A Quick Introduction to Approximate Query Processing Part II

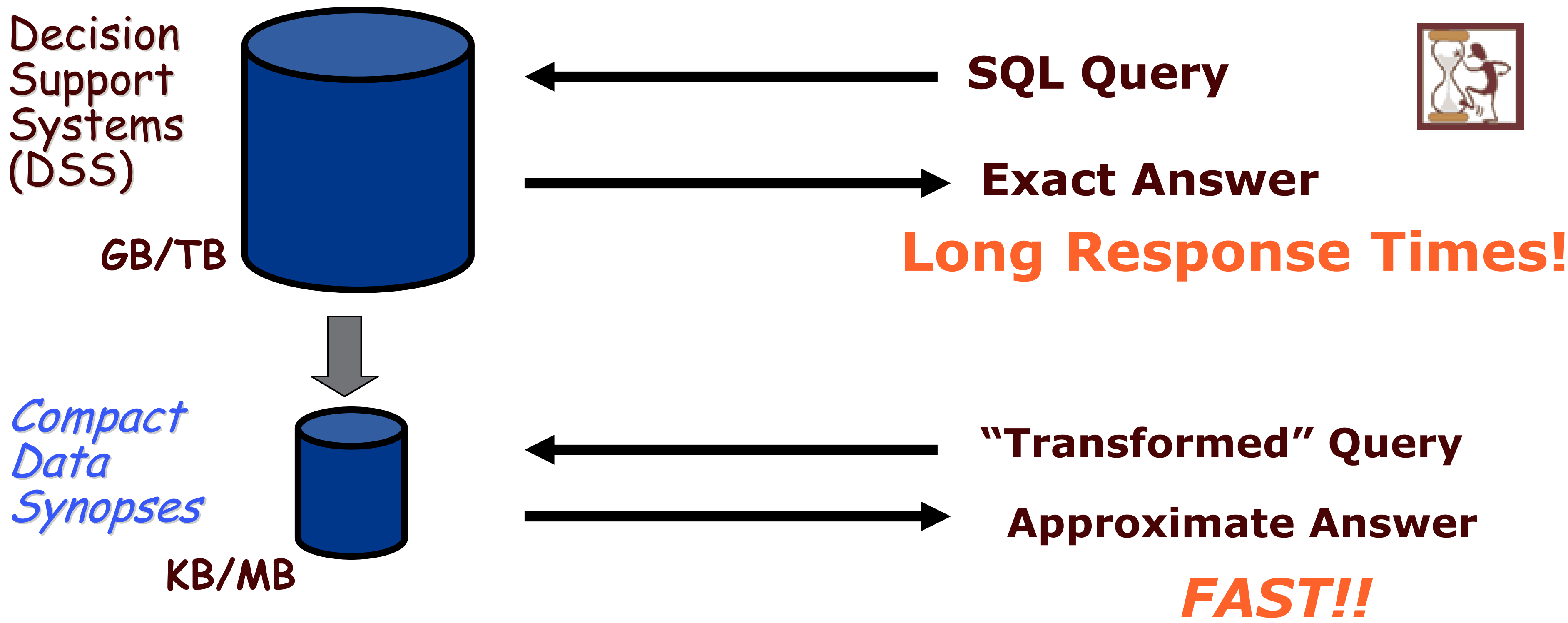
CS286, Spring'2007

Minos Garofalakis

Decision Support Systems

- Data Warehousing: Consolidate data from many sources in one large repository.
  - Loading, periodic synchronization of replicas.
  - Semantic integration.
- OLAP:
  - Complex SQL queries and views.
  - Queries based on spreadsheet-style operations and "multidimensional" view of data.
  - Interactive and "online" queries.
- Data Mining:
  - Exploratory search for interesting trends and anomalies. (Another lecture!)

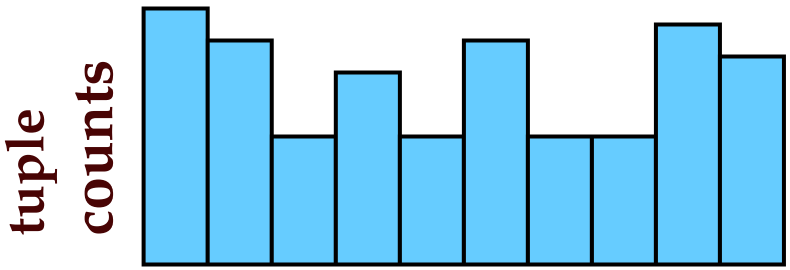
Approximate Query Processing using Data Synopses



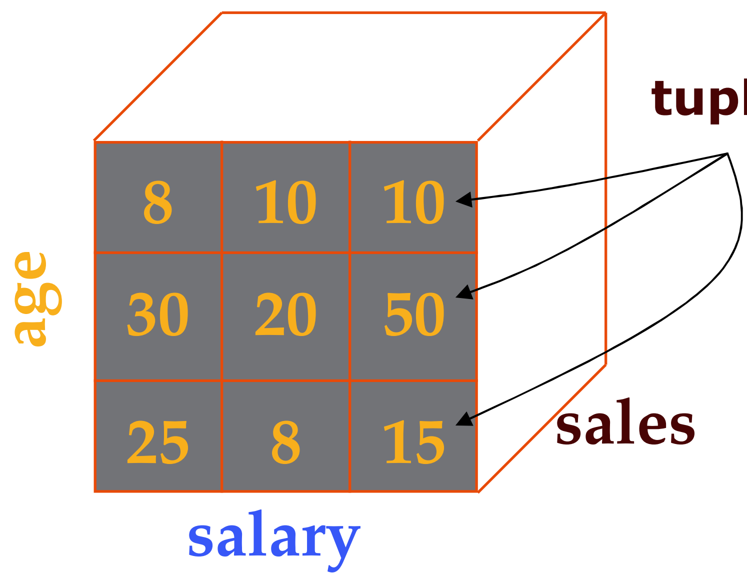
- How to construct effective *data synopses* ??

Relations as Frequency Distributions

One-dimensional distribution



Three-dimensional distribution



name	age	salary	sales
MG	34	100K	25K
JG	33	90K	30K
RR	40	190K	55K
JH	36	110K	45K
MF	39	150K	50K
DD	45	150K	50K
JN	43	140K	45K
AP	32	70K	20K
EM	24	50K	18K
DW	24	50K	28K

Outline

- Intro & Approximate Query Answering Overview
  - Synopses, System architectures, Commercial offerings
- One-Dimensional Synopses
  - **Histograms:** Equi-depth, Compressed, V-optimal, Incremental maintenance, Self-tuning
  - **Samples:** Basics, Sampling from DBs, Reservoir Sampling
  - **Wavelets:** 1-D Haar-wavelet histogram construction & maintenance
- Multi-Dimensional Synopses and Joins
- Set-Valued Queries
- Discussion & Comparisons
- Advanced Techniques & Future Directions

