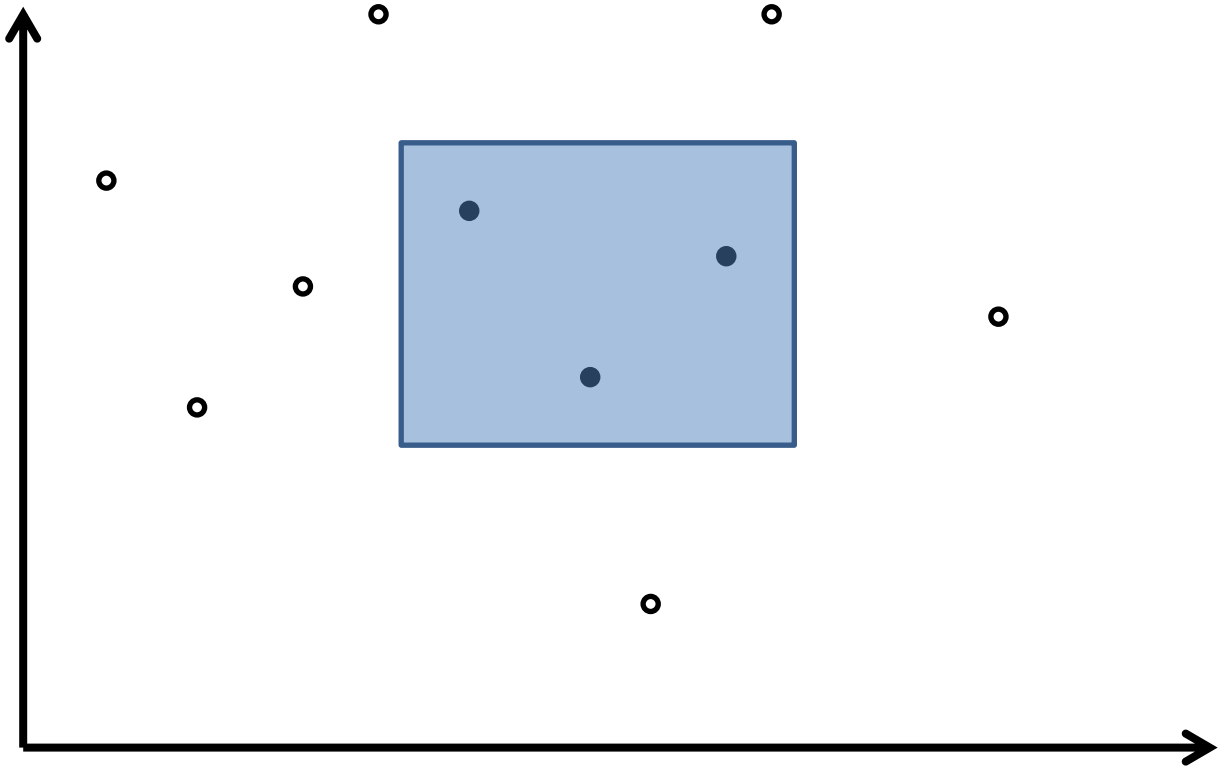
香港中文大學
The Chinese University of Hong Kong

Faculty of Engineering
Department of Computer Science and Engineering
Final Year Project

Range Count in External Memory

Student : Lo Yu Ho
Supervisor : Prof. Tao Yufei
Project ID : TAO1201
Date : 11st December, 2012

Range Count Problem

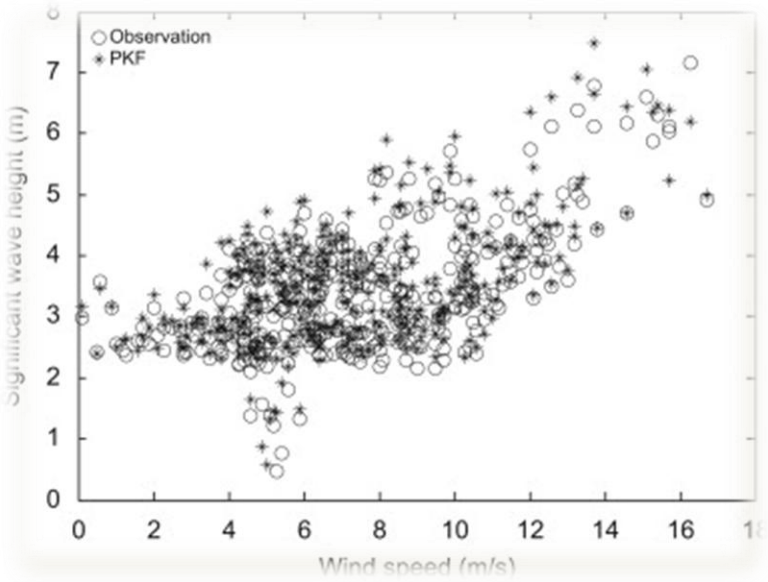


Real Life Applications

- Consider we have a dataset containing the location of trees
- And, EMO (Estates Management Office) is doing green planning
- They want someone to tell them the number of trees in any regions they pointed to **instantly**

Real Life Applications (cont.)

