GeoPrivacy: 2nd Workshop on Privacy in Geographic Information Collection and Analysis

Differentially Private H-Tree

Hien To, ***Liyue Fan***, *Cyrus Shahabi*
Integrated Media System Center
University of Southern California

November 3, 2015

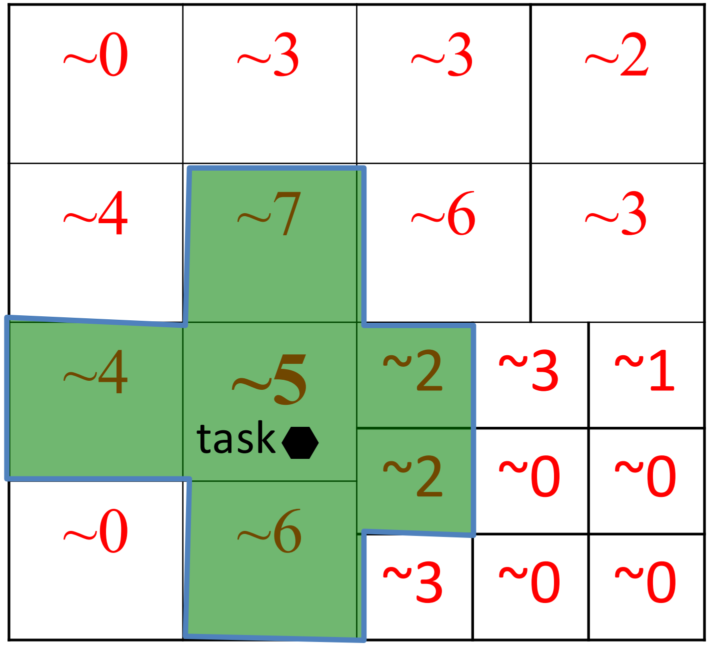
Motivation

Mobile devices collect/share *location data*

- Enable applications, e.g., spatial crowdsourcing, traffic monitoring, location-aware recommendation
- Adversary can infer users' *sensitive details*

Many location-based apps require only spatial aggregation of users

- e.g., spatial crowdsourcing
- Differential privacy serves that purpose



Noisy worker count per grid cell

Differential Privacy (DP)

Ensures adversary do not know whether an individual is present or not in dataset, regardless of background knowledge

Allows only aggregate queries, e.g., count, sum

ε*-indistinguishability* $\ln \frac{\Pr[QS^{D_1} = U]}{\Pr[QS^{D_2} = U]} \le \varepsilon$ *[Dwork'06]*

ε : privacy budget

L_1 *-sensitivity* $\sigma(QS) = \max_{D_1, D_2} \sum_{i=1}^{q} | QS(D_1) - QS(D_2) |$

D1 and D2 are sibling datasets that differ in only one record

Achieve ε*-DP* by adding random Laplace noise with mean 0 and standard deviation $\lambda = \sigma(QS) / \varepsilon$ *[Dwork'06]*

USC Viterbi School of Engineering Integrated Media Systems Center

Problem Definition

Publish private spatial decomposition (PSD) of 2-d dataset

Accurately answer count queries

Range query fully covers 2 cells and partially covers 2 cells → Estimated result set size:

200*2/2+50*2=300

Relative error

$$\mathrm{RE}_{PSD}(q) = \frac{Q_{PSD}(q) - A(q)}{A(q)}$$

actual count

published noisy counts

0 **50**	0 **50**	0 **100**
0 **200**	0 **200**	0 **0**
100 **50**	100 **50**	100 **200**

Related Work

- ✓ Kd-tree on top of fixed equal-size grid *[Xiao et al. 2010]*
- ✓ Wavelet transformation *[Xiao et al. 2011]*

- ✓ Kd-tree, Quad-tree *[Cormode et.al ICDE 2012]*

Perturbation error is excessively high on hierarchical partitions and high dimensional data ☹

- ✓ Uniform grid, adaptive grid *[Qardaji et al. ICDE 2013]*
- ✓ Extend to higher dimension *[Qardaji et al. VLDB 2013]*

