Spatial Online Sampling and Aggregation

Lu Wang[1], Robert Christensen[2], Feifei Li[2], Ke Yi[1]

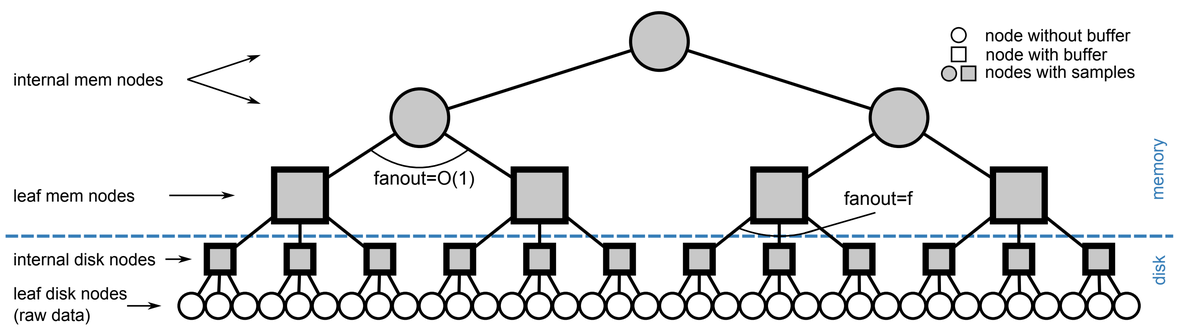
[1]Hong Kong University of Science and Technology
{luwang, yike}@cse.ust.hk

[2]University of Utah
{robertc, lifeifei}@cs.utah.edu

Motivation

- Geo-Spatial data is being collected on a massive scale.
- Approximate analysis is *fast* and often *effective* for this data.

Hybrid R-Tree



- Top levels are *memory only nodes* which are persisted in main memory for fast access.
- Lower levels are *disk nodes* which are persisted on disk and loaded into main memory when needed.

Baseline Algorithms

Query First — Calculate $S = P \cap Q$. Repeatably extract a sample from S upon request.

Sample First — Upon Request, pick a point p where $p \in P$. If $p \in Q$ report p, otherwise repeat.

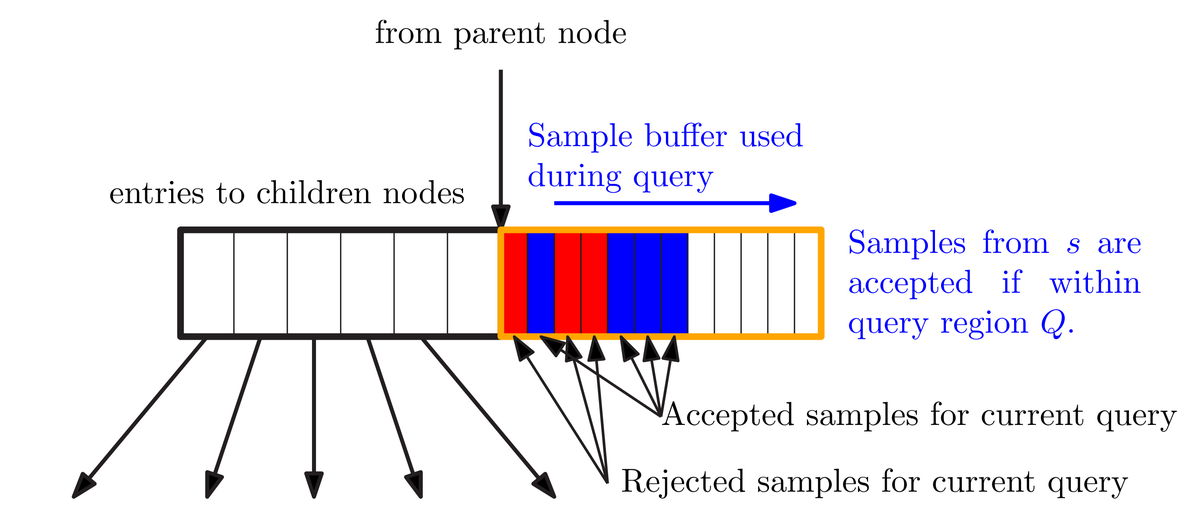
Data Structures

R-tree — A Hybrid R-tree implementation based off of Hilbert R-tree.

RS-tree — A single R-tree T with a sample buffer s attached to each internal node. For each internal node $u \in T$ with sample buffer s, we sample uniformly from the children of u, $p(x = y | x \in s \wedge p \in P(u)) = \frac{1}{|P(u)|}$

LS-tree — a collection of R-trees where each R-tree indexes a set of samples from the original data set. The sample rates for these sets of samples form a geometric series.