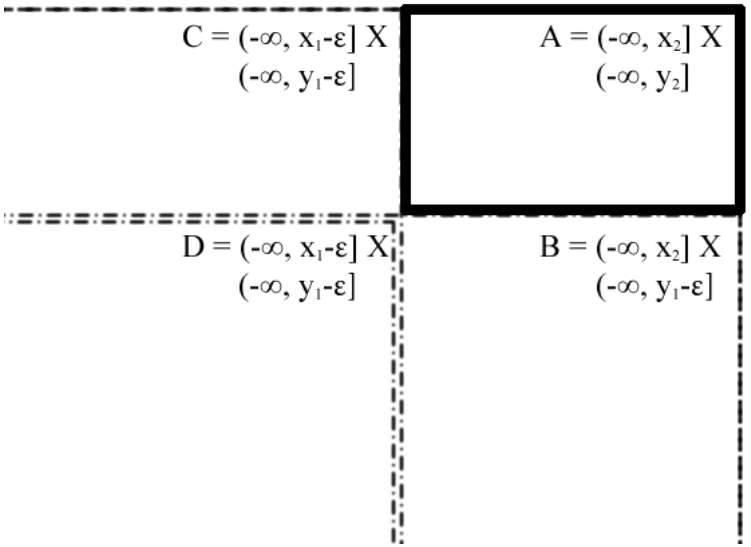
- Each point in a datum in a scatter diagram
- Report the number of data in a rectangular ROI (region of interest)



Our Structure

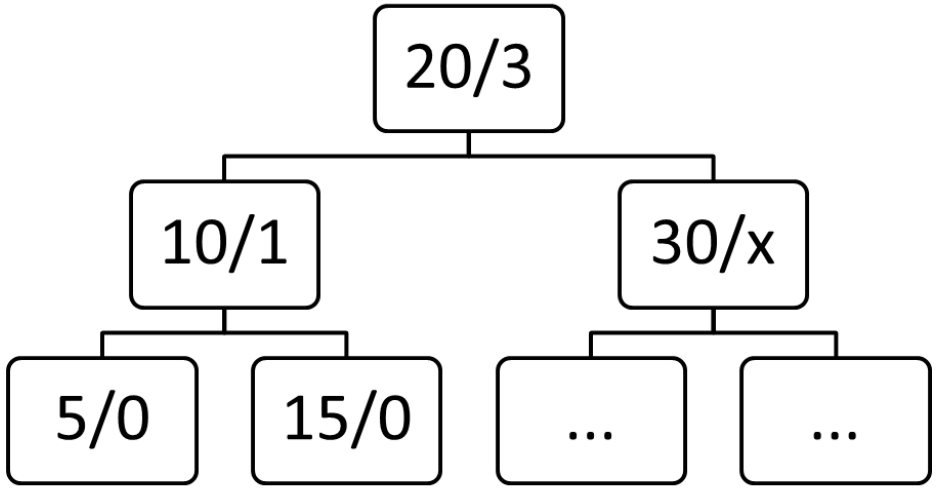
- Space complexity: $O\left(\frac{N}{B}\right)$ Blocks
- Query time complexity: $O(log_B^2 N)$ I/Os (can be further optimized to $O(log_B N)$I/Os)
- Building time complexity: $O\left(\frac{N}{B} log_B N\right)$ I/Os

Reduction of Range Count



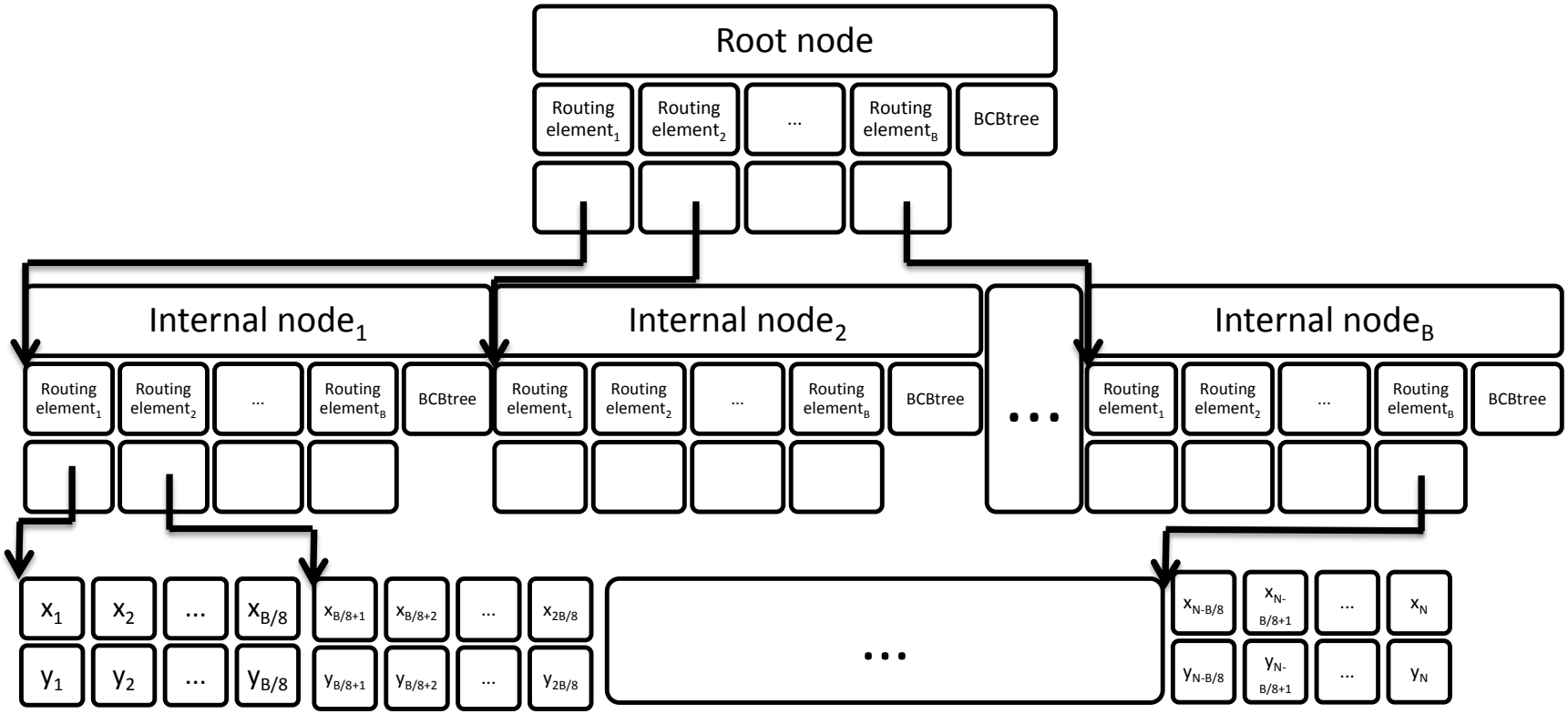
- Q = A - B - C + D, where Q is a 4-sided rectangle as $[x_1, x_2] \times [y_1, y_2]$

Range Count



- Query: 20 → 3
- Query: 12 → 1 + 1
- Query: 31 → 3 + x + …

Range Count (cont.)