One-Dimensional Haar Wavelets

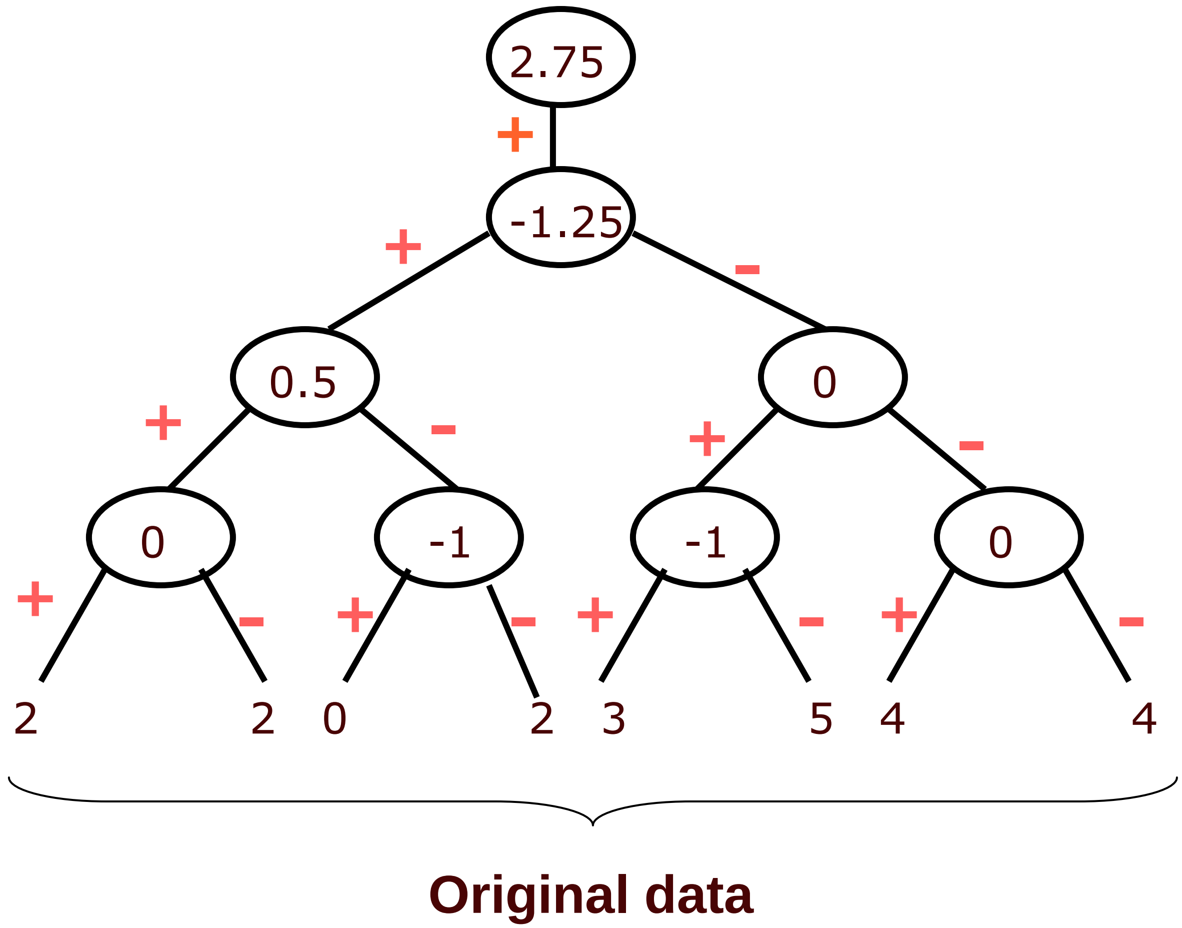
- Wavelets: mathematical tool for hierarchical decomposition of functions/signals
- Haar wavelets: simplest wavelet basis, easy to understand and implement
  - *Recursive pairwise averaging and differencing* at different resolutions

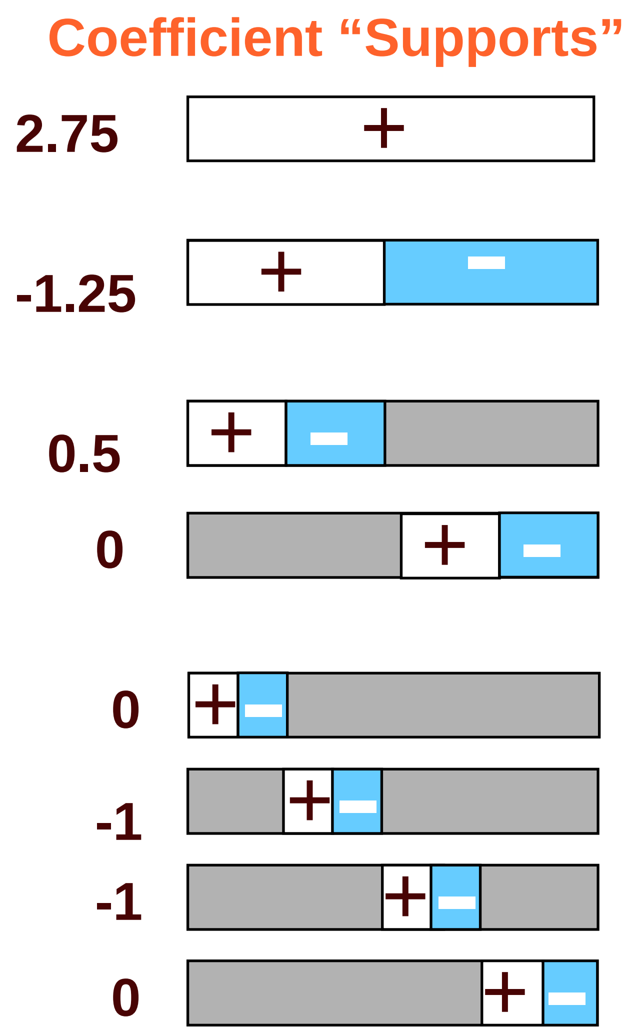
Resolution	Averages	Detail Coefficients
3	[2, 2, 0, 2, 3, 5, 4, 4]	----
2	[2, 1, 4, 4]	[0, -1, -1, 0]
1	[1.5, 4]	[0.5, 0]
0	[2.75]	[-1.25]

Haar wavelet decomposition: [2.75, -1.25, 0.5, 0, 0, -1, -1, 0]

Haar Wavelet Coefficients

- Hierarchical decomposition structure (a.k.a. "error tree")



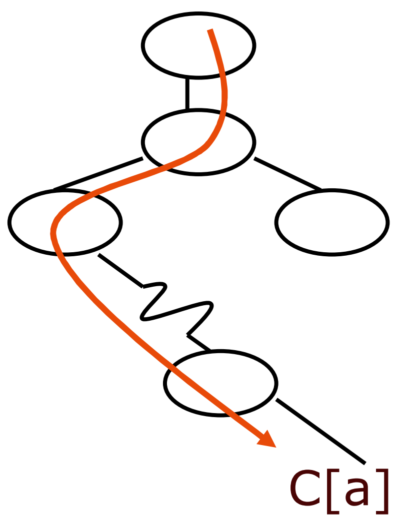


Wavelet-based Histograms [MVW98]

