Rethink Possible

Structure-Aware Sampling on Data Streams

Edith Cohen, Graham Cormode, Nick Duffield

AT&T Labs-Research

Summarizing Data Streams

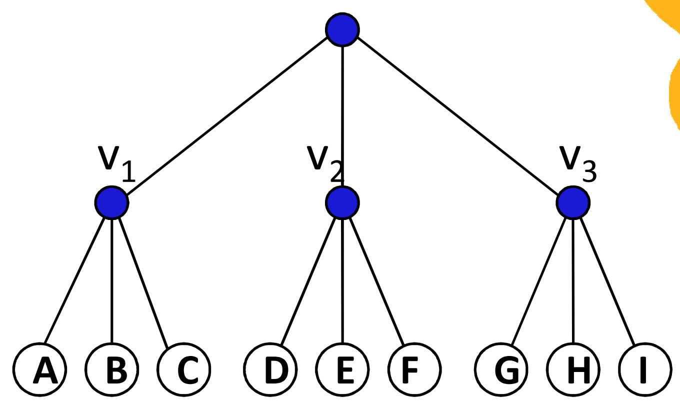
- Many applications generate streams of data
  - Core example: transient data in IP networks
  - Need to track aggregate statistics for later analysis
- State-of-the-art summarization via sampling
  - Widely deployed in current network elements
  - General purpose summary, enables subset-sum queries
  - Higher level analysis: quantiles, heavy hitters, other patterns & trends

Limitations of Sampling

- Current sampling methods are structure oblivious
  - But most queries are structure respecting!
- Most queries are actually range queries
  - "How much traffic from region X to region Y between 2am and 4am?"
- Much structure in data
  - Order (e.g. ordered timestamps, durations etc.)
  - Hierarchy (e.g. geographic and network hierarchies)
  - (Multidimensional) products of structures
- Can making sampling structure-aware improve accuracy?

Toy Example

- Two level hierarchy with 9 leaves
  - Draw a sample of size k=3
- Structure-aware sampling would pick only 1 item from each branch



- Uniform stream sampling picks each subset of 3 items uniformly
  - Probability of picking one from each branch = $3^3/9C3 = 9/28$
- There is an optimal structure aware sampling scheme that always picks one from each branch
  - Gives exact estimates for weight of {A,B,C} {D,E,F} and {G,H,I}
  - But not possible to sample from this distribution in the stream

Background on Stream Sampling

- Inclusion Probability Proportional to Size (IPPS):
  - Given parameter τ, probability of sampling key with weight w is $\min\{1, w/\tau\}$
  - Key i has adjusted weight $a_i = w_i/p_\tau(w_i) = \max\{\tau, w_i\}$ (Horvitz-Thompson)
  - Can pick a τ so that expected sample size is k
- Stream VarOpt sampling method is Variance Optimal on leaves:
  - Maintain a sample of size exactly k keys
  - Include key in the sample each new key i of weight w_i
  - Pivot: pick one key to eject, via IPPS
- Stream VarOpt is unique: no freedom to be structure aware
  - We must relax our requirements to allow structure to be used

Weight-bounded Summaries

- Generalize VarOpt summaries by relaxing requirements
- M-bounded summary stores k keys and adjusted weights a_i
  - Adjusted weights are unbiased: $E[a_i] = w_i$
  - Weights sum to correct value: $\sum_{i \in \text{sample}} a_i = \sum_{i \in \text{stream}} w_i$
  - Weights controlled by M: if $w_i \geq M$, then $a_i = w_i$, else $a_i \leq M$
  - Inclusion-exclusion bounds: for any subset of keys J and $N \geq \max_{i \in J} a_i$

    (Inclusion) $E[\prod_{i \in J} a_i] \leq \prod_{i \in J} w_i$

    (Exclusion) $E[\prod_{i \in J} (N-a_i)] \leq \prod_{i \in J} (N-w_i)$
- Intuition: VarOpt is M-bounded with M set to smallest value τ
  - So M-bounded shares many good properties of VarOpt

Candidate Set & Pivot Selection

- Given a set of $k+1$ keys and weights, pick 1 key to eject
  - Pivot selection: pick a subset of X based on structure conditions
  - Treat as an instance of Stream VarOpt on X
  - Compute a τ value for the set X
  - Candidate subset X: those keys with $a_i \leq \tau$
  - Pivot: pick one element of X to eject, via VarOpt algorithm
  - Adjust weights of other elements in X so they remain unbiased
- Structure awareness: pick X to be keys that are "close" in structure, so that shifting of "probability weight" is localized
  - Tradeoff optimality on leaves for better performance on ranges

Range Cost

- Many possible pivot sets X – how to choose?
- We define a "range cost" for each possible X
  - Measure as the local impact on the variance of adjusted weights
  - "Local": only the change caused at this step
  - Average over the impact on all range queries
- Pick the pivot set with least range cost
  - Many to consider, so may restrict to small sets, e.g. $|X|=2$