- Complexities:
- Building: $O\left(\frac{N}{B} log_B N\right)$ I/Os; query: $O(log_B^2 N)$ I/Os; space:$O\left(\frac{N}{B}\right)$ Blocks
- Query cost can be further optimized to $O(log_B N)$I/Os

Bundled compressed B-tree

- P1: {(16, 1), (25, 2), (29, 3)}
- P2: {(16, 1), (20, 2)}
- P3: {(24, 1), (26, 2), (27, 3), (37, 4)}

- $\{P_1, ..., P_b\}$: a bundle
- Each P_i : a category
- $\ell(k, i)$: a label / rank
- List Δ of triple $(\hat{\delta}(j), i, \delta(k_j, i))$ has length K, where $K = \Sigma_i |P_i|$, and $0 \leq j \leq K$
- Store Δ in fat blocks (each sized 4B) associated with relay sets

Gamma Elias code

- $x \rightarrow 2\lfloor log_2 x \rfloor + 1$ bits

Number	Encoding
1	1
2	010
3	011
⋮	⋮

- Drawback$_1$: not encoding zero or –ve integers
- Drawback$_2$: no terminators in a list of integers

Modified gamma Elias code

Bit pattern	Meaning
00	positive sign
01	zero
10	negative sign
11	terminator

- $x \rightarrow 2\lfloor log_2 x \rfloor + 3$ bits
- Support encoding zero
- Support encoding –ve integers
- Support encoding a list of integers in to a bit stream

Bundled compressed B-tree

- P1: {(16, 1), (25, 2), (29, 3)}
- P2: {(16, 1), (20, 2)}
- P3: {(24, 1), (26, 2), (27, 3), (37, 4)}

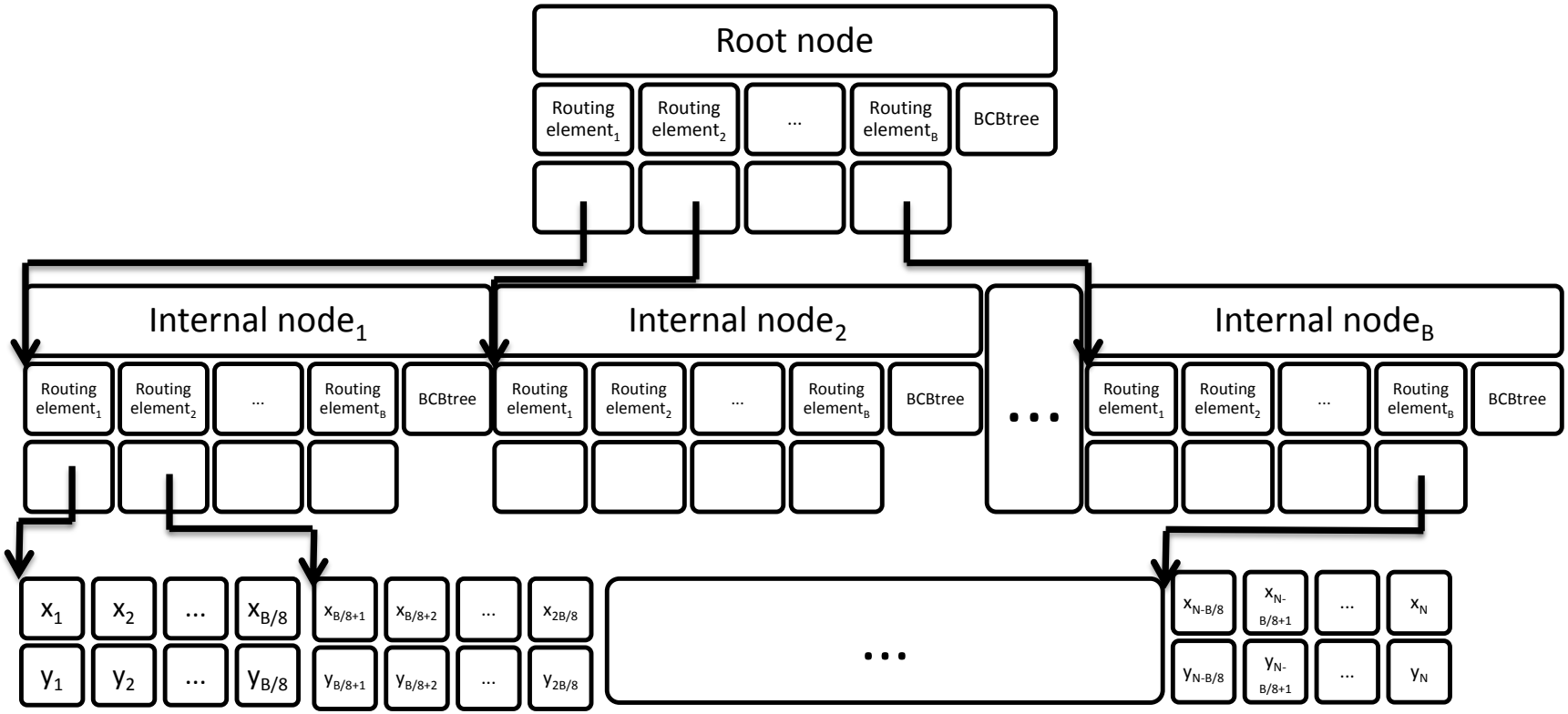
- $\{P_1, ..., P_b\}$: a bundle
- Each P_i : a category
- $\ell(k, i)$: a label / rank
- List Δ of triple $(\hat{\delta}(j), i, \delta(k_j, i))$ has length K, where $K = \Sigma_i |P_i|$, and $0 \leq j \leq K$
- Store Δ in fat blocks (each sized 4B) associated with relay sets

Bundled compressed B-tree (cont.)