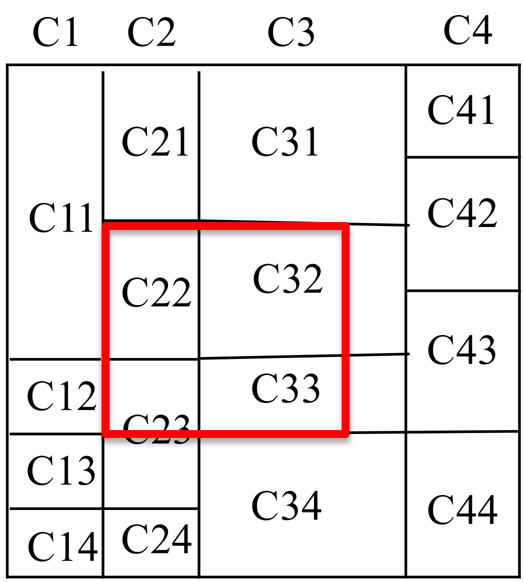
Grid-based partitions are not ideal for skewed datasets ☹

- ✓ H-Tree: two-level data-dependent tree *[This study]*

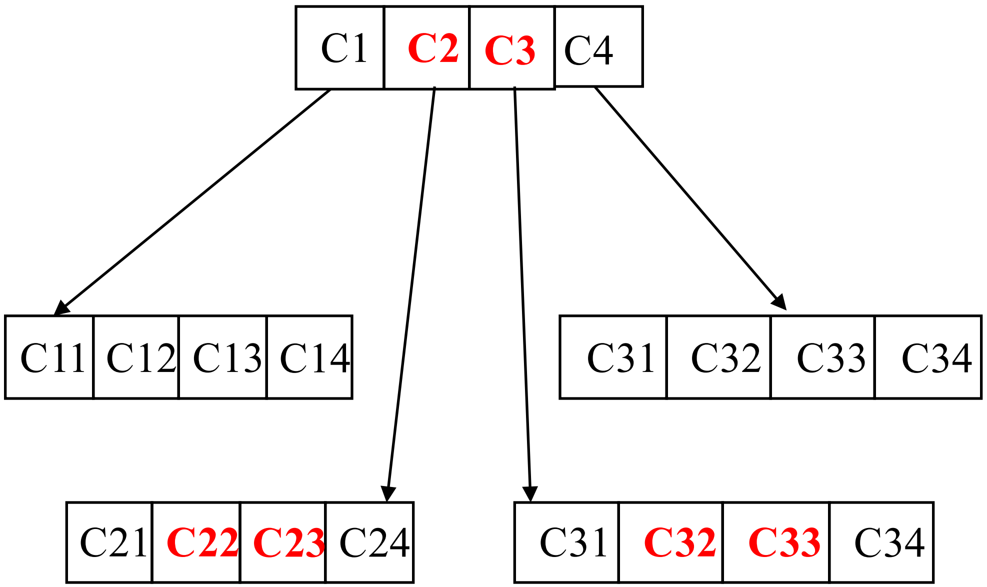
Kd-tree
Adaptive grid

Differentially Private H-Tree

Equi-depth multidimensional histograms *[Muralikrishna et. al SIGMOD 1988]*



H-Tree of size m=4



H-Tree structure

Canonical range query processing minimizes total error

1) Granularity 2) Count/Median budget 3) Post-processing

Granularity

Compute H-tree's size mxm that minimizes query estimation error

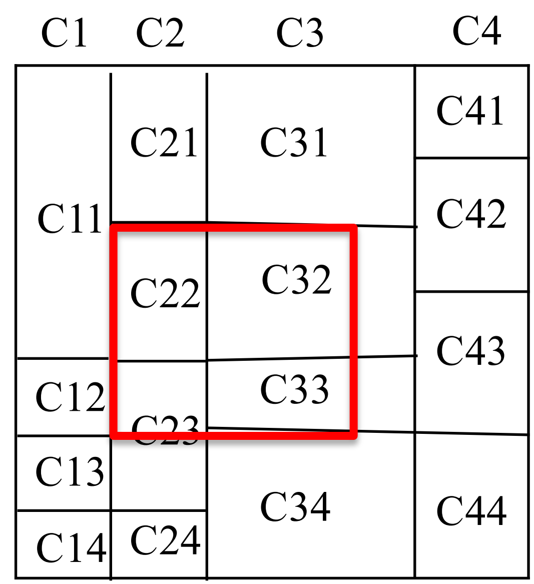
Perturbation error vs. Non-uniformity error