- Problem: range-query selectivity estimation
- Key idea: use a compact subset of Haar/linear wavelet coefficients for approximating the data distribution
- Steps
  - compute (cumulative) data distribution *C*
  - compute Haar (or linear) wavelet transform of *C*
  - coefficient *thresholding* : only b<<|C| coefficients can be kept
    - take largest coefficients in *absolute normalized value*
      - Haar basis: divide coefficients at resolution j by $\sqrt{2^j}$
      - *Optimal* in terms of the overall Mean Squared (L2) Error
    - Greedy heuristic methods
      - Retain coefficients leading to large error reduction
      - Throw away coefficients that give small increase in error

Using Wavelet-based Histograms

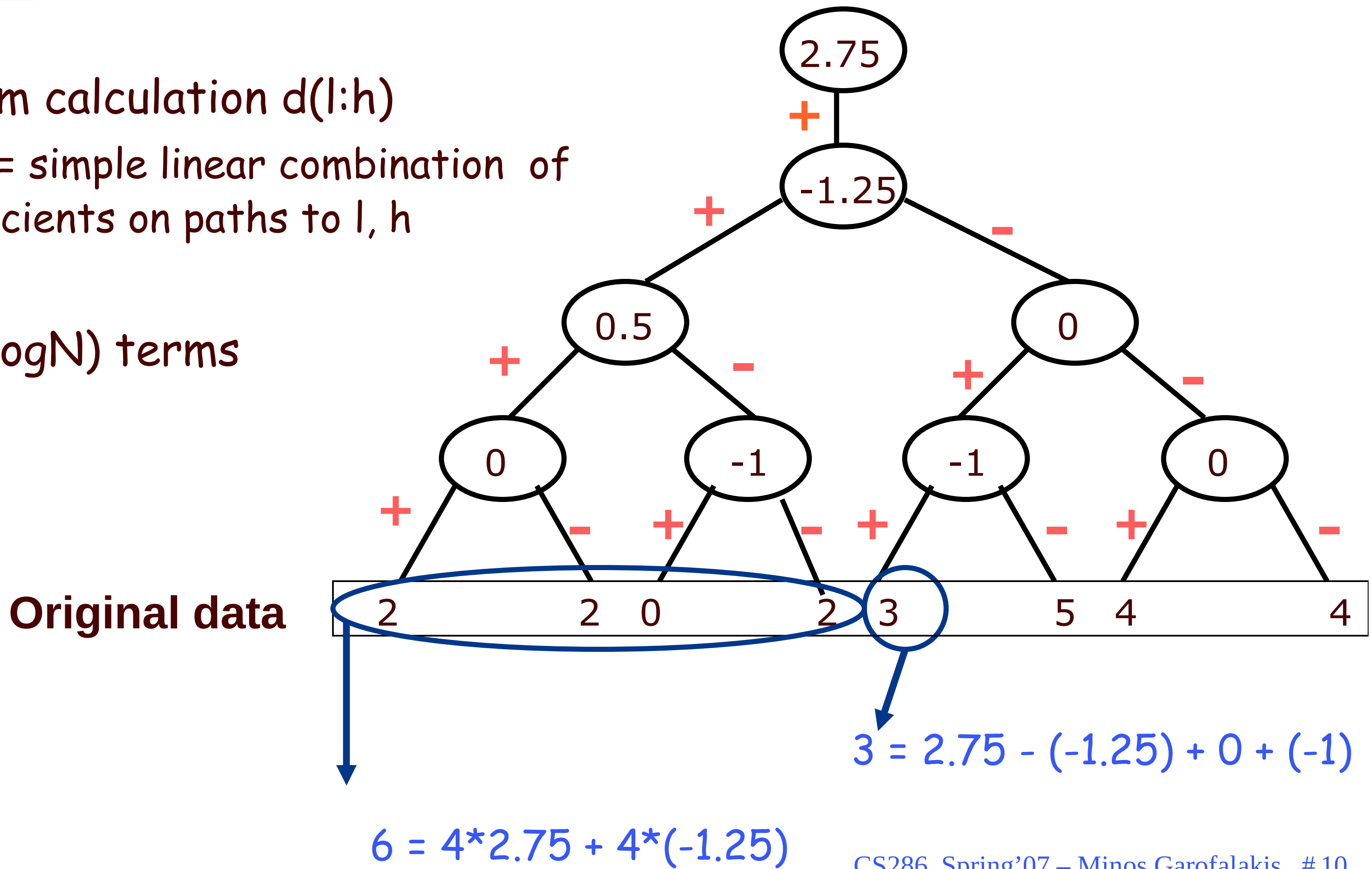
- Selectivity estimation: sel(a<= X<= b) = C'[b] - C'[a-1]
  - C' is the (approximate) "reconstructed" cumulative distribution
  - Time: O(min{b, logN}), where b = size of wavelet synopsis (no. of coefficients), N= size of domain



- At most logN+1 coefficients are needed to reconstruct any C value

Haar Wavelet Coefficients

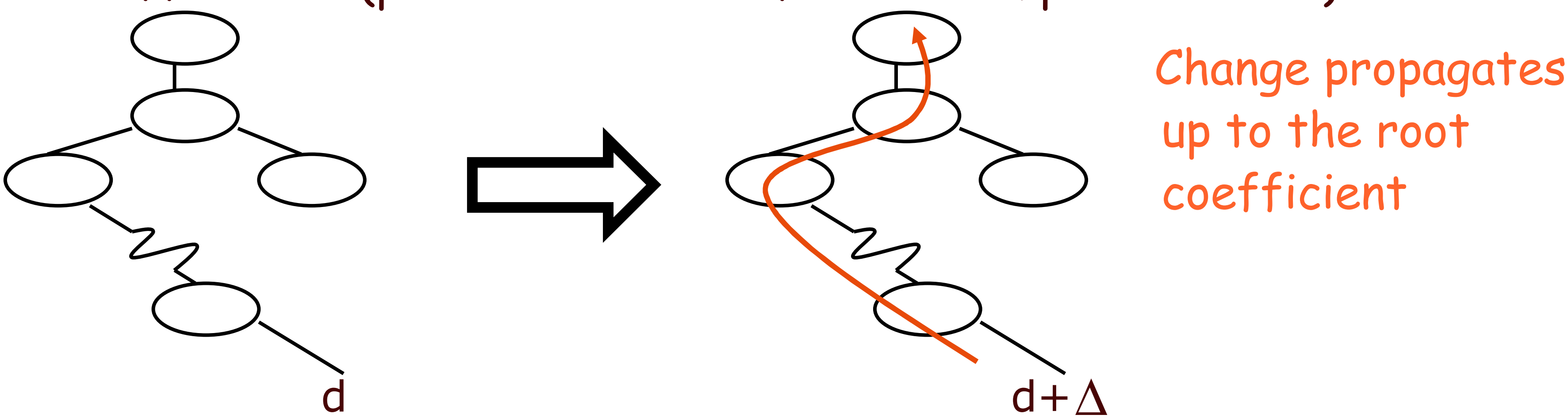
- Reconstruct data values d(i)
  - $d(i) = \sum (+/-1) * (\text{coefficient on path})$
- Range sum calculation d(l:h)
  - d(l:h) = simple linear combination of coefficients on paths to l, h
- Only O(logN) terms



Dynamic Maintenance of Wavelet-based Histograms [MVW00]

