A Quick Introduction to Approximate Query Processing Part-IV

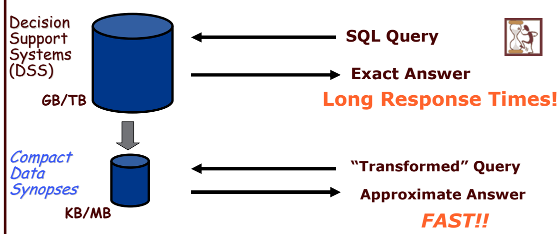
CS286, Spring'2007

Minos Garofalakis

Logistics…

- Draft CS286 web site is finally up!
  - http://db.cs.berkeley.edu/cs286sp07/
- Project list and guidelines being worked on
  - Please email me & Raghu to discuss your own project ideas…

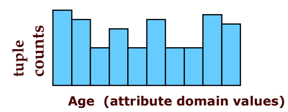
Approximate Query Processing using Data Synopses



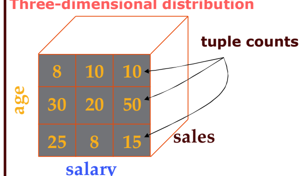
- How to construct effective *data synopses* ??

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

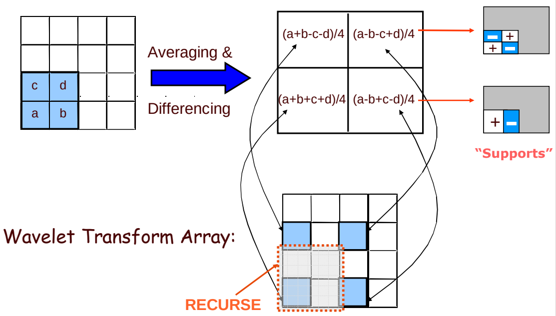
Outline

- 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

Outline

- Intro & Approximate Query Answering Overview
  - Synopses, System architecture, Commercial offerings
- One-Dimensional Synopses
  - Histograms, Samples, Wavelets
- Multi-Dimensional Synopses and Joins
  - Multi-D Histograms, Join synopses, Wavelets
- Set-Valued Queries
  - Error metrics; Using Histograms, Samples, Wavelets
- Discussion & Comparisons
- Advanced Techniques & Future Directions
  - Dependency-based, Streaming data

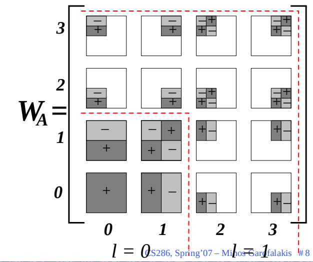
Two-dimensional Haar Wavelets -- Non-standard decomposition



Multi-dimensional Haar Wavelets

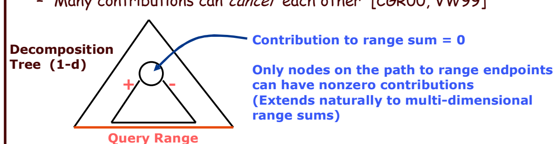
- Haar decomposition in d dimensions = d-dimensional array of wavelet coefficients
  - Coefficient *support region* = d-dimensional rectangle of cells in the original data array
  - *Sign* of coefficient's contribution can vary along the quadrants of its support

Support regions & signs for the 16 nonstandard 2-dimensional Haar coefficients of a 4X4 data array A



Range-sum Estimation Using Wavelet Synopses

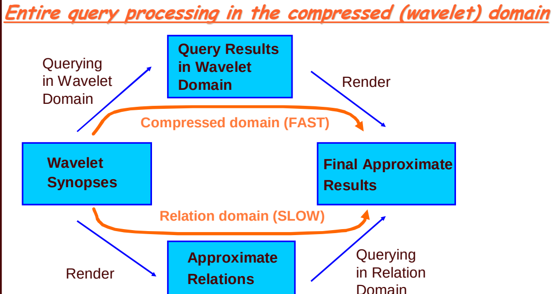
- Coefficient thresholding
  - As in 1-d case, normalizing by appropriate constants and retaining the largest coefficients minimizes the overall L2 error
- Range-sums: selectivity estimation or OLAP-cube aggregates [VW99] ("measure attribute" as count)
- Only coefficients with support regions intersecting the query hyper-rectangle can contribute
  - Many contributions can *cancel* each other [CGR00, VW99]



