Optimizing hierarchical queries for the attribution reporting API

Matthew Dawson, Badih Ghazi, Pritish Kamath, Kapil Kumar, Ravi Kumar, Bo Luan, Pasin Manurangsi, Nishanth Mundru, Harikesh Nair, Adam Sealfon, Shengyu Zhu
Google

The Attribution Reporting API

- Privacy-preserving tool for ad conversion measurement on Chrome/Android
- Can produce aggregate statistics about conversion attribution without using persistent cross-site identifiers
- Summary reports satisfy differential privacy: noise is added to limit how much can be inferred about individual impressions

Conversion reporting

- Goal: estimate the number of conversions attributed to impressions, where the impressions and conversions have a certain combination of features
- E.g. how many conversions were attributed to impressions from campaign 123 and took place in Los Angeles last Friday?

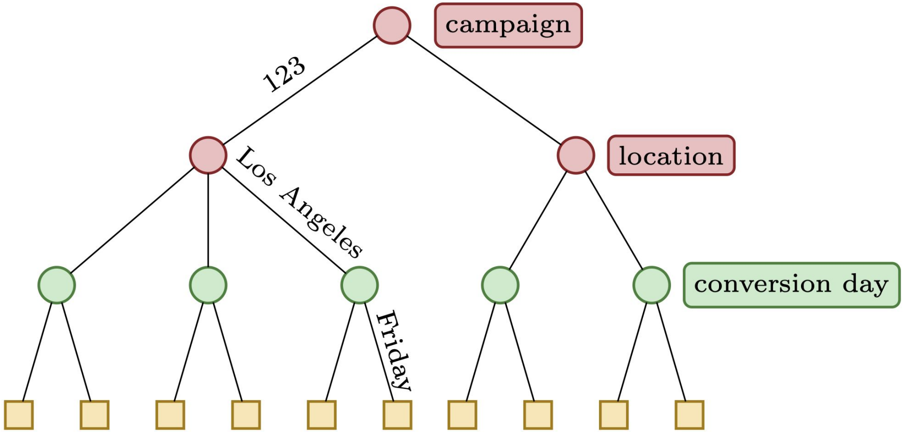
Click	campaign	location	conversion day
1	123	Paris	Monday
2	456	Chicago	Friday
3	789	London	*
4	123	Los Angeles	Friday
…	…	…	…

Hierarchical queries

- For each (city, day) of the campaign, how many attributed conversions?
- Higher-level aggregates: what is the total number of attributed conversions for this campaign? What about the total number in Los Angeles?
- Goal: given a tree that branches on impression/conversion features, want to estimate the number of conversions corresponding to each node in the tree.

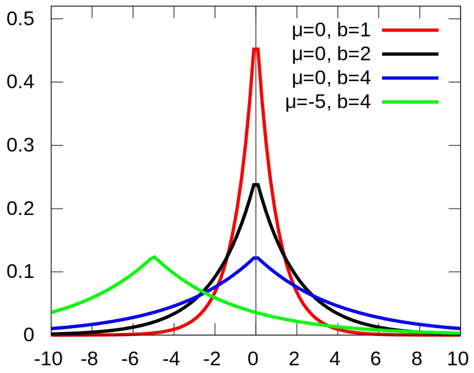
Error metric: thresholded RMS relative error averaged over the levels of the tree (same algorithms work with other metrics)

$$\mathrm{RMSRE}_\tau(T) := \sqrt{\mathbb{E}\left[\frac{1}{d+1}\sum_{i=0}^{d}\frac{1}{|L_i|}\sum_{v\in L_i}\left(\frac{|\hat{c}_v - c_v|}{\max(\tau, c)}\right)^2\right]}$$



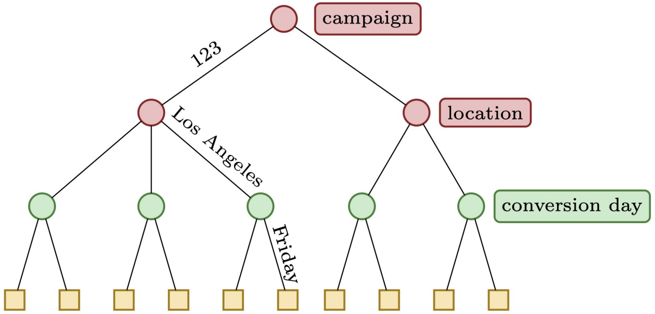
Differential privacy (DP) and the Laplace mechanism