- Build Haar-wavelet synopses on the original data distribution

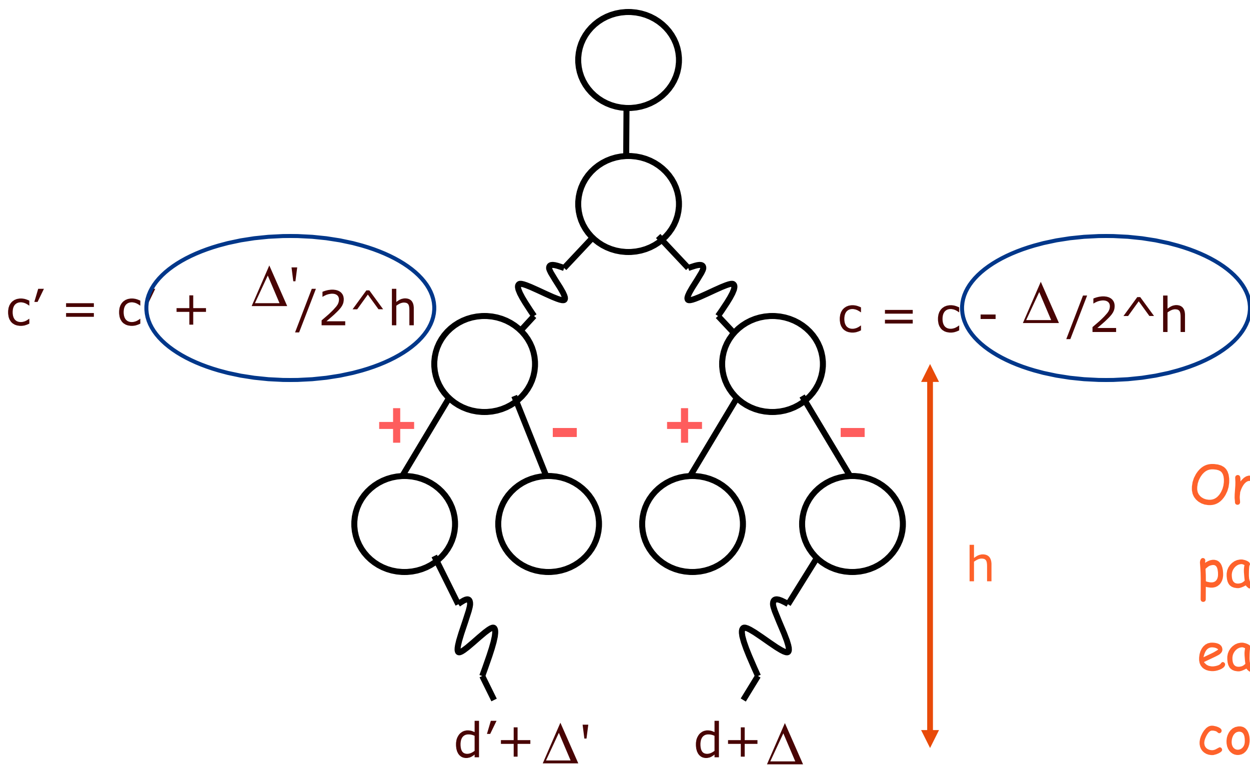
- Key issues with dynamic wavelet maintenance
  - Change in single distribution value can affect the values of many coefficients (path to the root of the decomposition tree)



  - As distribution changes, "most significant" (e.g., largest) coefficients can also change!
    - Important coefficients can become unimportant, and vice-versa

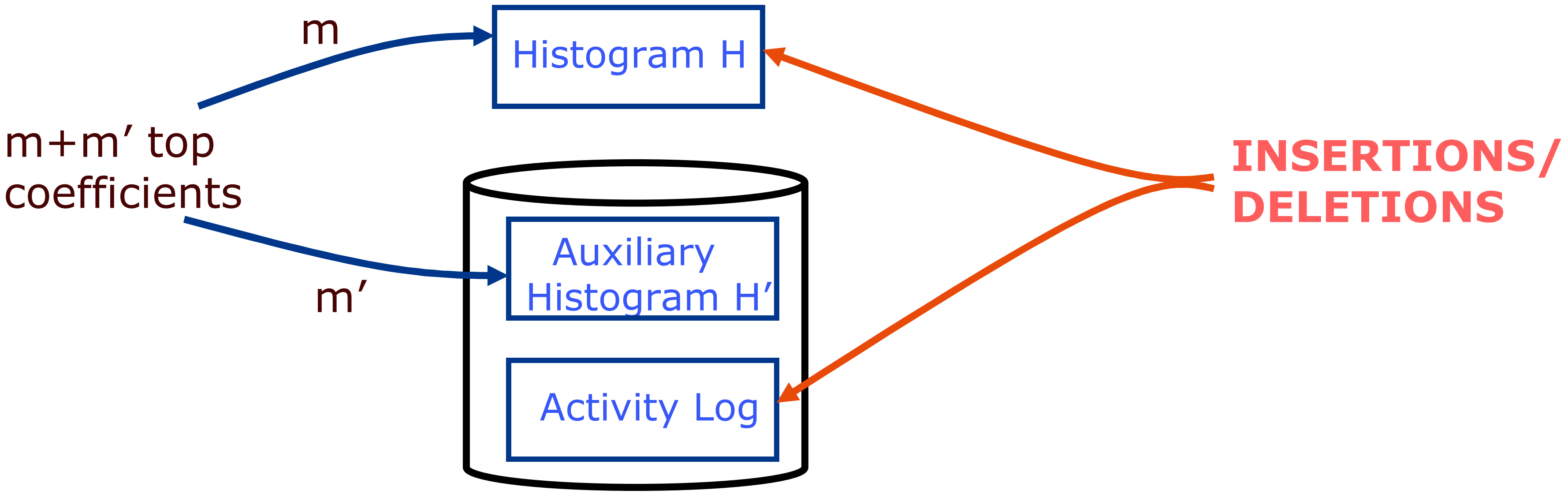
Effect of Distribution Updates

- Key observation: for each coefficient c in the Haar decomposition tree
  - c = (AVG(leftChildSubtree(c)) - AVG(rightChildSubtree(c))) / 2



Only coefficients on path(d) are affected and each can be updated in constant time

Maintenance Architecture



- "Shake up" when log reaches max size: for each insertion at d
  - for each coefficient c on path(d) and in H' : update c
  - for each coefficient c on path(d) and not in H or H':
    - insert c into H' with probability proportional to 1/2^h, where h is the "height" of c (*Probabilistic Counting* [FM85])
  - Adjust H and H' (move largest coefficients to H)

Problems with Conventional Wavelet Synopses

- An example data vector and wavelet synopsis (|D|=16, B=8 largest coefficients retained)

Over 2,000% relative error! — Always accurate!

Original Data Values	127	71	87	31	59	3	43	99	100	42	0	58	30	88	72	130
Wavelet Answers	65	65	65	65	65	65	65	65	100	42	0	58	30	88	72	130

Estimate = 195, actual values: d(0:2)=285, d(3:5)=93!