Range Cost for Order

- Range cost of X is weighted average of variance over prefixes
  - Prefixes, not all ranges simplifies analysis
  - Any range is difference of two prefixes
- Analyze the impact of a pivot on the distribution of weights
  - Measure weight crossing each quantile boundary
  - Range cost minimized by picking $|X| = 2$
- Given $X = \{i,j\}$, range cost is

  $$a_i\, a_j((a_i + a_j)/3 + \sum_{i < l < j} a_l)$$

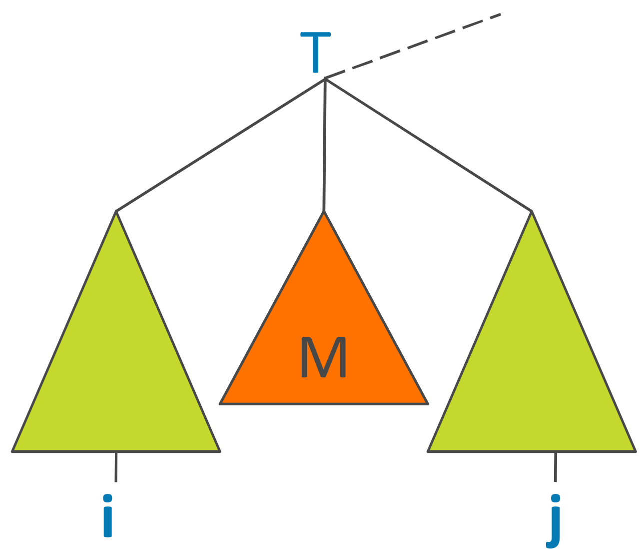
  - Given $\{i, j\}$, cost can be found in constant time
  - Takes $O(k)$ time to find best pair (using structure of minimal pairs)

Range Cost for Hierarchy

- Use result for order to analyze hierarchy
- Consider all possible linearizations of hierarchy to an order
- For $X = \{i,j\}$, the range cost is computed over subtree T of $\{i,j\}$

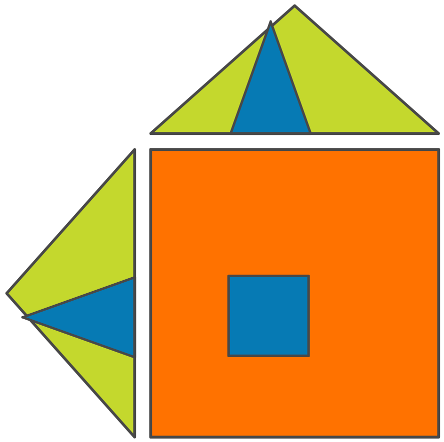
  $$a_i\, a_j \left(\left(\sum_{l \in T} a_l \right)/2 - \sum_{l \in M} a_l/6 \right)$$

  – where set M = leaves not in same subtree as i or j



Range Cost for Product Spaces

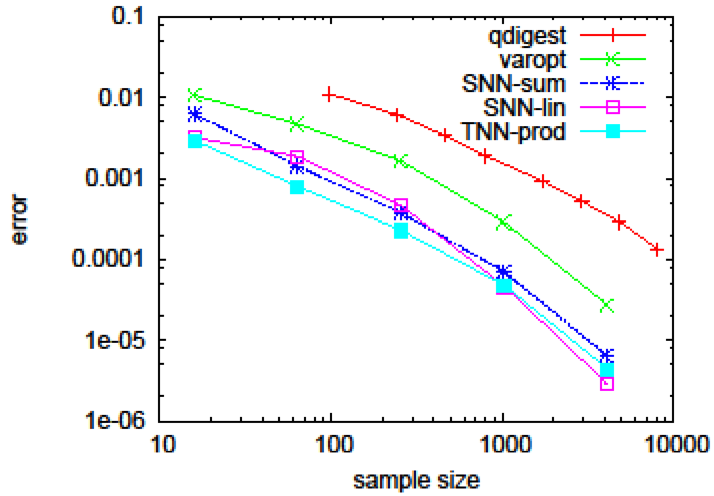
- Generalize range costs to products of ranges
  - Define as average of one-dimensional range costs
  - Equivalent to averaging over all axis-parallel halfspaces
  - Expressions for optimal range cost become complex
  - Fast heuristics are preferred