- DP provides worst-case guarantees about how much an adversary can infer about a single row of the dataset
- Privacy level is controlled by a parameter ε > 0; smaller ε ⇔ more private
- For a counting query, can satisfy ε-DP by adding noise of scale 1/ε from a (continuous or discrete) Laplace distribution.
- Such estimates can be obtained using the Attribution Reporting API



Privacy budgeting and hierarchical queries

- What if multiple queries involve the same data record?
- Composition: Algorithms A_1, A_2 are ε_1-DP and ε_2-DP ⇒ (A_1,A_2) is $(\varepsilon_1+\varepsilon_2)$-DP
- In a tree:
  - Queries to different nodes at the same level touch disjoint subsets of the data
  - Queries to nodes at different levels may touch the same data record
- Given a total privacy budget ε, can allocate it to the d+1 levels of the tree so that $\varepsilon_0 + \varepsilon_1 + \cdots + \varepsilon_d = \varepsilon$



Main question

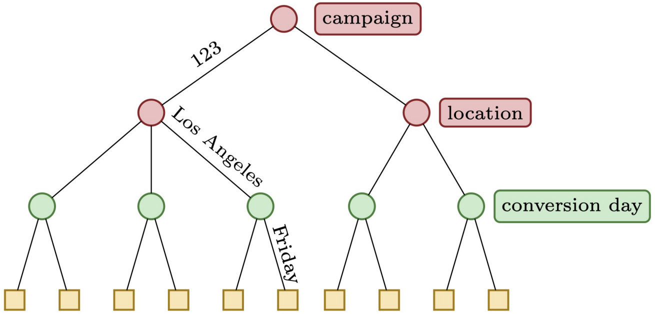
How can we obtain estimates for hierarchical queries that are consistent and have minimum possible error?

Two main results:

- A post-processing algorithm that reduces the error of estimates and ensures consistency with the hierarchical structure
- A procedure for optimizing the allocation of the privacy budget among the levels of the hierarchy

Post-processing algorithm

- Observation: the value of any internal node should equal the sum of the values of its children (*consistency*).
- Given independent estimates e_1, e_2 of the same quantity with variances v_1, v_2:
  - Can obtain other unbiased estimates by taking a convex combination $\alpha e_1 + (1- \alpha) e_2$
  - The optimal combination has $\alpha = v_2 / (v_1 + v_2)$, yielding an improved variance of $v_1 v_2/(v_1+v_2)$
- How can we optimally take into account all constraints encoded in the tree?



Post-processing algorithm

Given: estimates z_v of the count at each node v, and their variances var_v

Bottom-up pass. For each internal node v from largest to smallest depth:

Update z_v to be the minimum-variance convex combination of z_v and $\sum_{u \in child(v)} z_u$, and compute the corresponding variance var_v.

Top-down pass. For each internal node v from smallest to largest depth:

Update z_u for each $u \in$ child(v) by splitting the discrepancy $z_v - \sum_{u \in child(v)} z_u$ among the children proportionally to the variance var_u of each child estimate.

Output the final estimates z_u

Post-processing algorithm

- Optimal: computes best linear unbiased estimator
- Better privacy/accuracy tradeoff: given noisy estimates for each tree node, produces estimates with lower error, without any additional privacy leakage
- Produces consistent estimates
- Linear-time algorithm
- Can be extended to compute variances as well as estimates
- Extends the methods of [Hay et al., VLDB’10, Cormode et al., ICDE’12], which apply to regular trees; also related to the matrix mechanism of [Li et al., VLDB ’15, Nikolov et al., STOC’13], which in general requires ≥ quadratic time (n^{ω}).