- Large variation in answer quality
  - Within the same data set, when synopsis is *large,* when data values are about the same, when actual answers are about the same
  - Heavily-biased approximate answers!
- Root causes
  - Thresholding for aggregate L2 error metric
  - Independent, greedy thresholding (⇨ large regions without any coefficient!)
  - Heavy bias from dropping coefficients without compensating for loss

Approach: Optimize for Maximum-Error Metrics

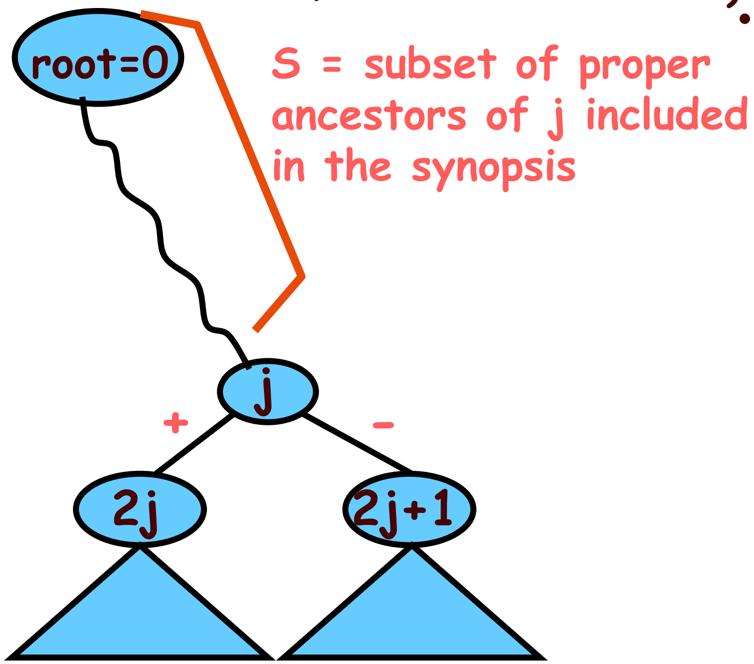
- Key metric for effective approximate answers: *Relative error with sanity bound* $\frac{|\hat{d}_i - d_i|}{\max\{|d_i|, s\}}$
  - Sanity bound "s" to avoid domination by small data values
- To provide tight error guarantees for *all* reconstructed data values

Minimize $\max_i \{\frac{|\hat{d}_i - d_i|}{\max\{|d_i|, s\}}\}$

  - Minimize *maximum relative error* in the data reconstruction
- Another option: Minimize *maximum absolute error* $\max_i \{|\hat{d}_i - d_i|\}$
- Algorithms can be extended to general *"distributive" metrics* (e.g., average relative error)

Our Approach: Deterministic Wavelet Thresholding for Maximum Error

- ***Key Idea:*** Dynamic-Programming formulation that *conditions the optimal solution on the error that "enters" the subtree* (through the selection of ancestor nodes)



- Our DP table:

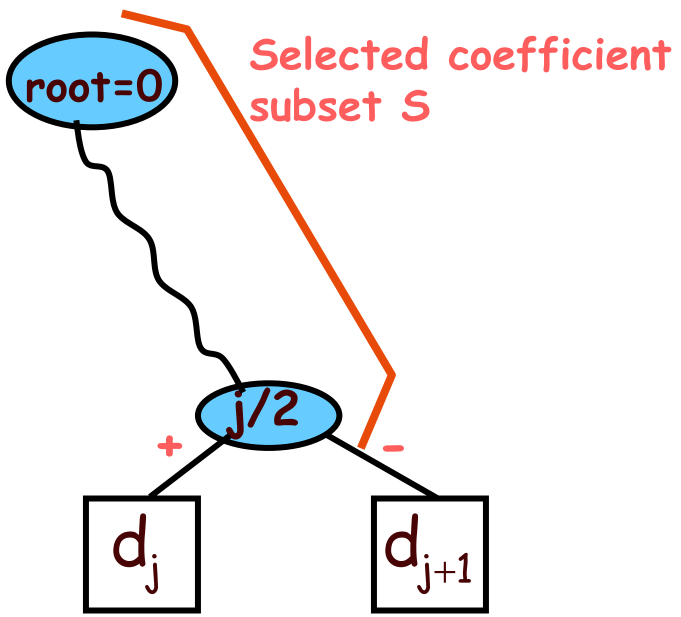
  M[j, b, S] = optimal maximum relative (or, absolute) error in T(j) with space budget of b coefficients (chosen in T(j)), *assuming subset S of j's proper ancestors have already been selected for the synopsis*

  - Clearly, $|S| \leq \min\{B-b, \log N+1\}$
  - Want to compute $M[0, B, \phi]$

- ***Basic Observation:*** Depth of the error tree is only logN+1 ⟹ we can explore and tabulate all S-subsets for a given node at a space/time cost of only O(N) !

Base Case for DP Recurrence: Leaf (Data) Nodes

- Base case in the bottom-up DP computation: Leaf (i.e., data) node d_j
  - Assume for simplicity that data values are numbered N, …, 2N-1



- M[j, b, S] is not defined for b>0
  - Never allocate space to leaves
- For b=0

$$M[j,0,S] = \frac{|d_j - \sum_{c \in S} sign(c,d_j) \cdot c|}{\max\{|d_j|, s\}}$$

for each coefficient subset $S \subseteq path(d_j)$ with $|S| \leq \min\{B, \log N+1\}$

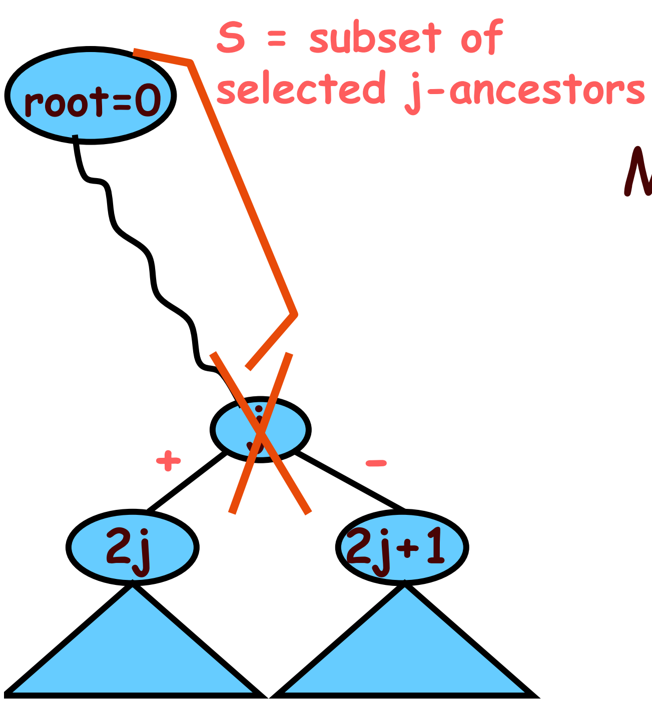
  - *Similarly for absolute error*

- Again, time/space complexity per leaf node is only O(N)

DP Recurrence: Internal (Coefficient) Nodes

- Two basic cases when examining node/coefficient j for inclusion in the synopsis: (1) Drop j; (2) Keep j

