ->  Index Scan using qq_p_idx on qq  (cost=0.00..82060.60 rows=1000000
width=24) (actual time=0.104..0.104 rows=1 loops=1)
         Sort Cond: (p <-> '(0,0)'::point)
         Buffers: shared hit=4
 Total runtime: 0.117 ms
```

4000 times faster !

```
--------------------------
Limit  (cost=24853.00..24853.00 rows=1 width=24) (actual time=469.129..469.130
rows=1 loops=1)
   Buffers: shared hit=7353
   ->  Sort  (cost=24853.00..27353.00 rows=1000000 width=24) (actual
time=469.128..469.128 rows=1 loops=1)
         Sort Key: ((p <-> '(0,0)'::point))
         Sort Method:  top-N heapsort  Memory: 25kB
         Buffers: shared hit=7353
         ->  Seq Scan on qq  (cost=0.00..19853.00 rows=1000000 width=24)
(actual time=0.007..241.539 rows=1000000 loops=1)
               Buffers: shared hit=7353
 Total runtime: 469.150 ms
```

Knn-search: Examples

N=1,000,000

k	:hit	:knn	: seq	:sortmem (seq)
1	:4	:0.117	:469.150	: 25
10	:17	:0.289	:471.735	: 25
100	:118	:0.872	:468.244	: 32
1000	:1099	:7.107	:473.840	: 127
10000	:10234	:31.629	:525.557	: 1550
100000	:101159	:321.182	:994.925	: 13957

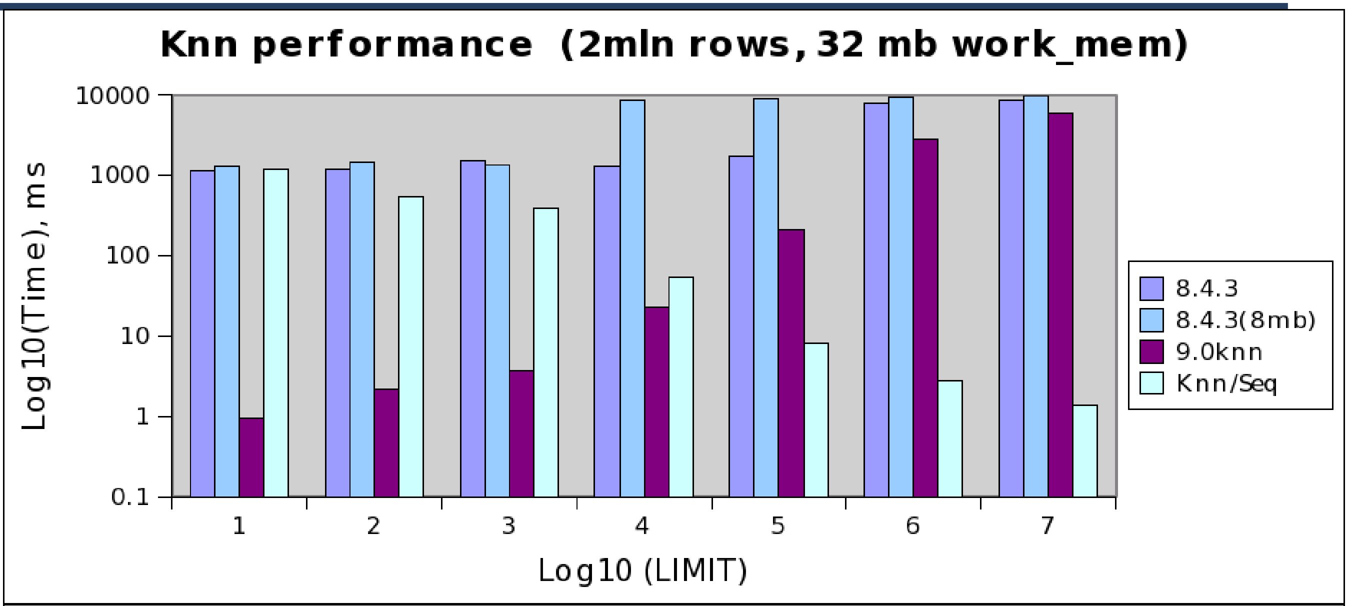
Knn-search: Examples

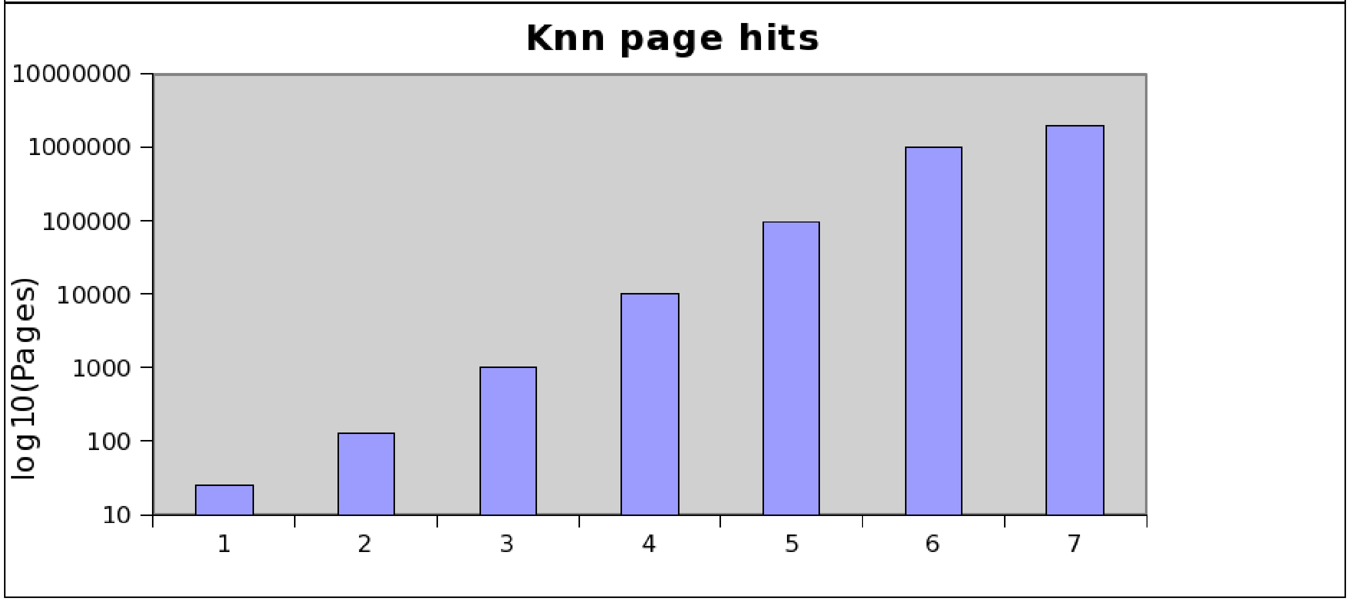
`n=10,000`

K	:hit	:knn	: seq	
1	:3	:0.117	:6.072	
10	:13	:0.247	:5.014	
100	:103	:0.295	:6.381	
1000	:996	:1.605	:8.670	
10000	:9916	:16.487	:14.706	-> knn lose if k=n, n is small

Knn-search: Examples

- Real data
  2 mln points
  US, geonames





Knn-search: Examples

- Query: find 10 closest points in US to the point (5,5) with 'mars' in names - create composite index:

```
create index pt_fts_idx on geo
         using gist(point, to_tsvector('english',asciiname));

=# explain (analyze on, buffers on)
select asciiname,point, (point <->'5.0,5.0'::point) as dist from geo
where to_tsvector('english', asciiname) @@ to_tsquery('english','mars')
order by dist asc limit 10;
                                                    QUERY PLAN
-----------------------------------------------------------------------------
 Limit  (cost=0.00..33.55 rows=10 width=35) (actual time=0.452..0.597 rows=10 loops=1)
   Buffers: shared hit=56
   ->  Index Scan using pt_fts_idx on geo  (cost=0.00..34313.91 rows=10227 width=35)
(actual time=0.452..0.592 rows=10 loops=1)
         Index Cond: (to_tsvector('english'::regconfig, (asciiname)::text) @@
'''mar'''::tsquery)
         Sort Cond: (point <-> '(5,5)'::point)
         Buffers: shared hit=56
 Total runtime: 0.629 ms
```

Knn-search: The Problem

```
knn=# select id, date, event from events
      order by date <-> '1957-10-04'::date asc limit 10;
   id   |    date    |                                 event
--------+------------+-----------------------------------------------------------------
  58137 | 1957-10-04 | U.S.S.R. launches Sputnik I, 1st artificial Earth satellite
  58136 | 1957-10-04 | "Leave It to Beaver," debuts on CBS
 117062 | 1957-10-04 | Gregory T Linteris, Demarest, New Jersey, astronaut, sk: STS 83
 117061 | 1957-10-04 | Christina Smith, born in Miami, Florida, playmate, Mar, 1978
 102670 | 1957-10-05 | Larry Saumell, jockey
  31456 | 1957-10-03 | Willy Brandt elected mayor of West Berlin
  58291 | 1957-10-05 | 12th Ryder Cup: Britain-Ireland, 7 -4 at Lindrick GC, England
  58290 | 1957-10-05 | 11th NHL All-Star Game: All-Stars beat Montreal 5-3 at Montreal
  58292 | 1957-10-05 | Yugoslav dissident Milovan Djilos sentenced to 7 years
 102669 | 1957-10-05 | Jeanne Evert, tennis player, Chris' sister
(10 rows)

Time: 115.548 ms
```

- Very inefficient:
  - Full table scan, btree index on date won't help.
  - Sort full table

Knn-search: The Problem

contrib/btree_gist

```
knn=# select id, date, event from events
      order by date <-> '1957-10-04'::date asc limit 10;
   id   |    date    |                                 event
--------+------------+-----------------------------------------------------------------
  58137 | 1957-10-04 | U.S.S.R. launches Sputnik I, 1st artificial Earth satellite
  58136 | 1957-10-04 | "Leave It to Beaver," debuts on CBS
 117062 | 1957-10-04 | Gregory T Linteris, Demarest, New Jersey, astronaut, sk: STS 83
 117061 | 1957-10-04 | Christina Smith, born in Miami, Florida, playmate, Mar, 1978
 102670 | 1957-10-05 | Larry Saumell, jockey
  31456 | 1957-10-03 | Willy Brandt elected mayor of West Berlin
  58291 | 1957-10-05 | 12th Ryder Cup: Britain-Ireland, 7 -4 at Lindrick GC, England
  58290 | 1957-10-05 | 11th NHL All-Star Game: All-Stars beat Montreal 5-3 at Montreal
  58292 | 1957-10-05 | Yugoslav dissident Milovan Djilos sentenced to 7 years
 102669 | 1957-10-05 | Jeanne Evert, tennis player, Chris' sister
(10 rows)

Time: 0.590 ms
```

- Very inefficient:
  - 8 index pages read + 10 tuples read, no sorting
  - No sorting
  - 200 times faster !

Knn-search: Status

Committed to 9.1