Approximate Query Processing Using Wavelets [CGR00]

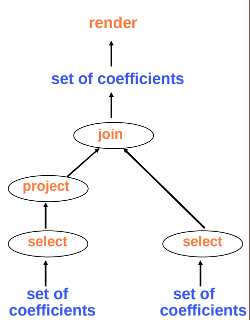
- Reduce relations into compact *wavelet-coefficient synopses*

Entire query processing in the compressed (wavelet) domain



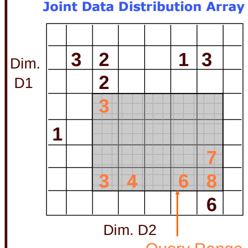
Wavelet Query Processing

- Each operator (e.g., select, project, join, aggregates, etc.)
  - *input:* set of wavelet coefficients
  - *output:* set of wavelet coefficients
- Finally, rendering step
  - *input:* set of wavelet coefficients
  - *output:* (multi)set of tuples



Selection -- Relational Domain

Joint Data Distribution Array

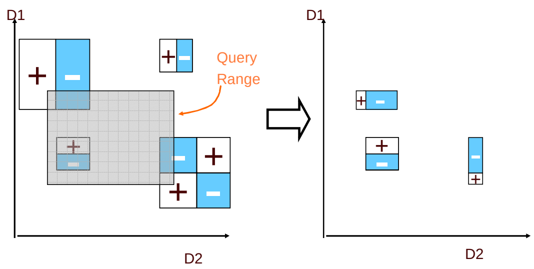


Relation

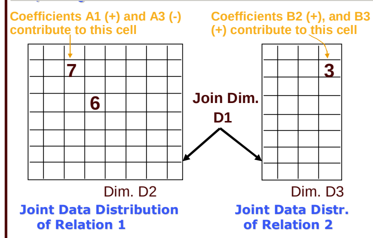
Dim D1 (Attr1)	Dim D2 (Attr2)	Count
0	6	6
1	2	3
1	3	4
1	5	6
1	6	8
2	6	7
3	0	1
4	2	3
5	2	2
6	1	3
6	2	2
6	5	1
6	6	3

- In relational domain, interested in only those cells inside query range
- In wavelet domain, interested in only the coefficients that contribute to those cells

Selection -- Wavelet Domain



Equi-join -- Relational Domain



Relation 1

Dim D1 (Attr1)	Dim D2 (Attr2)	Count
6	2	7
4	3	6

Relation 2

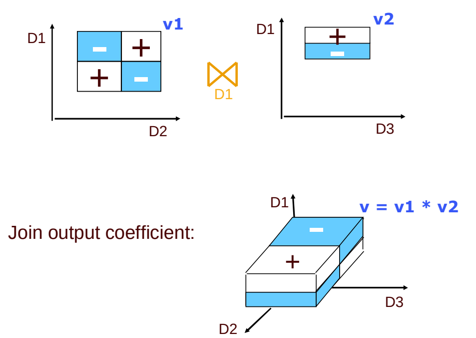
Dim D1 (Attr1)	Dim D3 (Attr3)	Count
6	3	3

Join along D1

Dim D1 (Attr1)	Dim D2 (Attr2)	Dim D3 (Attr3)	Count
6	2	3	21

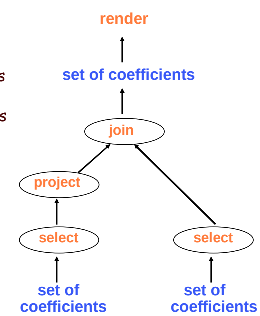
- *Relational domain:* Join count= 7*3 = (A1-A3)*(B2+B3)
- *Wavelet domain:* A1*B2 + A1*B3 - A3*B2 - A3*B3
- Consider all pairs of coefficients: (1) check joinability (overlap in join dimension(s)), (2) compute output coefficients