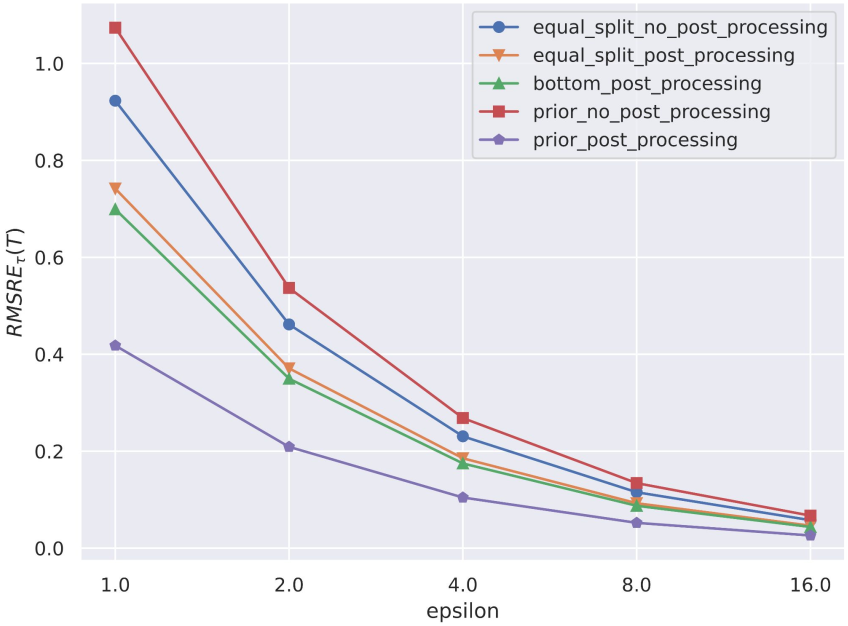
Allocating the privacy budget

- Post-processing tells us the optimal way to use a set of measurements, but which measurements should we take?
- For total budget ε, can split it in many ways among the levels of the tree
- Given (noisy) historical data or a prior, can compare these options
- Optimize to choose the best privacy budget split
- Simple greedy approach:
    - Divide total budget into k increments
    - In each iteration, allocate ε/k additional budget to the level that most decreases the overall error after post-processing

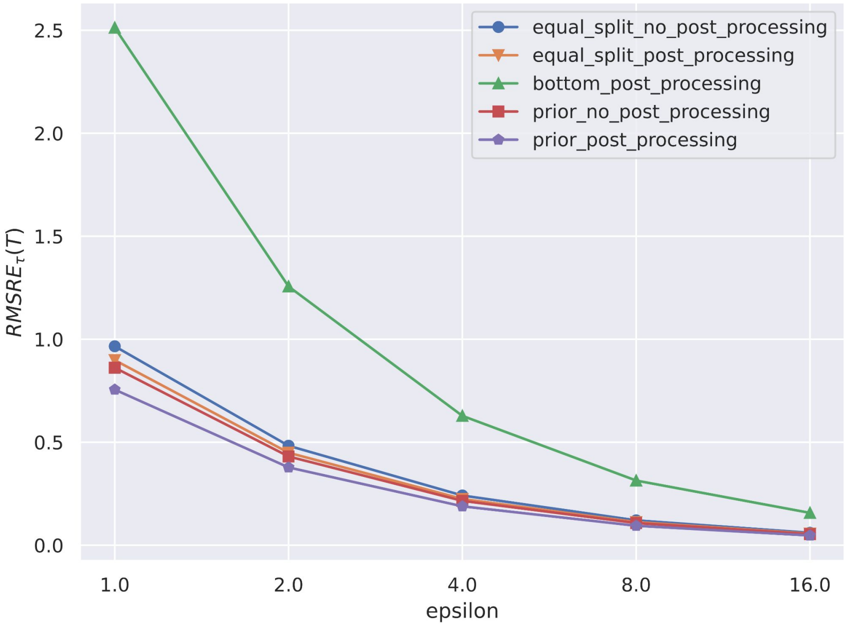
Evaluation

- Evaluated on two public Criteo datasets, Sponsored Search Conversion Log (CSSCL) and Attribute Modeling for Bidding (CAMB)
- Selected attributes from each dataset to construct hierarchy
- Split datasets into budgeting data and test data based on click time
- Compared five approaches:
  - equal budget split, with and without post-processing
  - all budget on bottom level, with post-processing
  - optimizing per-level privacy budgets, with and without post-processing