- Step 1: read metadata
- Step 2: jump to read B-tree node
- Step 3: go down to leaf node according to B-tree routing path
- Step 4: recover tuples from fat block and relay set
- Complexities:
- Building: $O\left(\frac{N}{B}\right)$ I/Os; query: $O(log_B N)$ I/Os; space:$O\left(\frac{N}{Blog_B N}\right)$Blocks

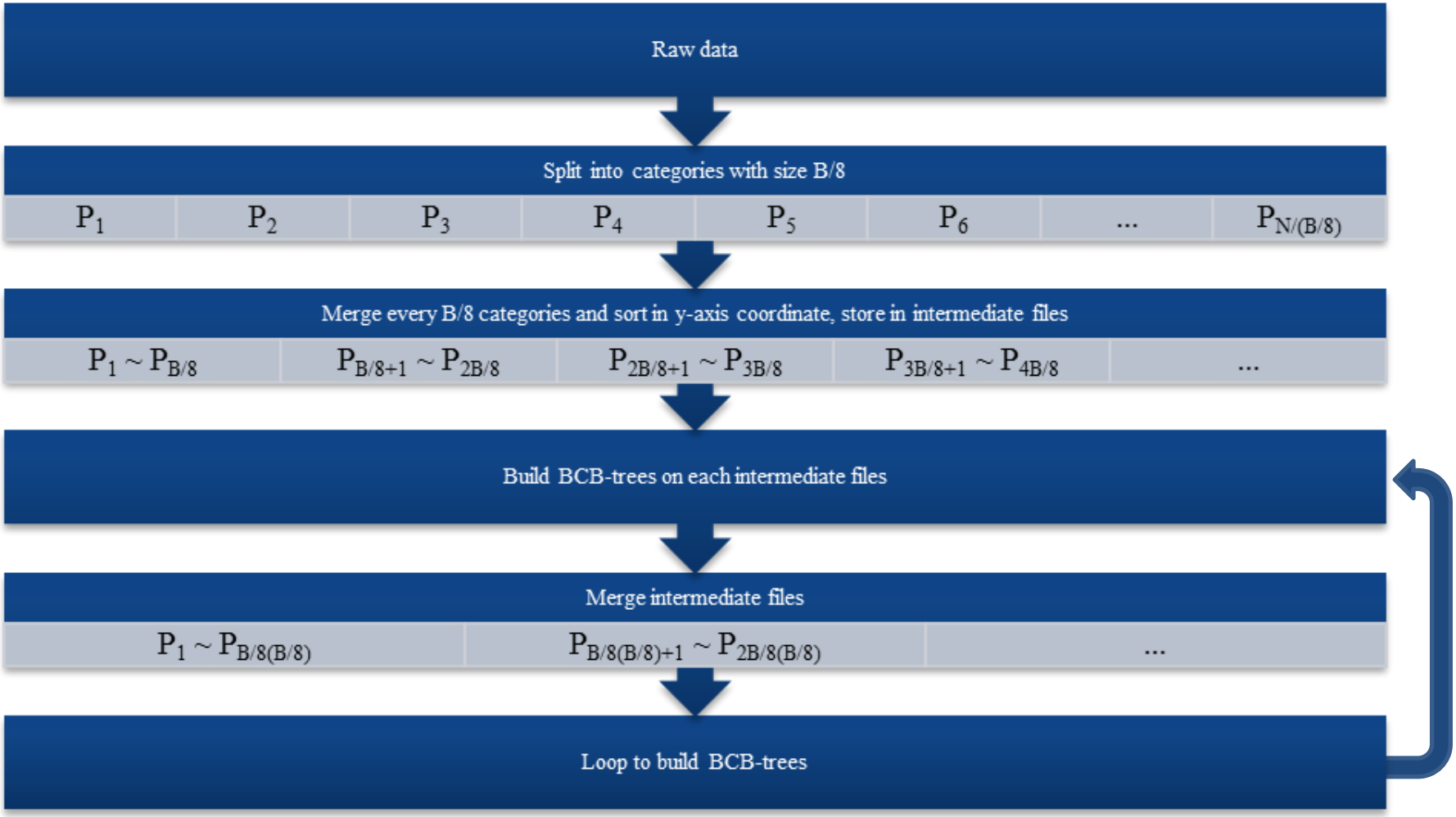
Typo in report(p.11) O(N / (Blog_B(N))) becomes O(N / B log_B(N)), which is incorrect

Range Count (revisit)