Case (1): Drop Coefficient j



- In this case, the minimum possible maximum relative error in T(j) is

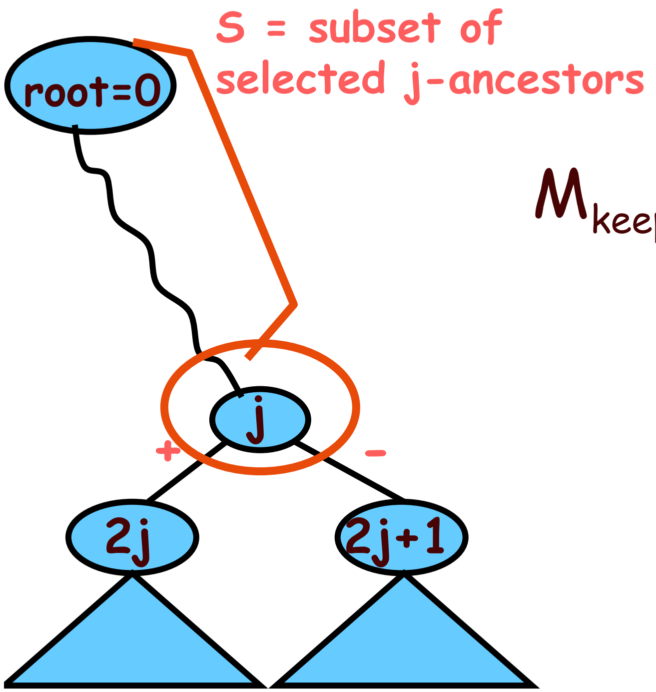
$$M_{drop}[j,b,S] = \min_{0 \le b' \le b} \max\{ M[2j,b',S], M[2j+1,b-b',S] \}$$

  - Optimally distribute space b between j's two child subtrees

- Note that the RHS of the recurrence is well-defined
  - Ancestors of j are obviously ancestors of 2j and 2j+1

DP Recurrence: Internal (Coefficient) Nodes (cont.)

Case (2): Keep Coefficient j



- In this case, the minimum possible maximum relative error in T(j) is

$$M_{keep}[j,b,S] = \min_{0 \le b' \le b-1} \max\{ M[2j,b',S \cup \{c_j\}],$$
$$M[2j+1, b-b'-1, S \cup \{c_j\}] \}$$

  - Take 1 unit of space for coefficient j, and optimally distribute remaining space
  - Selected subsets in RHS change, since we choose to retain j

- Again, the recurrence RHS is well-defined

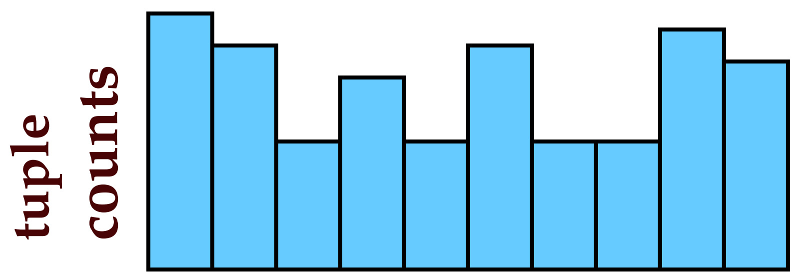
- Finally, define $M[j,b,S] = \min\{ M_{drop}[j,b,S], M_{keep}[j,b,S] \}$
- Overall complexity: $O(N^2)$ time, $O(N \min\{B, \log N\})$ space

Outline

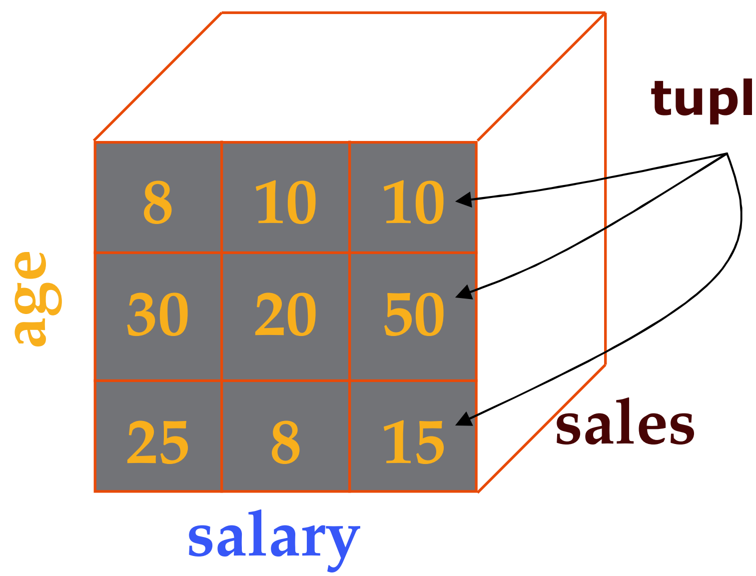
- Intro & Approximate Query Answering Overview
- One-Dimensional Synopses
- Multi-Dimensional Synopses and Joins
  - Multi-dimensional Histograms
  - Join sampling
  - Multi-dimensional Haar Wavelets
- Set-Valued Queries
- Discussion & Comparisons
- Advanced Techniques & Future Directions
- Conclusions

Relations as Frequency Distributions

One-dimensional distribution



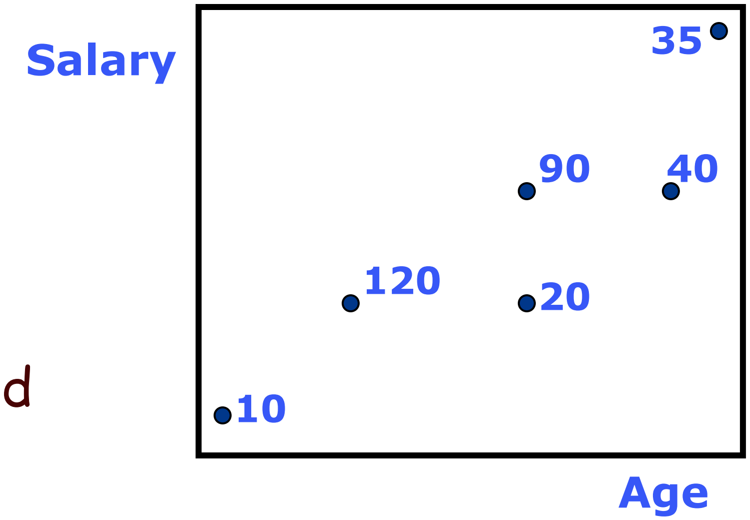
Three-dimensional distribution



name	age	salary	sales
MG	34	100K	25K
JG	33	90K	30K
RR	40	190K	55K
JH	36	110K	45K
MF	39	150K	50K
DD	45	150K	50K
JN	43	140K	45K
AP	32	70K	20K
EM	24	50K	18K
DW	24	50K	28K