Equi-join -- Wavelet Domain



Wavelet Query Processing

- Each operator (e.g., select, project, join, aggregates, etc.)
  - *input:* set of wavelet coefficients
  - *output:* set of wavelet coefficients
- Finally, rendering step
  - *input:* set of wavelet coefficients
  - *output:* (multi)set of tuples



Outline

- Intro & Approximate Query Answering Overview
- One-Dimensional Synopses
- Multi-Dimensional Synopses and Joins
- Set-Valued Queries
- Discussion & Comparisons
- Advanced Techniques & Future Directions
- Conclusions

Discussion & Comparisons (1)

- *Histograms & Wavelets:* Limited by "curse of dimensionality"
  - Rely on data space partitioning in "regions"
  - Ineffective above 5-6 dimensions
    - Value/frequency uniformity assumptions within buckets break down in medium-to-high dimensionalities!!
- *Sampling:* No such limitations, BUT...
  - Ineffective for ad-hoc relational joins over arbitrary schemas
    - Uniformity property is lost
    - Quality guarantees degrade
  - Effectiveness for *set-valued* approximate queries is unclear
    - Only (very) small subsets of the answer set are returned (especially, when joins are present)

Discussion & Comparisons (2)

- *Histograms & Wavelets:* Compress data by accurately capturing rectangular "regions" in the data space
  - Advantage over sampling for typical, "range-based" relational DB queries
  - BUT, unclear how to effectively handle unordered/non-numeric data sets (no such issues with sampling...)
- *Sampling:* Provides strong probabilistic quality guarantees (unbiased answers) for individual aggregate queries
  - *Histograms & Wavelets:* Can guarantee a bound on the overall error (e.g., L2) for the approximation, BUT answers to individual queries can be heavily biased!!

No clear winner exists!! (Hybrids??)

Outline

- Intro & Approximate Query Answering Overview
- One-Dimensional Synopses
- Multi-Dimensional Synopses and Joins
- Set-Valued Queries
- Discussion & Comparisons
- Advanced Techniques & Future Directions
  - Dependency-based Synopses
  - Streaming Data
  - XML Synopses
- Conclusions

Dependency-based Histogram Synopses [DGR01]

Attribute Value Independence
**** simplistic***
**** inaccurate***

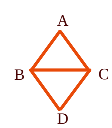
Multi-dimensional histograms on joint data distribution
**** expensive***
**** ineffective in high dimensions***

Fully independent attributes ⟷ **Fully correlated attributes**

- Extremes in terms of the underlying correlations!!
- Dependency-Based Histograms: explore space between extremes by explicitly identifying data correlations/independences
  - Build a *statistical interaction model* on data attributes
  - Based on the model, build a collection of low-dimensional histograms
  - Use this histogram collection to provide approximate answers
- General methodology, also applicable to other synopsis techniques (e.g., wavelets)

Dependency-based Histograms

- Identify (and exploit) attribute correlation and independence
  - Partial Independence :
    p(salary, height, weight) = p(salary) * p(height, weight)
  - Conditional Independence :
    p(salary, age | YPE) = p(salary| YPE) * p(age | YPE)
- Use forward selection to build a *decomposable statistical model* [BFH75], [Lau96] on the attributes
  - A,D are conditionally independent given B,C
    - p(AD|BC) = p(A|BC) * p(D|BC)
  - Joint distribution
    - p(ABCD) = p(ABC) * p(BCD) / p(BC)
  - ***Build histograms on model cliques***



- Significant accuracy improvements (factor of 5) over pure MHIST
- New histogram construction & usage algorithms, etc.

Workload-tuned Biased Sampling -- Congressional Samples [AGP00]

- Decision support queries routinely segment data into groups & then aggregate the information within each group
  - Each table has a set of "grouping columns": queries can group by any subset of these columns
- Goal: Maximize the accuracy for all groups (large or small) in each Group-by query
  - E.g., census DB with state (s), gender(g), and income (i)
  - Q: Avg(i) group-by s : seek good accuracy for all 50 states
  - Q: Avg(i) group-by s,g : seek good accuracy for all 100 groups
- Technique: Congressional Samples
  - House: Uniform sample: good for when no group-by
  - Senate: Same size sample per group when use all grouping columns: good for queries with all columns
  - Congress: Combines House & Senate, but considers all subsets of grouping columns, and then scales down