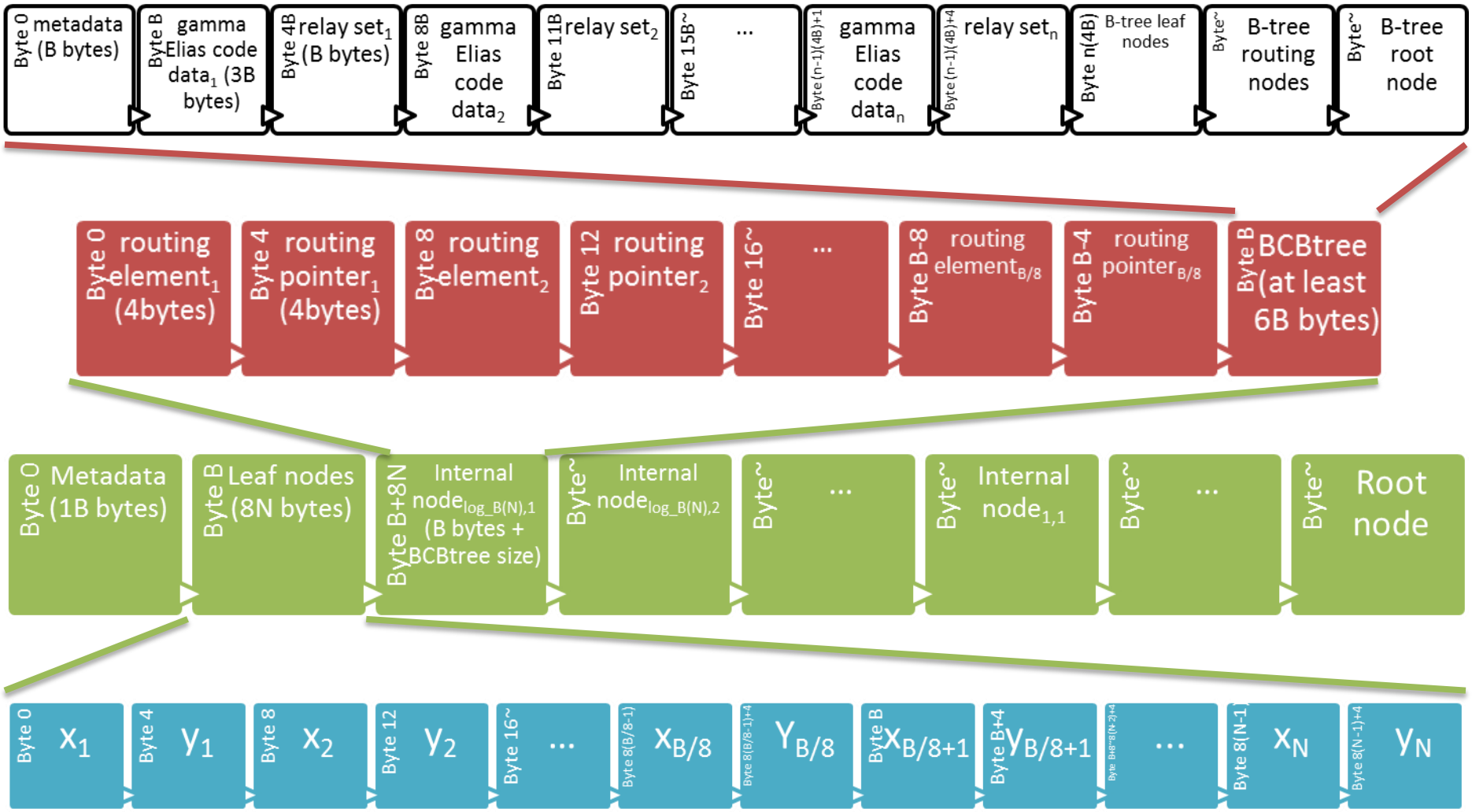
- Complexities:
- Building: $O\left(\frac{N}{B} log_B N\right)$ I/Os; query: $O(log_B^2 N)$ I/Os; space:$O\left(\frac{N}{B}\right)$ Blocks
- Query cost can be further optimized to $O(log_B N)$I/Os

Range Count (cont.)



Building: $O\left(\frac{N}{B} log_B N\right)$ I/Os

Range Count (cont.)



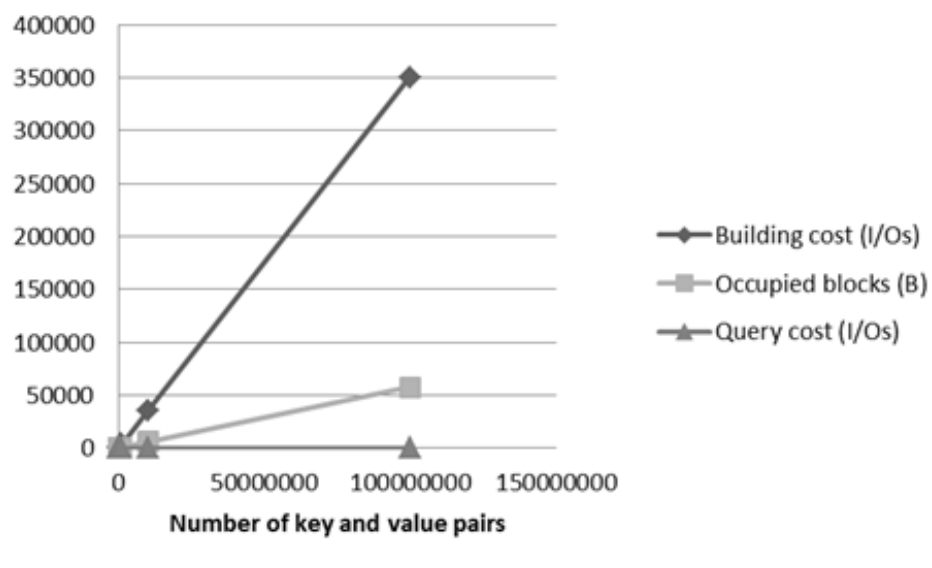
API

```
int build_BCBtree(const char* input_file_name, const char*
output_file_name, int block_offset);

int query_BCBtree(const char* input_file_name, int
block_offset, int** output_list, int query);

int build_BCBtree_range_count(const char* input_file_name,
const char* output_file_name);

int query_BCBtree_range_count(const char* input_file_name,
int query_x, int query_y);
```