Evaluation

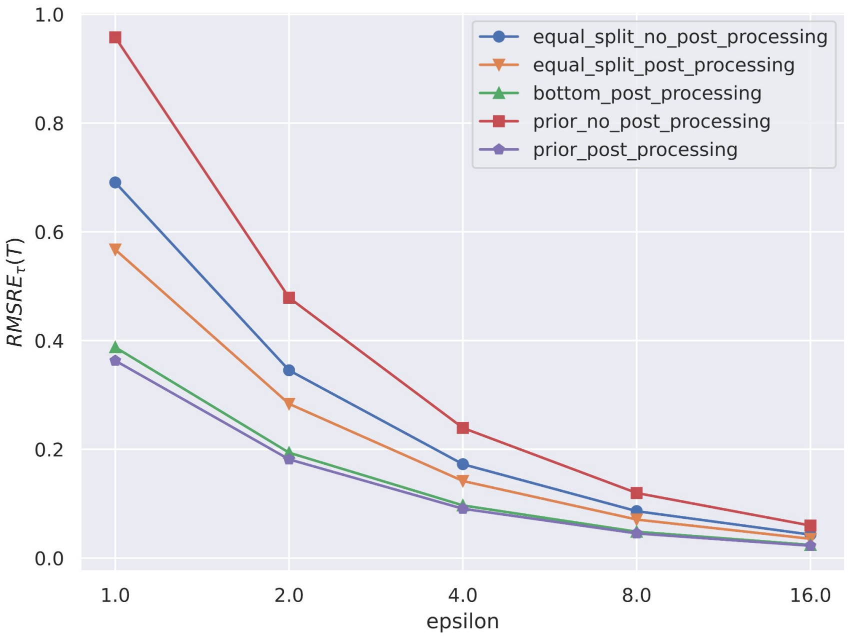


Five-attribute hierarchy using Criteo Sponsored Search Conversion Log (CSSCL) dataset, τ = 10

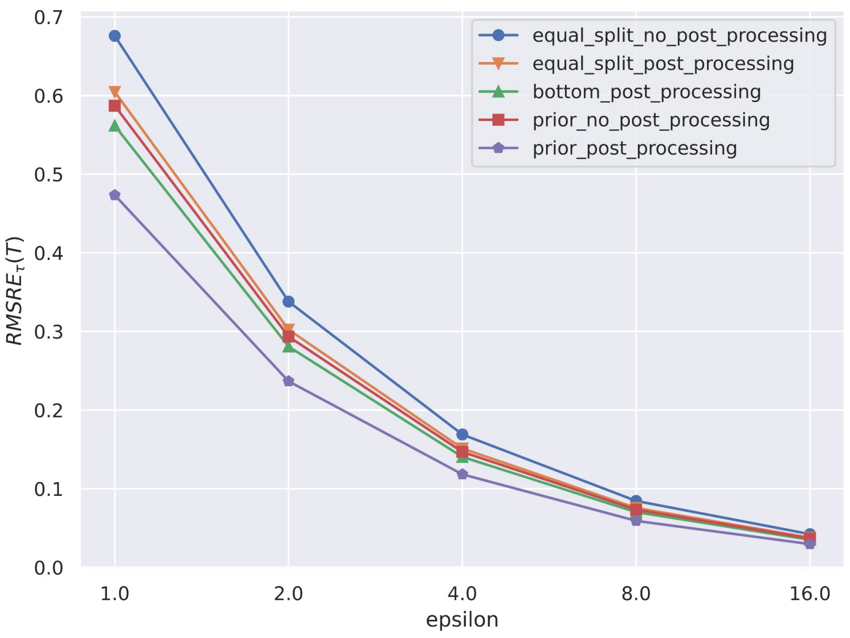


Four-attribute hierarchy using Criteo Attribution Modeling for Bidding (CAMB) dataset, τ = 10

Evaluation



Four-attribute hierarchy using Criteo Sponsored Search Conversion Log (CSSCL) dataset, τ = 10



Three-attribute hierarchy using Criteo Attribution Modeling for Bidding (CAMB) dataset, τ = 10