$$\sqrt{\frac{m^2 w}{W}} \times \frac{\sqrt{2}}{\varepsilon^c} \quad \text{trade-off} \; + \quad \frac{4\sqrt{wW}}{c_0 m}$$

leaf nodes

Laplace error

data points in the border



H-tree partition

Query size increase

→ Perturbation error increases

→ Non-uniformity error decreases

Granularity $m = \sqrt{W\varepsilon^c / c}$

- c is a small constant
- W is the domain size
- ε^c is the count budget

Budget Allocation Strategy

Two kinds of budgets

1. Median budget for 2 levels

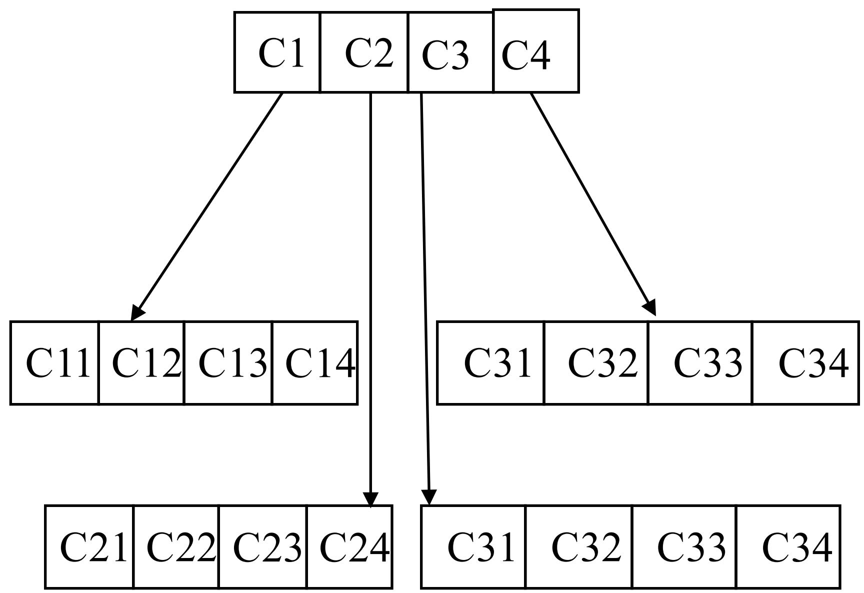
$$\varepsilon^{m}$$

1. Count budget for 2 levels

$$\varepsilon^{c} = \varepsilon_1^{c} + \varepsilon_2^{c}$$

Total budget

$$\varepsilon = \varepsilon^{m} + \varepsilon^{c}$$



H-Tree structure

Count Budget Allocation

Split count budget across levels of the tree index

$$Minimize\ \ Err(q) = n_1 \frac{2}{(\varepsilon_1^c)^2} + n_2 \frac{2}{(\varepsilon_2^c)^2},\ \ subject\ \ to\ \ \varepsilon^c = \varepsilon_1^c + \varepsilon_2^c$$

- n1: number of level-1 nodes
- n2: number of level-2 nodes

$$\boxed{n_2 \approx m \times n_1}$$

The proof uses Cauchy Schwarz inequality

$$\left(\varepsilon_1^c + \varepsilon_1^c\right)\left(\frac{n_1}{(\varepsilon_1^c)^2} + \frac{n_2}{(\varepsilon_2^c)^2}\right) \geq \left(\frac{\sqrt{n_1}}{\sqrt{\varepsilon_1^c}} + \frac{\sqrt{n_2}}{\sqrt{\varepsilon_2^c}}\right)^2$$