Multi-dimensional Data Synopses

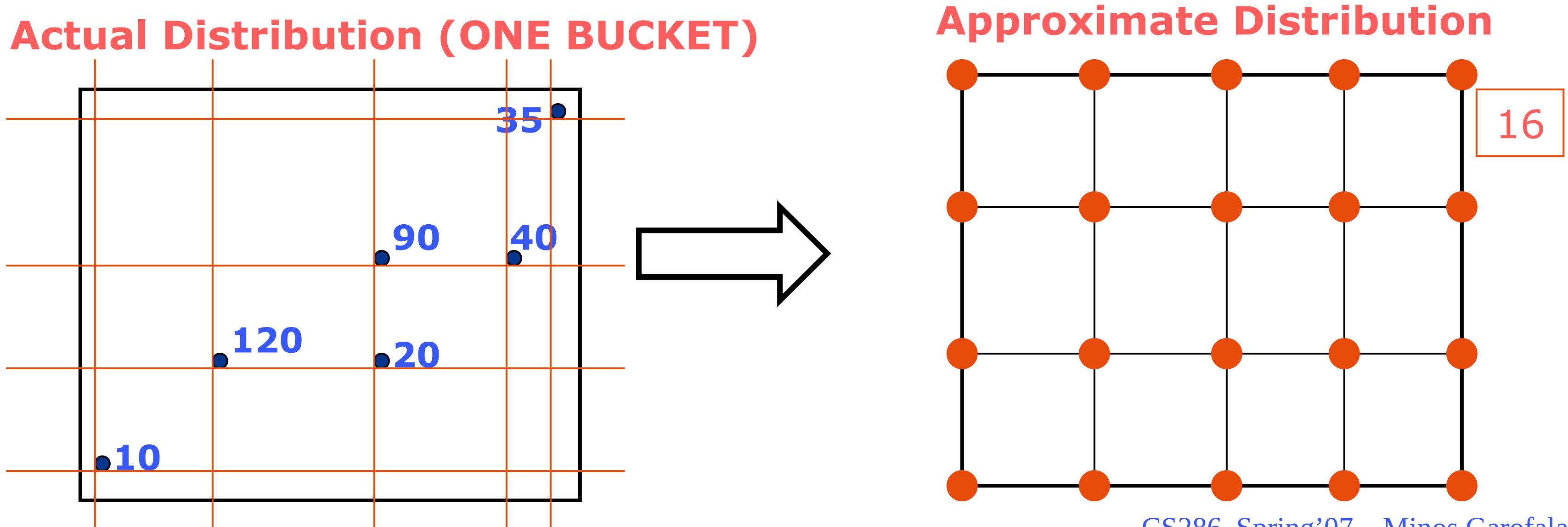
- Problem: Approximate the *joint data distribution* of multiple attributes
  - Motivation
    - Selectivity estimation for queries with multiple predicates
    - Approximating OLAP data cubes and general relations



- Conventional approach: Attribute-Value Independence (AVI) assumption
  - sel(p(A1) & p(A2) & . . .) = sel(p(A1)) * sel(p(A2) * . . .
  - Simple -- one-dimensional marginals suffice
  - BUT: almost always inaccurate, gross errors in practice (e.g., [Chr84, FK97, Poo97]

Multi-dimensional Histograms

- Use small number of multi-dimensional buckets to *directly* approximate the joint data distribution
- Uniform spread & frequency approximation within buckets
  - n(i) = no. of distinct values along Ai, F = total bucket frequency
  - approximate data points on a n(1)*n(2)*. . . uniform grid, each with frequency F / (n(1)*n(2)*. . .)

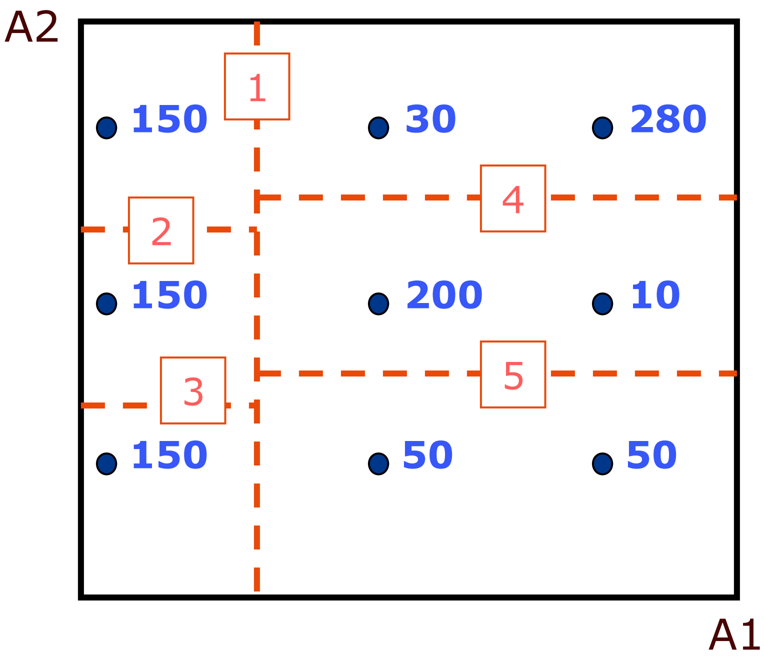


Multi-dimensional Histogram Construction

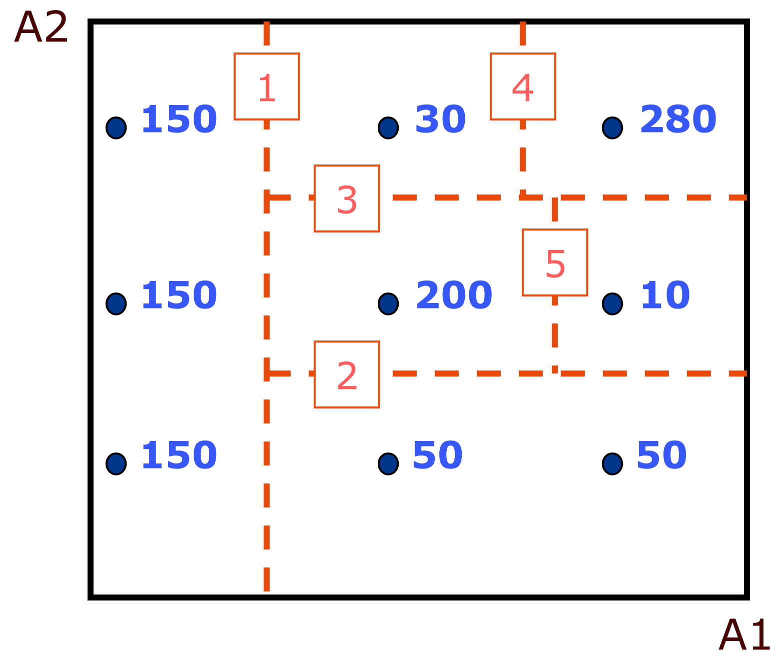
- Construction problem is much harder even for two dimensions [MPS99]
- *Multi-dimensional equi-depth histograms* [MD88]
  - Fix an ordering of the dimensions A1, A2, . . ., Ak, let $\alpha \approx$kth root of desired no. of buckets, initialize B = { data distribution }
  - For i=1, . . ., k: Split each bucket in B in α equi-depth partitions along Ai; return resulting buckets to B
  - Problems: limited set of bucketizations; fixed α and fixed dimension ordering can result in poor partitionings
- *MHIST-p histograms* [PI97]
  - At each step
    - Choose the bucket b in B containing the attribute Ai whose marginal *is the most in need of partitioning*
    - Split b along Ai into p (e.g., p=2) buckets