Workload-tuned Biased Sampling -- ICICLES [GLR00]

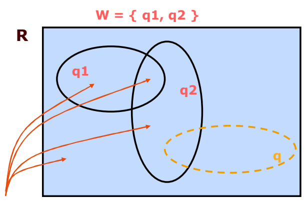
- Biased sampling scheme that *dynamically adapts* to query workload
  - Exploit data locality -- more focus (i.e., #sample points) in frequently-queried regions
  - Let Q = {q1, q2, . . .} be a query workload, R(qi) = subset of R used in answering query qi
  - L(R, Q) = Extension of R wrt Q = $R \bigcup_{qi \in Q} R(qi)$ (multiset of tuples)

Icicle: Uniform random sample of L(R,Q)

- Incrementally maintained and adapt ("self-tune") to workload through *Reservoir Sampling* technique [Vit85]
- *Unbiased Icicle estimators:* New formulas to account for duplicates and bias in sample selection
- *Provably better* (smaller variance) than uniform for "focused" queries (that follow the workload model)

Workload-tuned Biased Sampling -- Lifted Workloads [CDN01]

- Formulate sample selection as an *optimization problem*
  - Minimize query-answering error for a given workload model
- Technique for *"lifting a fixed workload W"* to produce a probability distribution over all possible queries
  - Similar to kernel density estimation (queries in W = "sample points")



prob(q|W) = parametric function of q's overlap with queries in W

"Fundamental regions" induced by W

Workload-tuned Biased Sampling -- Lifted Workloads

- Problem: Find sample of size k that minimizes expected error for a given "lifted" workload
- Solution: *Stratified sampling* [Coc77]
  - Collection of uniform samples (of total size k) over disjoint subsets ("strata") of the population
  - Much better estimates when variance within strata is small [Coc77]
- *Stratification:* Selecting appropriate partitioning of R
  - Using "fundamental regions" as strata is *optimal* for COUNT
  - For SUM, partition "fundamental regions" further to reduce variance of the aggregated attribute (Neymann technique [Coc77])
- *Allocation:* Dividing k among strata
  - Closed form solutions (valid under certain simplifying assumptions)

Data Streams

- Data is continually arriving. Collect & maintain synopses on the data. Goal: Highly-accurate approximate answers
  - State-of-the-art: Good techniques for narrow classes of queries
  - E.g., Any one-pass algorithm for collecting & maintaining a synopsis can be used effectively for data streams
- Alternative scenario: A collection of data sets. Compute a compact **sketch** of each data set & then answer queries (approximately) comparing the data sets
  - E.g., detecting near-duplicates in a collection of web pages: Altavista
  - E.g., estimating join sizes among a collection of tables [AGM99]

Looking Forward...

- Optimizing queries for approximation
  - e.g., minimize length of confidence interval at the plan root
- Exploiting mining-based techniques (e.g., decision trees) for data reduction and approximate query processing
  - see, e.g., [BGR01], [GTK01], [JMN99]
- Dynamic maintenance of complex (e.g., dependency-based [DGR01] or mining-based [BGR01]) synopses
- Synopses and approximate query processing for richer data models and data streams
  - e.g., XPath/XQuery over XML databases

XML Data (Text)

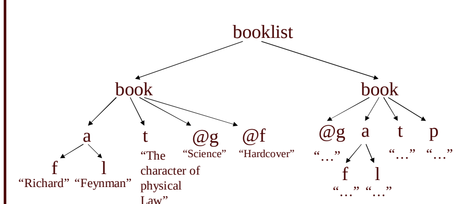
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<booklist>
<book genre="Science" format="Hardcover">
<author>
<firstname>Richard</firstname>
<lastname>Feynman</lastname>
</author>
<title>The character of Physical Law</title>
</book>
<book genre="Fiction">
<author>
<firstname>R.K.</firstname>
<lastname>Narayan</lastname>
</author>
<title>Waiting for the Mahatma</title>
<published>1981</published>
</book>
</booklist>

Content

Element

Nesting

XML Data (Tree)