Performance Evaluation

- BCB-tree

Number of key and value pairs	Building cost (I/Os)	Occupied blocks (B)	Query cost (I/Os)
10	7	6	6
100	7	6	6
1000	9	6	6
10000	40	10	6
100000	356	62	6
1000000	3507	574	6
10000000	35031	5705	7
100000000	350302	57046	7

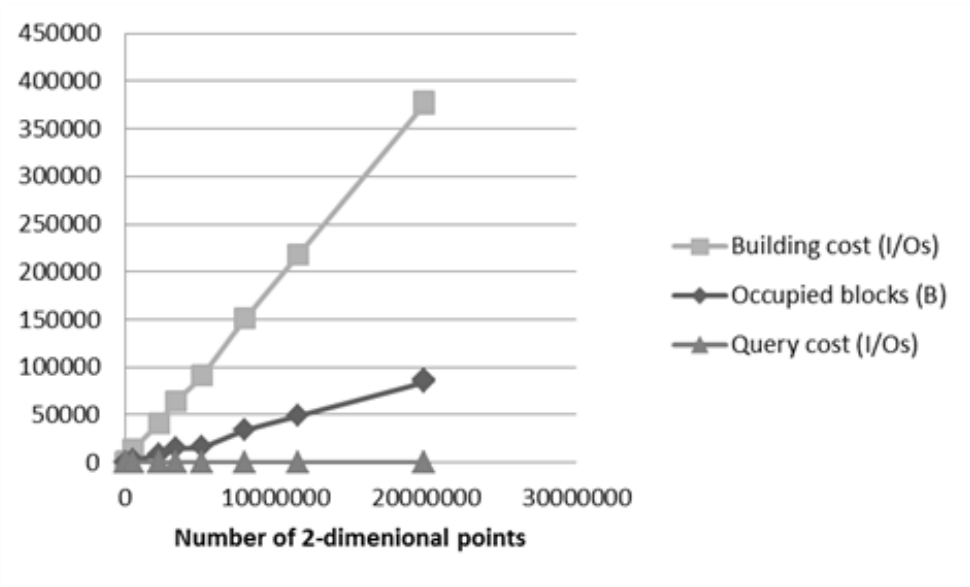


- B = 4096 Bytes

Performance Evaluation

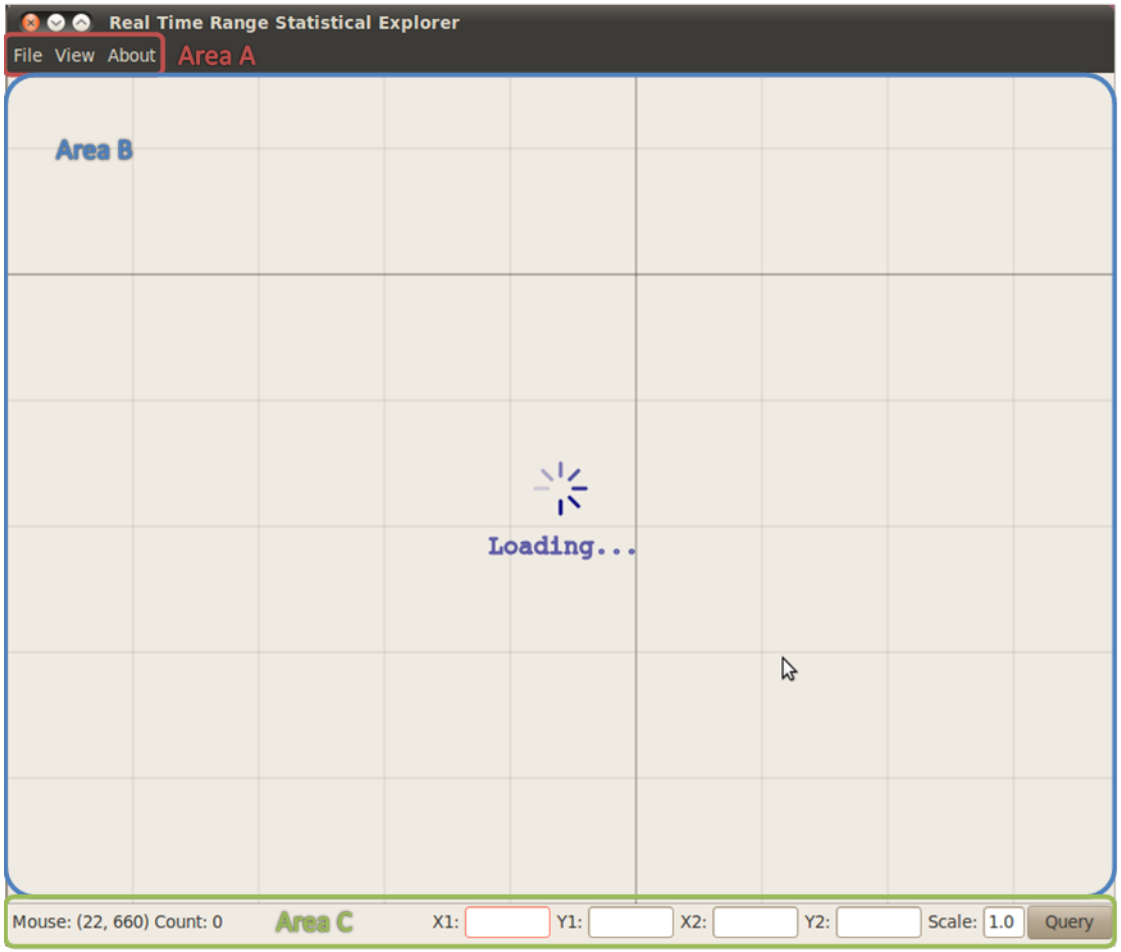
- Range count structure using BCB-tree

Dataset name	Number of 2-dimenional points	Building cost (I/Os)	Occupied blocks (B)	Query cost (I/Os)
minimal.dat	2	11	9	8
LA_original.dat	131461	1205	433	9
TCB_original.dat	556696	13751	2229	16
CAR.dat	2249727	41005	8009	16
CAR_original.dat	2249727	41245	8249	16
RAIL.dat	3350387	63852	14713	17
RedList.dat	5126830	91538	16348	17
UAC.dat	7980981	150666	33617	17
BLOCK.dat	11496067	217394	48796	17
NE.dat	19909725	376205	84216	17
NE_original.dat	19909725	379122	87133	17

