



Online Query Processing

A Tutorial

Peter J. Haas
IBM Almaden Research Center
Joseph M. Hellerstein
UC Berkeley

Goals for Today

- Exposure to online query processing algorithms and fundamentals
 - Usage examples
 - Basic sampling techniques and estimators
 - Preferential data delivery
 - Online join algorithms
 - Relation to OLAP, etc.
 - Some thoughts on research directions
- More resources to appear on the web
 - Annotated bibliography
 - Extended slide set
 - Survey paper

2

Road Map

- Background and motivation
 - Human-computer interaction
 - Tech trends and prognostications
 - Goals for online processing
- Examples of online techniques
- Underlying technology
- Related work
- Looking forward

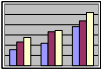
3

Human-Computer Interaction

- Iterative querying with progressive refinement
- Real-time interaction (impatience!)
 - Spreadsheets, WYSIWYG editors
 - Modern statistics packages
 - Netscape STOP button
- Visually-oriented interface

Year	Category 1	Category 2	Category 3
1,0000	1.01231	1.23456	1.56789
2,0000	1.14433	1.26194	1.41643
3,0000	1.44571	1.27148	10.2342

VS

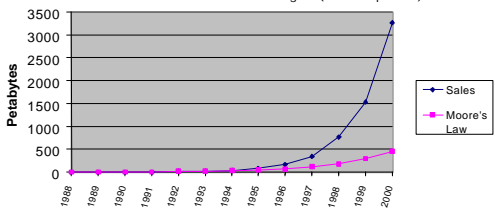


- Approximate results are usually OK

4

Disk Appetite

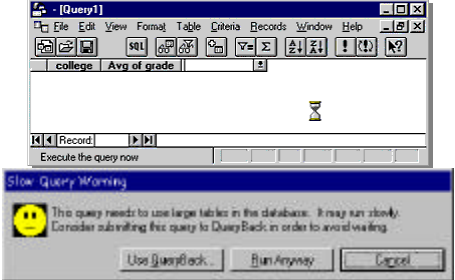
Greg Papadopoulos, CTO Sun:
"Moore's Law Ain't Good Enough" (Hot Chips '98)



Year

Source: J. Porter, Disk/Trend, Inc.
<http://www.disktrend.com/pdf/portrpk.pdf>

The Latest Commercial Technology



6

Drawbacks of Current Technology

- Only exact answers are available
 - A losing proposition as data volume grows
 - Hardware improvements not sufficient
- Interactive systems fail on massive data
 - E.g., spreadsheet programs (64Krow limit)
- DBMS not interactive
 - No user feedback or control ("back to the 60's")
 - Long processing times
 - Fundamental mismatch with preferred modes of HCI
- OLAP: a partial solution
 - Can't handle *ad hoc* queries or data sets

Goals for Online Processing

- New "greedy" performance regime
 - Maximize 1st derivative of the "mirth index"
 - Mirth defined on-the-fly
 - Therefore need FEEDBACK and CONTROL

Road Map

- Background and Motivation
- Examples of Online Techniques
 - Aggregation, visualization, cleaning/browsing
- Underlying technology
- Related work
- Looking Forward

Online Aggregation

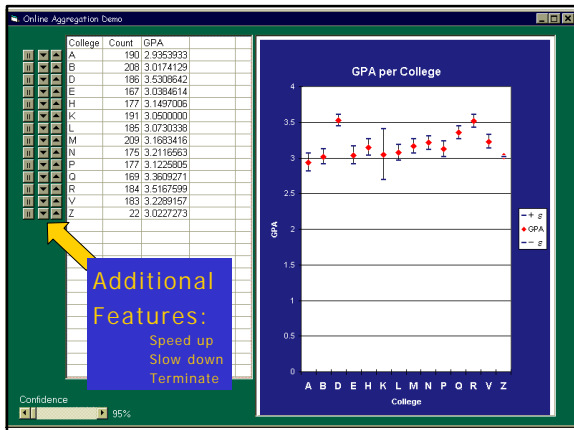
- `SELECT AVG(temp) FROM t GROUP BY site`
- 330K rows in table
- the exact answer:

Online Aggregation, cont'd

- A simple online aggregation interface (after 74 rows)

Online Aggregation, cont'd

- After 834 rows:



Online Data Visualization

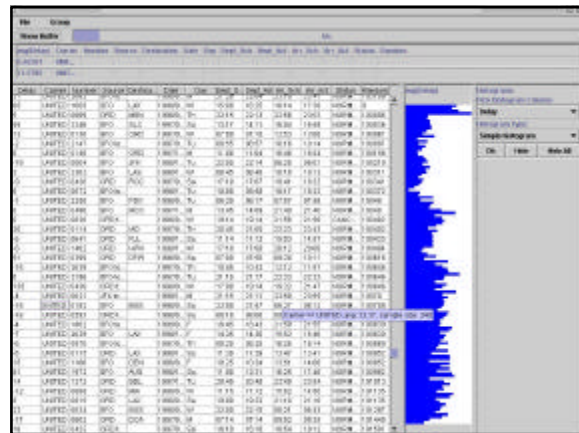
- In Tioga DataSplash

14

Online Enumeration

- Potter's Wheel [VLDB 2001]
- Scalable spreadsheet
 - A fraction of data is materialized in GUI widget
 - Scrolling = preference for data delivery in a quantile
 - Permits "fuzzy" querying
- Interactive data cleaning
- Online structure and discrepancy detection
- Online aggregation

15



Delay	Source	Dest	Carrier	Class	Date	Day	Direct discrepancies in data	Family()
-52	SFO to HNL							
-51	ORD	FDX						
-51	ORD	SFO						
-45	JFK	LAX						
-43	ORD	SFO						
-43	ORD	SEA						
-43	ORD to SEA							
-19	JFK	LAX						
-13	SFO	CVG						
-13	ORD	SNA						
-13	JFK	BJC						
-12	SFO	PIT						
-12	SFO	DNR						
-11	ORD	SEA						
-11	SFO to MSP							
-10	SFO	PHL						
-10	ORD	LAX						
-10	ORD to LAX							
-10	SFO	MIA						
-10	ORD	SFO						
-10	ORD to AAH							

Delay	Source	Destination	Date	Day	Direct discrepancies in data	Family()
1	SFO to JFK		1997/08/27	W	16:15	16:17
1	ORD	CLT	1997/09/29	W	07:00	07:02
1	SFO to BNA		1997/09/28	W	21:15	21:14
1	SFO to SEA		1997/09/27	F	22:30	22:36
1	ORD	MSP	1997/09/06	Tu	20:25	20:52
1	ORD to R		1997/09/04	Su	18:35	18:38
1	ORD	MSP	1997/09/10	Th	18:45	18:51
1	ORD	PEP	1997/09/22	Su	14:50	14:58
1	ORD	SEA	1997/09/22	Su	09:45	09:51
1	SFO	PHL	1997/09/03	F	11:00	11:03
1	SFO	MRY	1997/09/25	F	21:15	21:18
1	ORD	PHL	1997/09/30	Th	13:30	13:28
1	ORD to PHL		1997/09/24	Tu	11:00	10:58
1	ORD to IL		1997/09/08	Th	15:15	15:21
1	ORD	MSP	1997/09/05	W	13:00	11:57
1	ORD	OMA	1997/09/14	Sa	15:20	15:18
1	ORD to C...		1997/09/29	W	17:40	17:46
1	JFK	CLE	1997/09/12	W	16:59	16:58
1	ORD	MSP	1997/09/05	Th	22:15	22:13
1	ORD	TPA	1997/09/16	F	19:55	19:59
1	ORD to TPA		1997/09/20	Su	09:45	09:43

Road Map

- Background and motivation
- Examples of online techniques
- Underlying technology
 - Building blocks: sampling, estimation
 - Preferential data delivery
 - Pipelined adaptive processing algorithms
- Related work
- Looking forward

19

Sampling – Design Issues

- Granularity of sample
 - Instance-level (row-level): high I/O cost
 - Block-level (page-level): high variability from clustering
- Type of sample
 - Often simple random sample (SRS)
 - Especially for on-the-fly
 - With/without replacement usually not critical
- Data structure from which to sample
 - Files or relational tables
 - Indexes (B+ trees, etc)

20

Row-level Sampling Techniques

- Maintain file in random order
 - Sampling = scan
 - Is file initially in random order?
 - Statistical tests needed: e.g., Runs test, Smirnov test
 - In DB systems: cluster via RAND function
 - Must “freshen” ordering (online reorg)
- On-the-fly sampling
 - Via index on “random” column
 - Else get random page, then row within page
 - Ex: extent-map sampling
 - Problem: variable number of records on page