Equi-depth vs. MHIST Histograms

Equi-depth (a1=2,a2=3) [MD88]



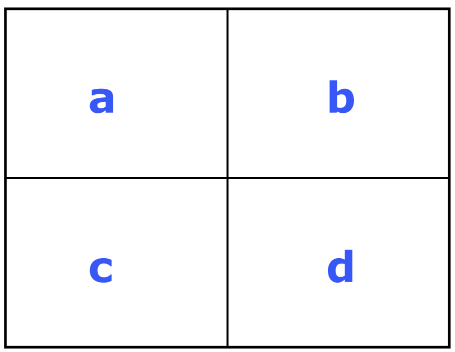
MHIST-2 (MaxDiff) [PI97]

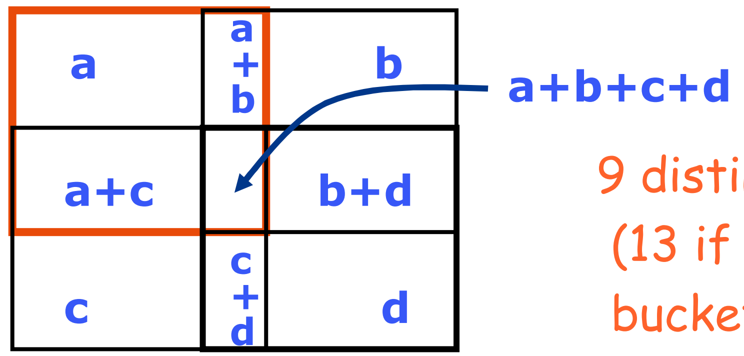


- MHIST: choose bucket/dimension to split based on its *criticality* ; allows for much larger class of bucketizations (*hierarchical* space partitioning)
- Experimental results verify superiority over AVI and equi-depth

Other Multi-dimensional Histogram Techniques -- GENHIST [GKT00]

- Key idea: allow for *overlapping* histogram buckets
  - Allows for a much larger no. of distinct frequency regions for a given space budget (= #buckets)



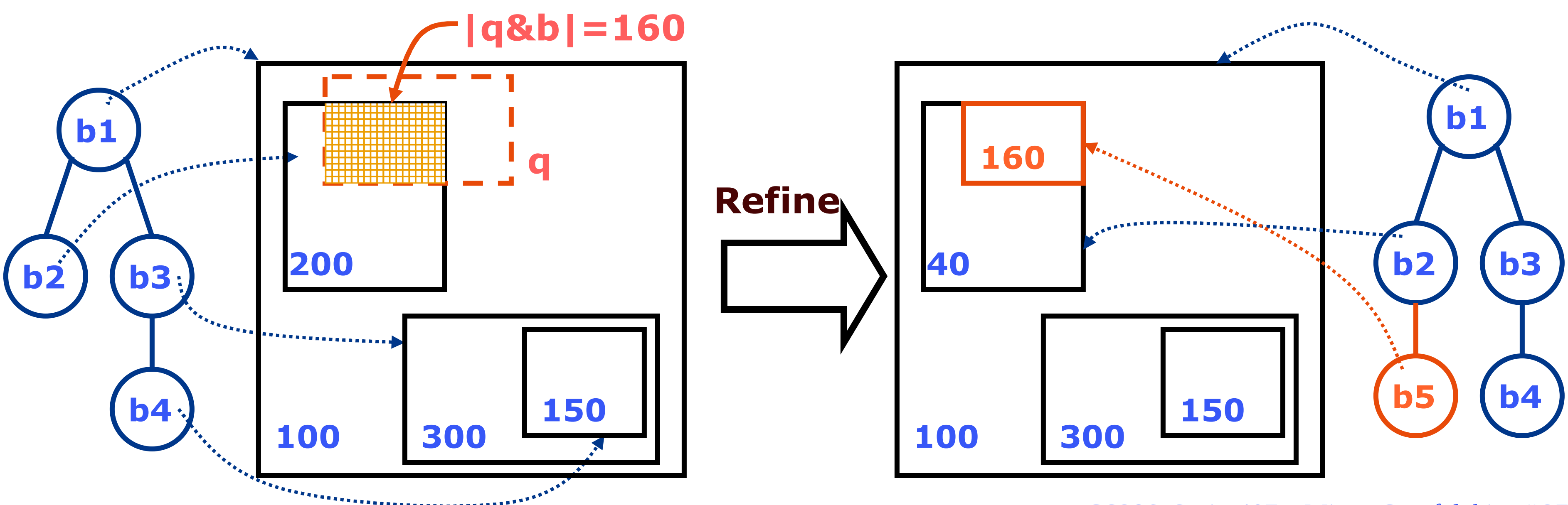


9 distinct frequencies (13 if different-size buckets are used)

- Greedy construction algorithm: Consider increasingly-coarser grids
  - At each step select the cell(s) c of highest density and move enough randomly-selected points from c into a bucket to make c and its neighbors "close-to-uniform"
  - *Truly multi-dimensional* "split decisions" based on *tuple density* -- unlike MHIST

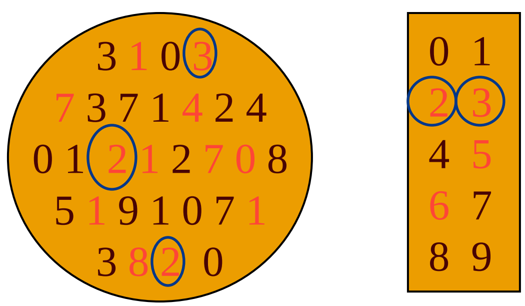
Other Multi-dimensional Histogram Techniques -- STHoles [BCG01]

- Multi-dimensional, workload-based histograms
  - Allow *bucket nesting* -- "bucket tree"
  - Intercept query result stream and count $|q \cap b|$ for each bucket b (< 10% overhead in MS SQL Server 2000)
  - Drill "holes" in b for regions of different *tuple density* and "pull" them out as children of b (first-class buckets)
  - Consolidate/merge buckets of similar densities (keep #buckets constant)



Sampling for Multi-D Synopses

- Taking a sample of the rows of a table captures the attribute correlations in those rows
  - Answers are unbiased & confidence intervals apply
  - Thus guaranteed accuracy for count, sum, and average queries on single tables, as long as the query is not too selective
- Problem with joins [AGP99,CMN99]:
  - Join of two uniform samples is not a uniform sample of the join
  - Join of two samples typically has very few tuples



Foreign Key Join
40% Samples in Red
Size of Actual Join = 30
Size of Join of samples = 3

Join Synopses for Foreign-Key Joins [AGP99]

- Based on sampling from materialized foreign key joins
  - Typically < 10% added space required
  - Yet, can be used to get a uniform sample of ANY foreign key join
  - Plus, fast to incrementally maintain
- Significant improvement over using just table samples
  - E.g., for TPC-H query Q5 (4 way join)
    - 1%-6% relative error vs. 25%-75% relative error, for synopsis size = 1.5%, selectivity ranging from 2% to 10%
    - 10% vs. 100% (no answer!) error, for size = 0.5%, select. = 3%