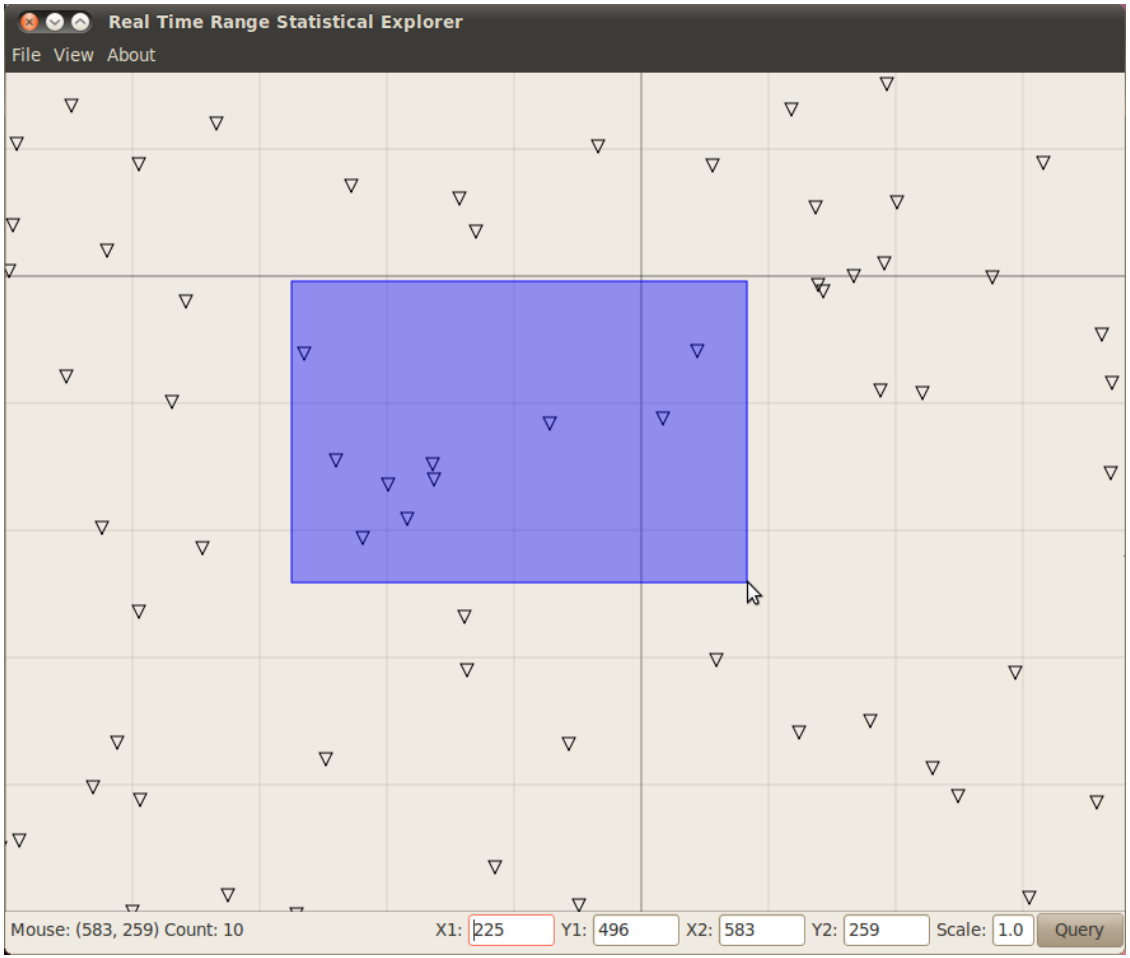


- B = 4096 Bytes

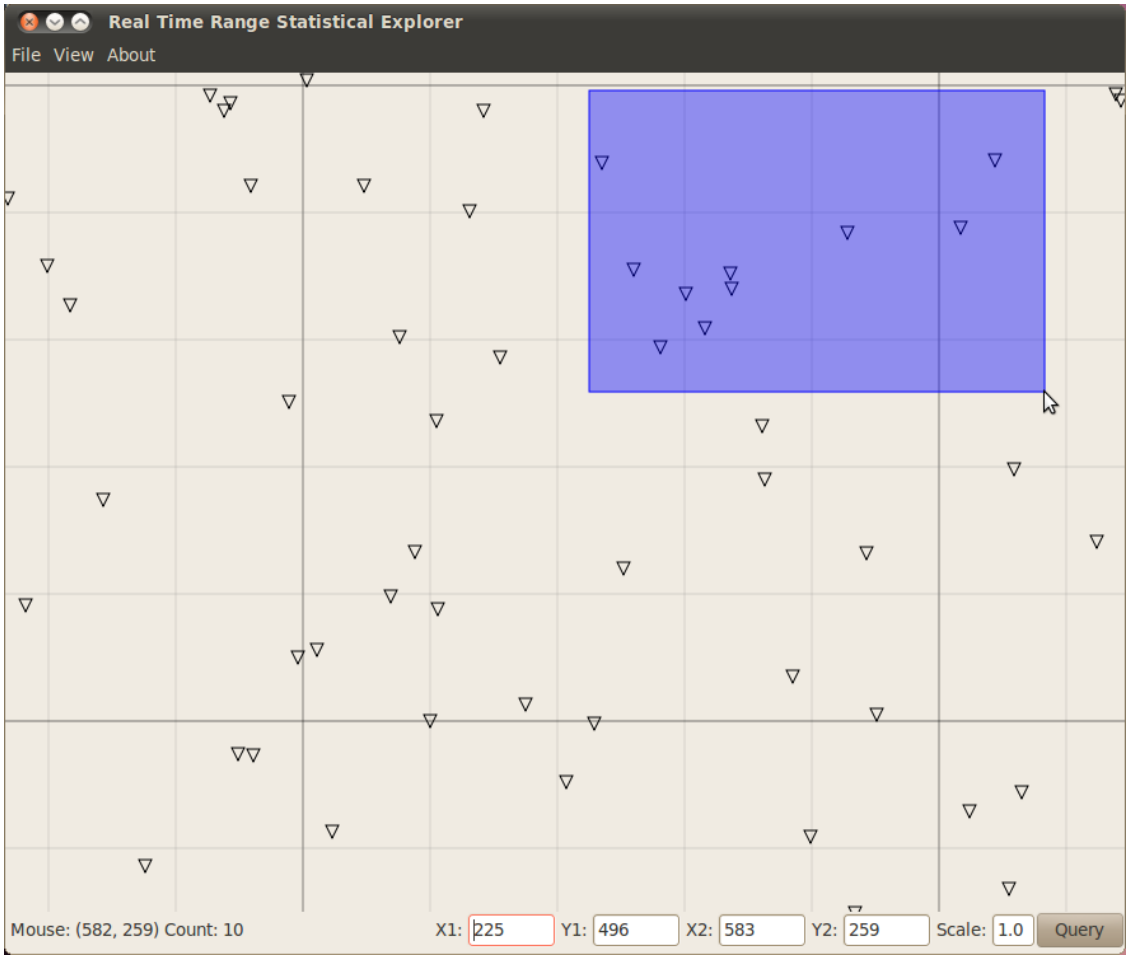
Real Time Range Statistical Explorer