Err(q) is minimized when

$$\boxed{\varepsilon_1^c = \frac{\varepsilon^c}{1+\sqrt[3]{m}}, \varepsilon_2^c = \frac{\varepsilon^c \sqrt[3]{m}}{1+\sqrt[3]{m}}}$$

Median Budget Allocation

Private H-Tree requires selecting private medians

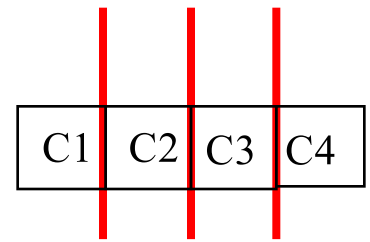
Splits apply to the same data → sequential composition

Recursively splits each dimensional range → parallel composition

Each split $$\frac{\varepsilon^{m}}{2\log_2 m}$$

Use exponential mechanism

[McSherry SIGMOD 2009]

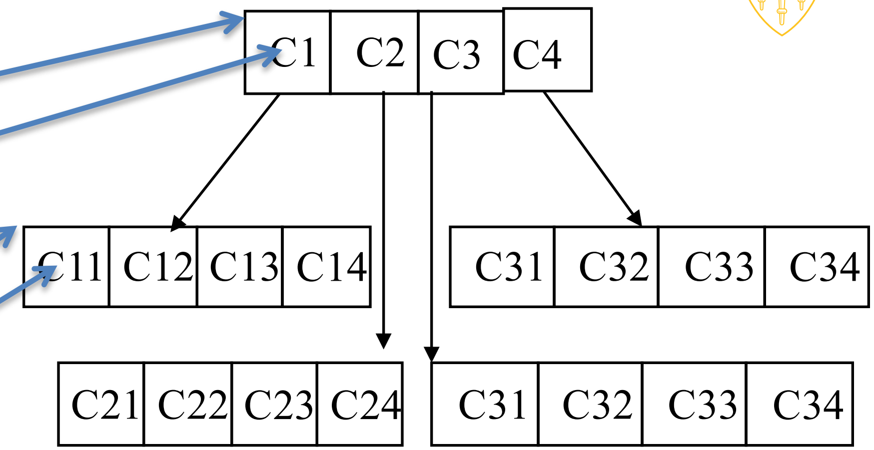


Proposed **Slicing Algorithm** recursively splits a range at points that are closest to the corresponding medians

DP H-tree Algorithm

Input: h-tree of size mxm

1. *Median budget* ε_1^m
2. *Count budget* ε_1^c
3. *For each level-1 node:*
    1. *Median budget* ε_2^m
    2. *Count budget* ε_2^c



H-Tree structure

The entire H-tree satisfies ε-DP by composition property

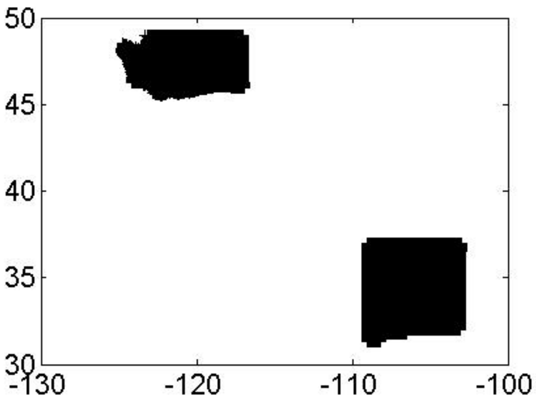
$$\varepsilon = \varepsilon^m + \varepsilon^c \qquad \varepsilon^m = \varepsilon_1^m + \varepsilon_2^m \qquad \varepsilon^c = \varepsilon_1^c + \varepsilon_2^c$$