Individual Nodes of RS-tree



Once all local samples are observed during a query, children within MBB are added to *frontier*

Notation

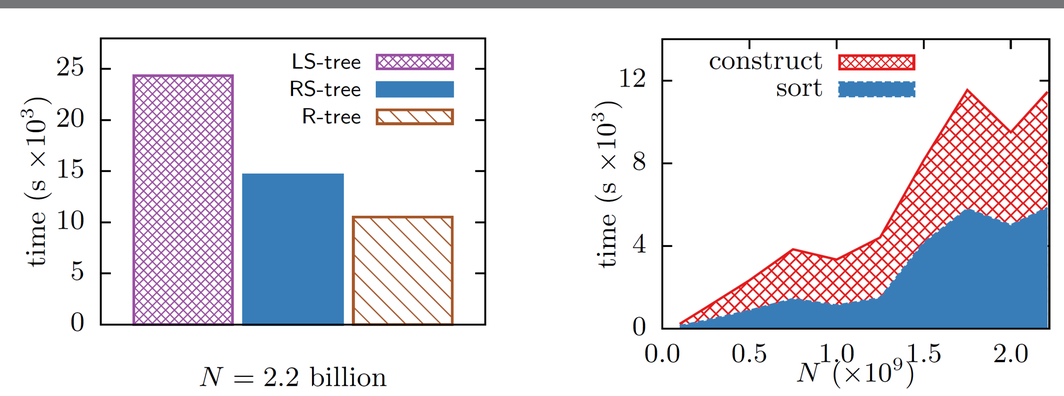
P	The raw data set in $\mathbb{R}^d$.
k	The number of samples to report.
N	$\|P\|$, the size of the raw data set.
Q	A range query in $\mathbb{R}^d$.
P_Q	$P \cap Q$, elements in the query range.
q	$\|P_Q$, the number of elements in the query range.
$u, v, \cdots$	Tree nodes.
$T(u)$	The subtree rooted at node u.
$P(u)$	The set of all data points covered by $T(u)$.
$r(N)$	The size of a canonical set in a R-tree of size N.
B	The size of a disk block.
s	The sample buffer size in RS-tree.

Comparison of various sampling algorithms

Algorithm	Query Time	Update Time
QueryFirst	$r(N) + q$	$\log N$
SampleFirst	kN/q	$\log N$
RandomPath	$k \log N$	$\log N$
RandomShuffle	kN/q	$\log N$
LS-tree	$\sum_{j=\log q/k}^{\ell} r\left(\frac{N}{2^j}\right) + k$	$\log N$
RS-tree	$r(kN/q) + k$	$\log N$

- Full analysis of algorithms, including I/O cost can be found in the paper.

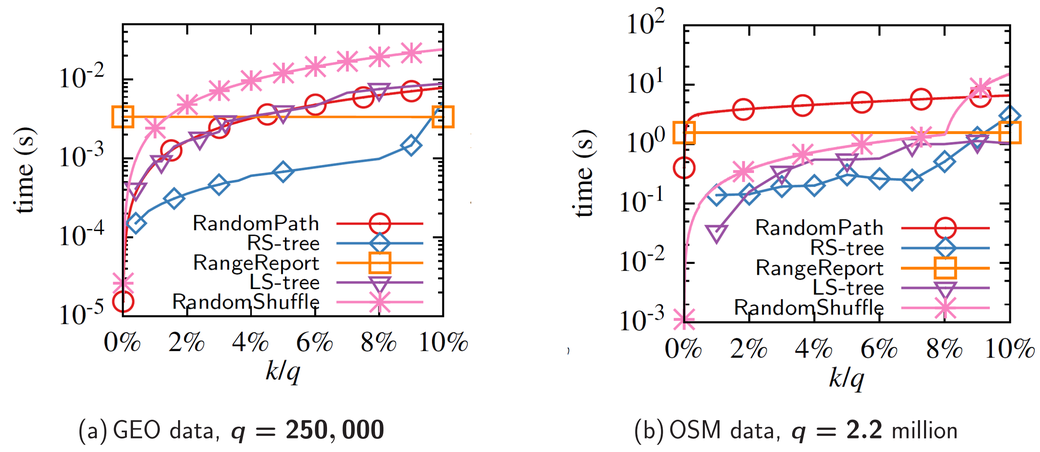
Building Index Experiments



(a) construction time on full OSM (b) scalability of RS-Tree.

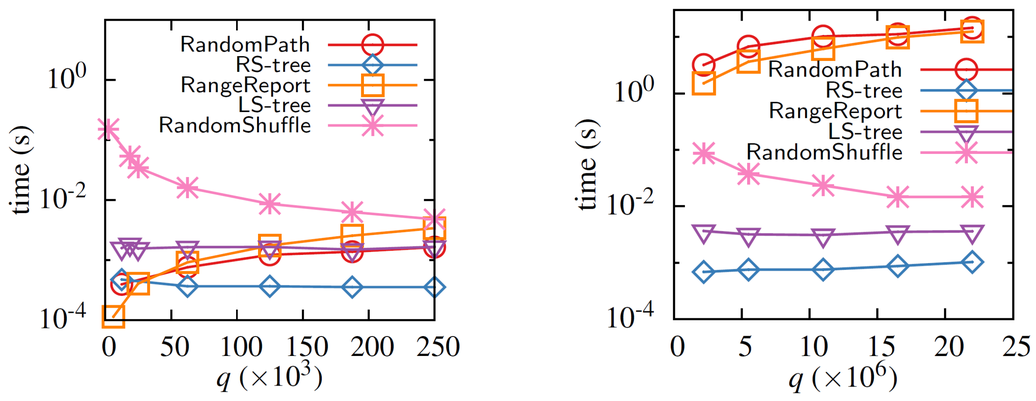
Figure: Construction time of different indexes.

Sample Query Experiments



(a) GEO data, $q = 250,000$ (b) OSM data, $q = 2.2$ million

Figure: vary k, the number of samples.



(a) construction time on full OSM (b)

Figure: vary q, the number of samples in the query region.