21

Acceptance/Rejection Sampling

- Accept row on page i with probability = n_i/n_{MAX}

Original pages		Modified pages	
rr	rrrrr	rrrrrr	rrrrrr
rrrrrr	rrrrrrr	rrrrrrr	rrrrrrr

- Commonly used in other settings
 - E.g. sampling from joins
 - E.g. sampling from indexes

22

Cost of Row-Level Sampling

Pages fetched (%)

Sampling Rate (%)

- 100,000 pages
- 200 rows/page

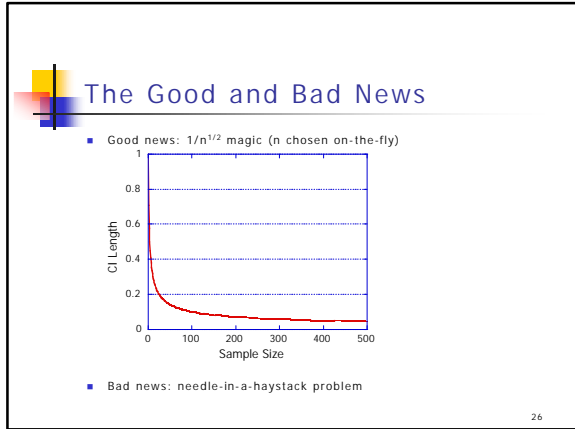
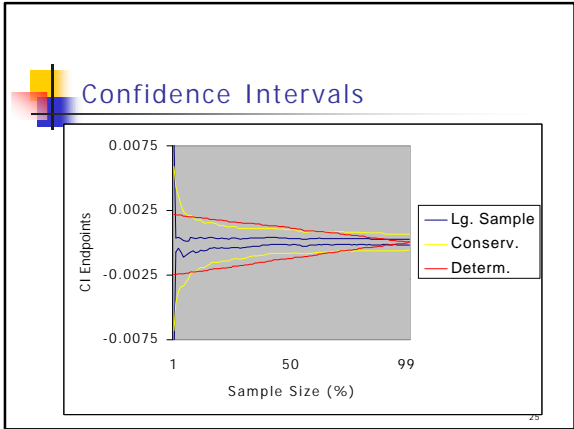
23

Estimation for Aggregates

- Point estimates
 - Easy: SUM, COUNT, AVERAGE
 - Hard: MAX, MIN, quantiles, distinct values
- Confidence intervals – a measure of precision

- Two cases: single-table and joins

24



- ## Sampling Deployed in Industry
- “Simulated” Bernoulli sampling
 - SQL: `SELECT * WHERE RAND() <= 0.01`
 - Similar capability in SAS
 - Bernoulli Sampling with pre-specified rate
 - Informix, Oracle 8i, (DB2)
 - Ex: `SELECT * FROM T1 SAMPLE ROW(10%), T2`
 - Ex: `SELECT * FROM T1 SAMPLE BLOCK(10%), T2`
 - Not for novices
 - Need to pre-specify precision
 - no feedback/control
 - recall the “multiresolution” patterns from example
 - No estimators provided in current systems
- 27

- ## Precomputation Techniques
- Two components
 - Data reduction (often expensive)
 - Approximate reconstruction (quick)
 - Pros and cons
 - Efficiency vs flexibility
 - Class of queries that can be handled
 - Degree of precision
 - Ease of implementation
 - How much of system must be modified
 - How sophisticated must developer be?
 - More widely deployed in industry
 - Will give overview later
- 28

- ## Road Map
- Background and motivation
 - Examples of online techniques
 - Underlying technology
 - Building blocks: sampling, estimation
 - Preferential data delivery
 - Pipelined adaptive processing algorithms
 - Related technology: precomputation
 - Looking forward
- 29

- ## Preferential Data Delivery
- Why needed
 - Speedup/slowdown arrows
 - Spreadsheet scrollbars
 - Pipeline quasi-sort
 - Continuous re-optimization (eddies)
 - Index stride
 - High I/O costs, good for outliers
 - Online Reordering (“Juggle”)
 - Excellent in most cases, no index required
 - [VLDB '99, VLDBJ '00]
- 30

Online Reordering

- Deliver "interesting" items first
 - "Interesting" determined on the fly
- Exploit rate gap between produce and process/consume