Trade-off between median budget and count budget

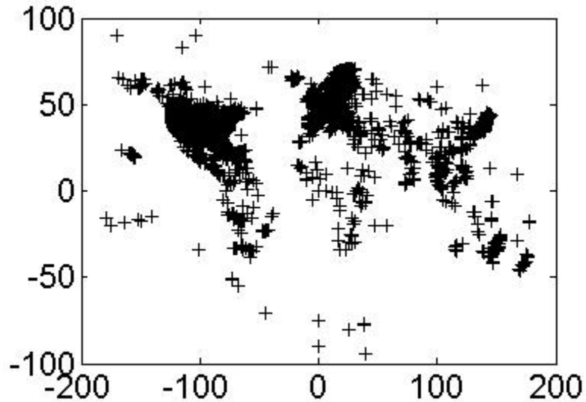
$\varepsilon^m = 0.3\varepsilon$ *[Cormode et.al ICDE 2012]*

Experimental Setup

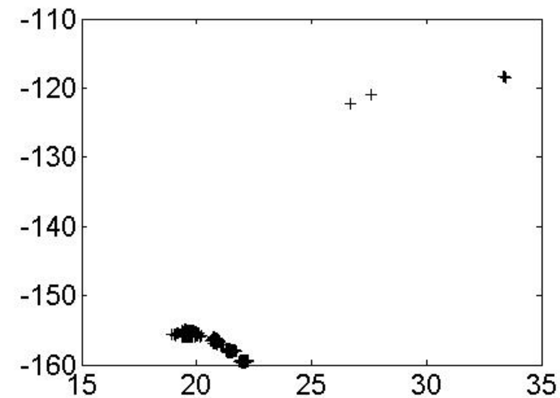
Datasets



Tiger_NMWA



Brightkite



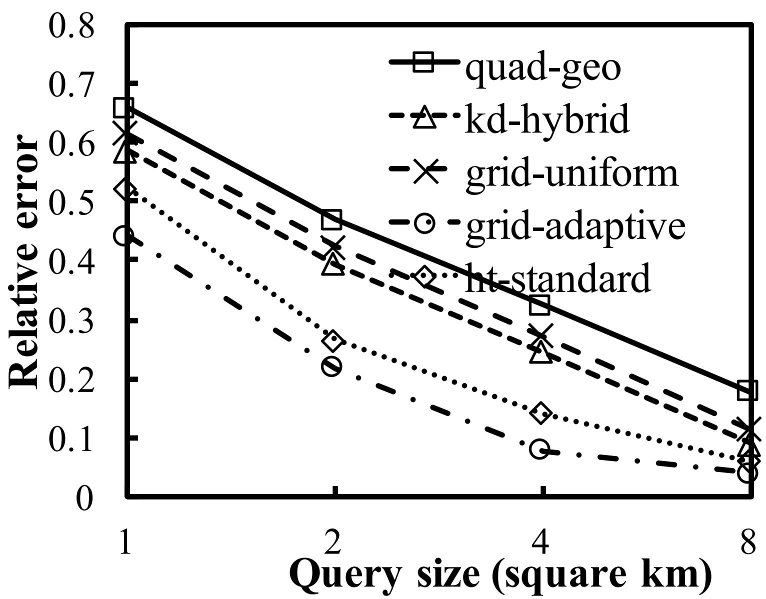
Gowalla-Sparse

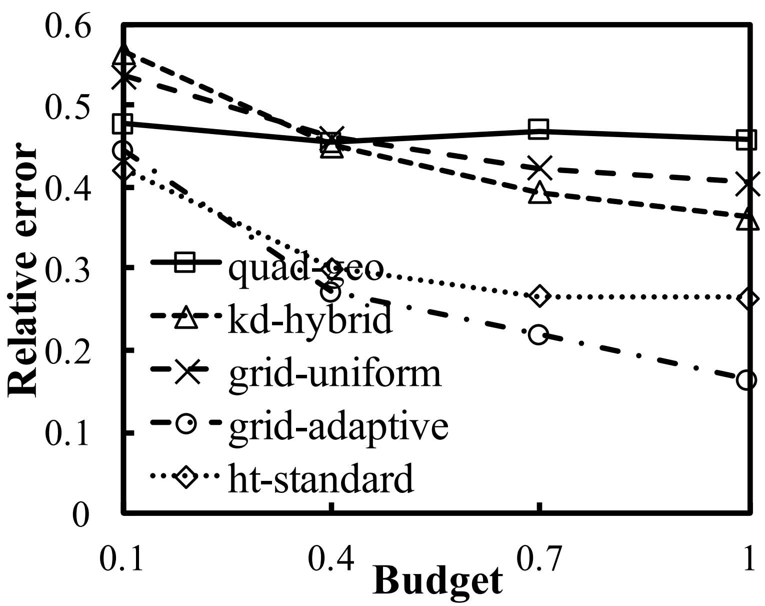
Queries

- Privacy budget $\varepsilon = \{.1, .4, .7, 1\}$
- Query size = {1, 2, 4, 8} square km