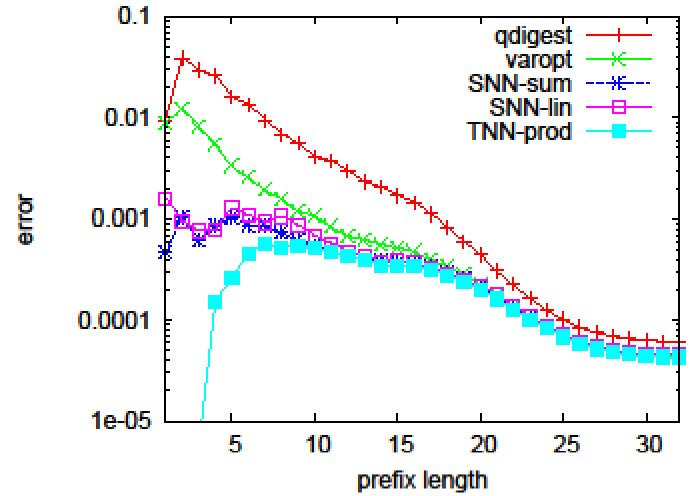


Fast Pivot Selection

- Analysis allows optimal range-aware pivot selection
  - Slow: even $|X|=2$ leads to $O(k^2)$ time, too slow for stream
- "Pair heuristics" let us pick a good pair quickly
  - SNN-Sum: each node paired with nearest neighbor under order
    - cost of a pair is sum of weights ($O(\log k)$) to maintain least)
  - SNN-Lin: same pairing, but cost is computed from 'hierarchy' case
  - TNN-Prod: nodes paired with min-weight hierarchy neighbor
    - cost of a pair is product of weights (slower to maintain)
  - VSNN (multi-D): use KD-tree to find near neighbor for each node
    - cost of a pair is product of weights ($O(\log k)$) time to maintain)
  - SpanApprox (multi-D): search over keys in same hierarchy (v slow)

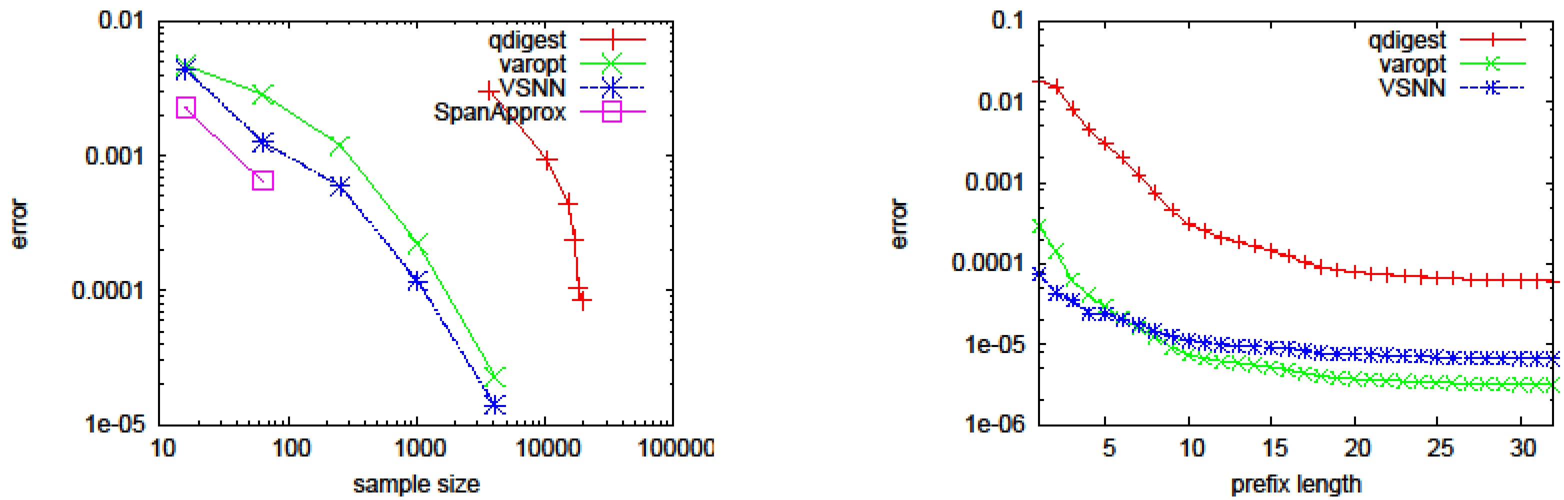
1-dimensional Experiments





- Measure mean relative error on queries over IP data flows
- Compare to oblivious VarOpt and deterministic qdigest
- Order of magnitude improvement for structure awareness
- Benefit decreases over longer prefixes
  - Becomes like arbitrary subset queries (can't beat VarOpt on leaves)

2-dimensional Experiments



- Benefit still clear, but less pronounced
- SpanApprox very promising, but too slow for large samples
- Crossover at 2D prefix length 8
  - Beyond this, queries touch 2^{-16} ~ 1e-5 fraction of area

Concluding Remarks

- Structure awareness in sampling can improve accuracy on common range queries
- Guarantee performance no worse than a smaller VarOpt sample
- Can work at streaming speed: $O(\log k)$ work per step
- Open problems:
  - More fast pair heuristics
  - Extend analysis for unaggregated streams
  - Maintain samples over distributed data