31

Online Reordering

- Deliver "interesting" items first
 - "Interesting" determined on the fly
- Exploit rate gap between produce and process/consume

32

Mechanism

- Two threads -- prefetch from input -- spool/enrich from auxiliary side disk
- Juggle data between buffer and side disk
 - keep buffer full of "interesting" items
 - getNext chooses best item currently on buffer
- getNext, enrich/spool decisions -- based on reordering policy
- Side disk management
 - hash index, populated in a way that postpones random I/O
 - play both sides of sort/hash duality

33

Policies

- "good" permutation of items $t_1 \dots t_n$ to $t_{p_1} \dots t_{p_n}$
- quality of feedback for a prefix $t_{p_1} t_{p_2} \dots t_{p_k}$

$$QOF(UP(t_{p_1}), UP(t_{p_2}), \dots, UP(t_{p_k})), \quad UP = \text{user preference}$$
 - determined by application
- goodness of reordering: $dQOF/dt$
- implication for juggle mechanism
 - process gets item from buffer that increases QOF the most
 - juggle tries to maintain buffer with such items

34

QOF in Online Aggregation

- avg weighted confidence interval
- preference acts as weight on confidence interval
 - $QOF = -\hat{\alpha} UP_i / (n_i)^{3/2}$, n_i = number of tuples processed from group i
- process pulls items from group with $\max UP_i / n_i^{3/2}$
 - desired ratio of group i in buffer = $UP_i^{2/3} / \hat{\alpha}_j UP_j^{2/3}$
 - juggle tries to maintain this by enrich/spool
- Similar derivations for other preferences
 - e.g. explicit rates, explicit ranking, etc.

35

Road Map

- Background and motivation
- Examples of online techniques
- Underlying technology
 - Building blocks: sampling, estimation, pre-computation
 - Preferential data delivery
 - Pipelined adaptive processing algorithms
- Related work
- Looking forward

36

Pipelined Data Processing

- Never, ever wait for anything to finish
- Selection: no problem
- Grouping: hash, don't sort
- Sorting: juggle if possible
- Joins?
 - SELECT AVG(R.a * S.b)
 - FROM R, S
 - WHERE R.c = S.c
- Sample of joins vs. join of samples

37

Traditional Nested Loops

R

```

1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7
2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5

```

S

38

Ripple Joins [SIGMOD '99]

- designed for online performance goals
 - Completely pipelined
 - Adapt to data characteristics
- designed for online performance goals
- simplest version
 - read new tuples *s* from *S* and *r* from *R*
 - join *r* and *s*
 - join *r* with old *S* tuples
 - join *s* with old *R* tuples

39

Basic Ripple Join

R

```

XXXX
XXXX
XXXX
XXXX

```

S

40

Block Ripple Joins (Size = 2)

R

```

XXXXXX
XXXXXX
XXXXXX
XXXXXX
XXXXXX
XXXXXX

```

S

41

Rectangular Ripple Join

R

```

XXXXXXXX
XXXXXXXX
XXXXXXXX
XXXXXXXX

```

S

42

Ripple Joins, cont'd

- Variants:
 - Block: minimizes I/O in alternating nested loops
 - Index: coincides with index-nested loop
 - Hash: symmetric hash tables
- Adaptive aspect ratio
 - User sets animation rate (via slider)
 - System goal:
 - minimize CI length
 - Subject to time constraint
 - System solves optimization problem (approximately)
 - Samples from higher-variance relation faster

43

Ripple Joins, cont'd

- Prototypes in Informix, IBM DB2
- Ongoing work on scalability issues
 - Memory compaction technique
 - Parallelism
 - Graceful degradation to out-of-core hashing
 - a la Tukwila, XJoin, but sensitive to statistical issues
 - Nested queries
 - Optimization issues
- A number of API and other systems issues
 - DMKD journal paper on Informix implementation
 - Forthcoming paper on sampling in DB2

44

Road Map

- Background and motivation
- Examples of online techniques
- Underlying technology
- Related work
 - Online query processing
 - Precomputation
- Looking forward

45

Related Work on Online QP

- Morgenstein's PhD, Berkeley '80
- Online Association Rules
 - Ng, et al's CAP, SIGMOD '98
 - Hidber's CARMA, SIGMOD '99
- Implications for deductive DB semantics
 - Monotone aggregation in LDL++, Zaniolo and Wang
- Online agg with subqueries
 - Tan, et al. VLDB '99
- Dynamic Pipeline Scheduling
 - Urhan/Franklin VLDB '01
- Pipelining Hash Joins
 - Raschid, Wilschut/Apers, Tukwila, XJoin
 - Relation to semi-naive evaluation
- Anytime Algorithms
 - Zilberstein, Russell, et al.