- $\varepsilon^m = 0.4\varepsilon$; $c = 3$
- Average relative error over 1000 random queries

USC Viterbi
School of Engineering
Integrated Media Systems Center

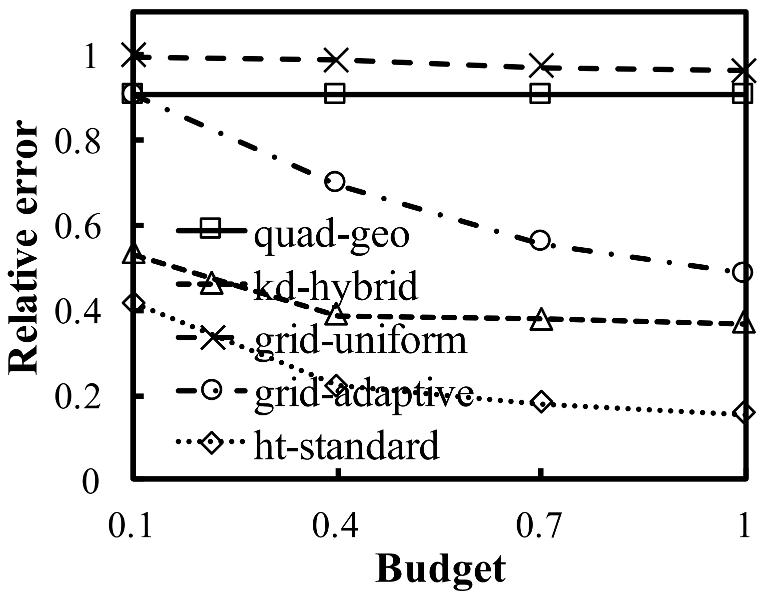
Tiger dataset (similar result on Brightkite)



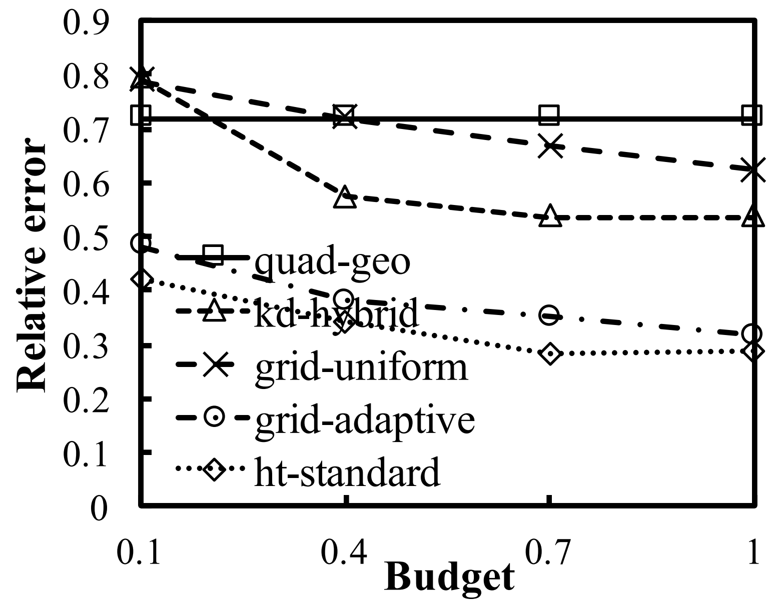


Grid-adaptive performs well and even better than data-dependent methods

Gowalla-Sparse



Tiger-Syn



Grid-adaptive performs arbitrarily worse in the presence of sparseness and outliers