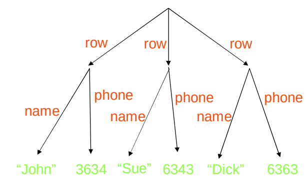


XML Basics

- Elements
  - Encode "concepts" in the XML database
  - Nesting denotes association/inclusion
- Attributes
  - Record information specific to an element (e.g., the genre of a book)
- References
  - Links between elements in different parts of the document

XML vs. Relational Data

name	phone
John	3634
Sue	6343
Dick	6363



Relation

XML

XML vs. Relational Data

- A relation instance is basically a tree with:
  - Unbounded fanout at level 1 (i.e., any # of rows)
  - Fixed fanout at level 2 (i.e., fixed # fields)
- XML data is essentially an arbitrary tree
  - Unbounded fanout at all nodes/levels
  - Any number of levels
  - Variable # of children at different nodes, variable path lengths

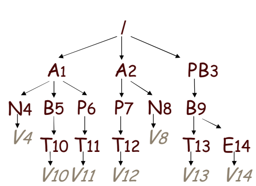
XPath Expressions

Examples:

- /booklist/book
- /booklist/book/author
- /booklist/book/author/lastname

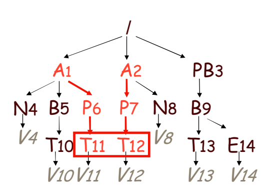
Given an XML document, the *value* of a path expression *p* is a set of elements (= XML subtrees)

Path Expressions



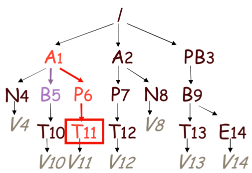
- XPath expressions
  - Simple: /A/P/T
  - Branching: /A[B]/P/T
  - Values: /A/P/T[=v11]
- Result is a set

Path Expressions



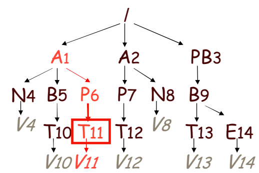
- XPath expressions
  - Simple: /A/P/T
  - Branching: /A[B]/P/T
  - Values: /A/P/T[=v11]
- Result is a set

Path Expressions



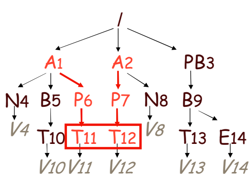
- XPath expressions
  - Simple: /A/P/T
  - Branching: /A[B]/P/T
  - Values: /A/P/T[=v11]
- Result is a set

Path Expressions



- XPath expressions
  - Simple: /A/P/T
  - Branching: /A[B]/P/T
  - Values: /A/P/T[=v11]
- Result is a set

Path Expressions



- XPath expressions
  - Simple: /A/P/T
  - Branching: /A[B]/P/T
  - Values: /A/P/T[=v11]
- Result is a set

XPath Syntax

- Path wildcards
  - // = descendant at any level (or self)
  - * = any (single) tag
  - Example: /booklist//lastname
- Query attributes and attribute content
  - Use "@"
  - Examples: /booklist//book[@format="Paperback"], /booklist//book/@genre
- Branching predicates: A[*pred*]
  - Predicate on A's subtree using *logical connectives* (and, or, etc.), *path expressions*, *built-in functions* (e.g., contains()), etc.
  - Example: //author[contains(./lastname, "Fey")]

Synopses for XML

- Summarize labeled tree/graph structure for approximate path navigation queries
  - Selectivity estimation: How many elements satisfy p?
  - Approximate answers: Return an *approximate* XML document as output of an XQuery fragment
- **Key idea:** Build a concise *Graph Synopsis* that captures the path/branching distribution in limited space
  - Use appropriate uniformity/independence assumptions to approximate path structure
  - Refine synopsis in parts of the XML document where assumptions fail
  - *XSketches* [SIGMOD'02,VLDB'02], *TreeSketches* [SIGMOD'04]

Conclusions

- Commercial data warehouses: approaching several 100's TB and continuously growing
  - Demand for high-speed, interactive analysis (click-stream processing, IP traffic analysis) also increasing
- Approximate Query Processing
  - "Tame" these TeraBytes and satisfy the need for interactive processing and exploration
  - Great promise
  - Commercial acceptance still lagging, but will most probably grow in coming years
  - *Still lots of interesting research to be done!!*