46

Precomputation: Explicit

- OLAP Data Cubes (drill-down hierarchies)
 - MOLAP, ROLAP, HOLAP
- Semantic hierarchies
 - APPROXIMATE (Vrbsky, et al.)
 - Query Relaxation, e.g. CoBase
 - Multiresolution Data Models (Silberschatz/Reed/Fussell)
- More general materialized views
 - See Gupta/Mumick's text

47

Precomputation: Stat. Summaries

- Histograms
 - Originally for aggregation queries, many flavors
 - Extended to enumeration queries recently
 - Multi-dimensional histograms
- Parametric estimation
 - Wavelets and Fractals
 - Discrete cosine transform
 - Regression
 - Curve fitting and splines
 - Singular-Value Decomposition (aka LSI, PCA)
- Indexes: hierarchical histograms
 - Ranking and pseudo-ranking
 - Aoki's use of GiSTs as estimators for ADTs
- Data Mining
 - Clustering, classification, other multidimensional models

48

Precomputed Samples

- Materialized sample views
 - Olken's original work
 - Chaudhuri et al.: join samples
 - Statistical inferences complicated over "recycled" samples?
- Barbara's quasi-cubes
- AQUA "join synopses" on universal relation
- Maintenance issues
 - AQUA's backing samples
- Can use fancier/more efficient sampling techniques
 - Stratified sampling or AQUA's "congressional" samples
- Haas and Swami AFV statistics
 - Combine precomputed "outliers" with on-the-fly samples

49

Stratified Sampling

simple stratified

50

Road Map

- Background and motivation
- Examples of online techniques
- Underlying technology
- Related Work
- Looking forward
 - Adaptive systems
 - Human-centered systems

51

Looking Forward: Adaptive Systems

- Observation/Decision \approx Modeling/Prediction
 - usually statistical
- Already critically important in today's systems
 - And imagine how important in ubiquitous computing!

52

A DBMS Tradition


- One instance: System R optimization
 - Observe: Runstats
 - Decide: Query Optimization
 - Act: Query Processing
- A powerful aspect of our technologies
 - Data independence & declarative languages
- Yet quite coarse-grained
 - Runstats once per day/week
 - Actions only per-query
 - Disk resource management: index and matview selection
 - Memory resource management: buffers and sort/hash space
 - Concurrency management: admission control

53

"Built-in" adaptivity

- Info systems should have adaptivity as a basic goal
 - Not just best-case performance
- Needs to pervade system
 - Core architectural work to be done here
 - E.g. pipelining required for multi-operator adaptivity
 - Observe more than one thing at a time
 - E.g. adaptive operators (a la ripple join)
 - E.g. adaptive optimization architectures (a la Eddies)
 - E.g. unify query processing with database design
 - Adaptivity should be built-in, not "bolted-on"
 - Wizards to turn existing knobs
 - Less helpful
 - Certainly less elegant
 - Might be technically more difficult!

54



Looking Forward: Human-Centered Systems

- Annual plea for UI work in DB Directions Workshops
 - UI's perceived as "soft", hard to measure/publish
- Yet people use our systems
 - And arguably we are trying to make them better for people
- Problem: our performance metrics
 - "Mirth index" vs. wall-clock time
 - One can find reasonable "hard" metrics for mirth
 - Many of these metrics may be statistical
 - Also consider "woe index", e.g. in maintainability
 - Most of these indices have to do with user time
 - Not, e.g., resource utilization
- Good UI work need not require good UIs!
 - Can attack new metrics directly
 - We don't have to go back to art school

55



Lessons Learned

- Dream about UIs, work on systems
 - User needs drive systems design!
- Systems and statistics intertwine
- All 3 go together naturally
 - User desires and behavior: 2 more things to model, predict
 - "Performance" metrics need to reflect key user needs

"What unlike things must meet and mate..."

-- Art, Herman Melville

56



More?

- Annotated bibliography & slides soon...

<http://control.cs.berkeley.edu/sigmod01/>

57