Conclusion

- ✓ Observed drawbacks of high-level trees and grid-based structures
- ✓ Proposed several analysis on DP H-tree, i.e., budget allocation, median splitting, post-processing
- ✓ DP H-Tree consistently performs well on various datasets
  - i.e., h-tree outperforms kd-tree and quadtree in all cases and adaptive grid for sparse datasets

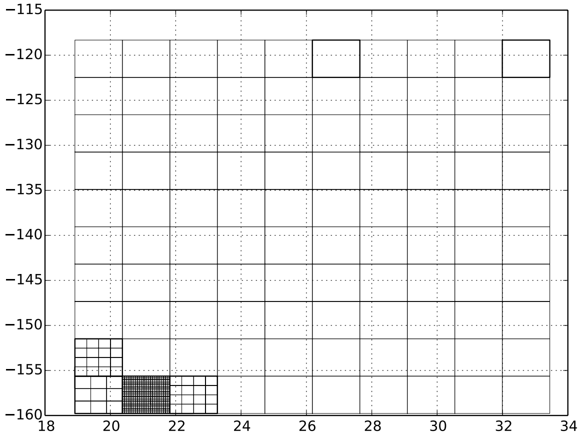
Q/A

Liyue Fan

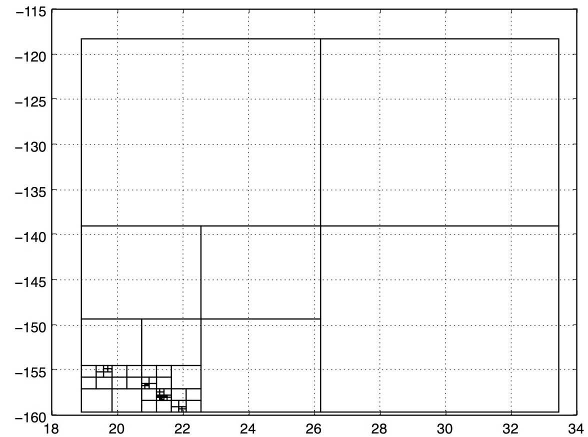
University of Southern California

liyuefan@usc.edu

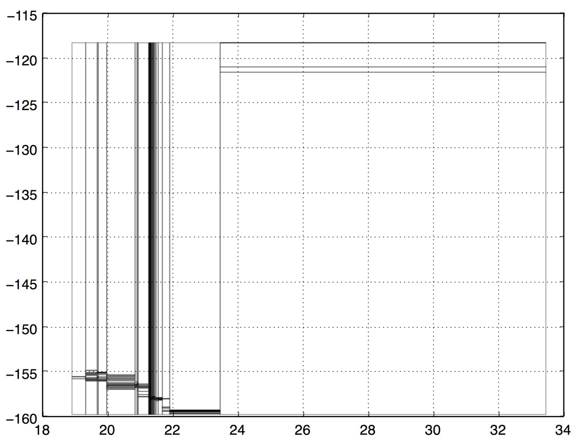
Partitions



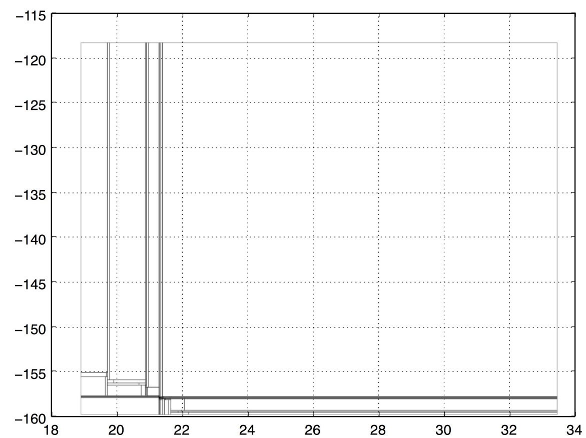
Adaptive grid



Quadtree



H-tree



Kd-tree