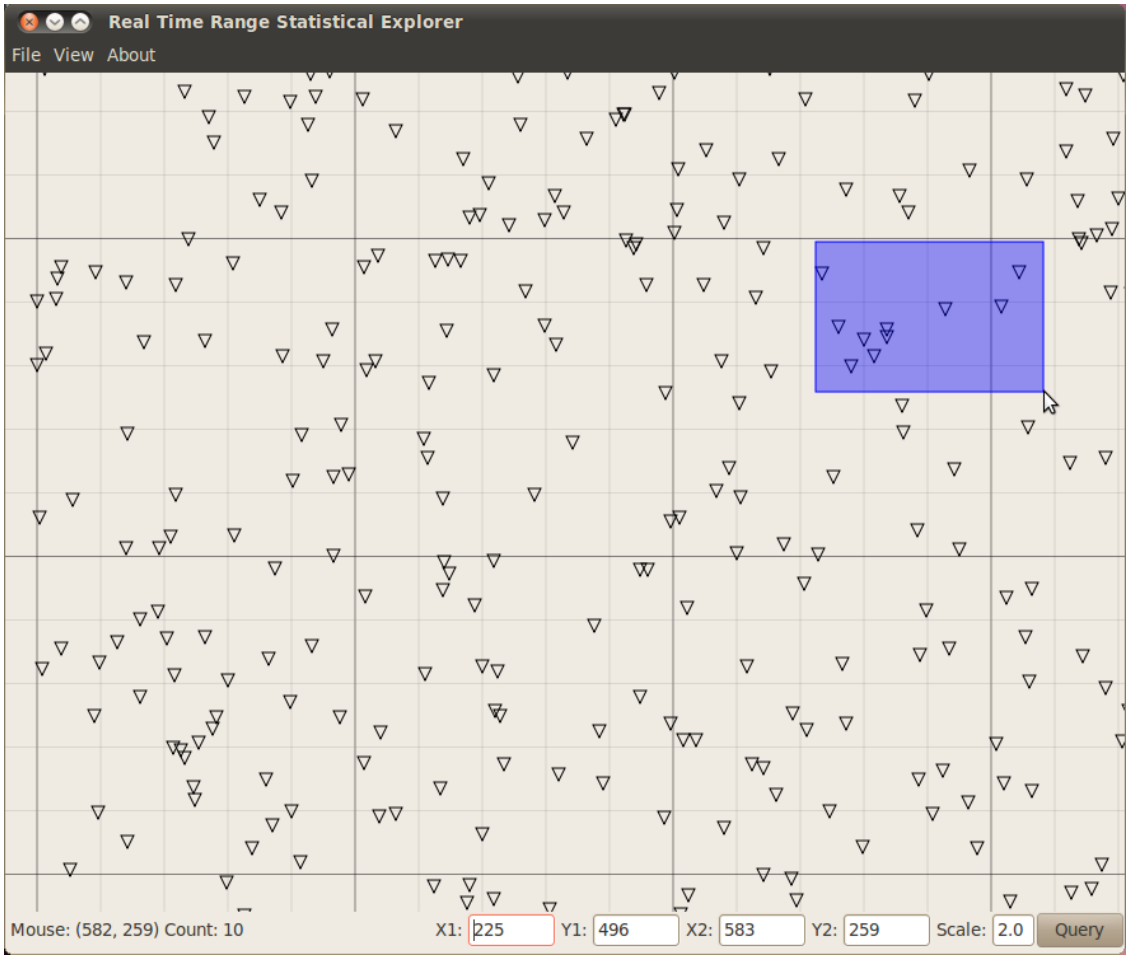
Query by Dragging



Move around by Ctrl + Dragging



Zoom by Scrolling



Q & A