K-Relevance: A Spectrum of Relevance for Data Sources Impacting a Query

Huang and Naughton (SIGMOD '07)

Demian Lessa

CSE Department, SUNY at Buffalo

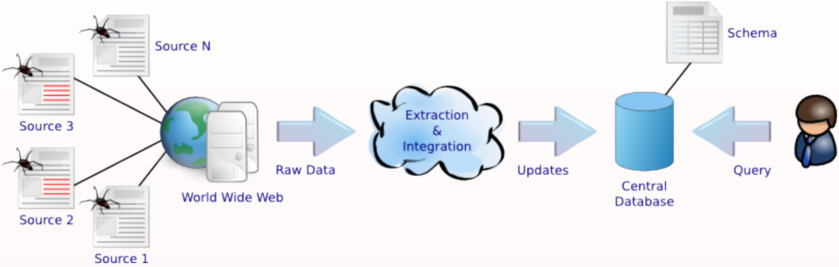
Spring 2008

Modern data management systems

- Remote sources and central database
- Sources periodically update database
- Database is usually out of date
- For instance:

  sensor data management
  distributed system monitoring
  web-based integration, etc

Web Integration Architecture Overview

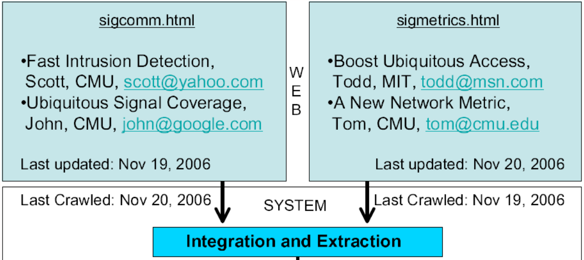


What are some of the issues?

- Data extraction might be fuzzy
  - updates to the database might be missing
  - users might not understand results completely
- Interpreting or debugging (results of) queries is challenging
  - especially in the presence of many sources

Investigating query answers

- Surprising tuples
  - which sources are responsible for the tuples?
  - why did they contribute such tuples?
- Suspiciously missing tuples
  - which sources could have contributed?
  - why did they not contribute such tuples?
  - could one or more (source) updates fix the problem?



Authors

source	authorName	authorOrg	authorEmail
sigcomm.html	John	CMU	John@google.com
sigcomm.html	Scott	CMU	Scott@yahoo.com

Papers

source	paperAuthor	paperTitle
sigcomm.html	John	Ubiquitous Signal Coverage
sigcomm.html	Scott	Fast Intrusion Detection

schema: {Authors, Papers}
sources: $\{S_1, S_2\}$

S_1 = sigcomm (synced)
S_2 = sigmetrics (not synced)

notice: source column
assume: joins on source

Q_1: papers by MIT authors with ‘Ubiquitous’ in title
$A_1 = \emptyset$

Q_2: names of CMU authors

A_2 = {John, Scott}

If only the system could restrict the sources to investigate...

If only the system could restrict the sources to investigate...

Here comes into play the notion of relevant sources.

What is a relevant source?

What is a relevant source?

Relevance may mean different things to

- different users
- the same user when posing different queries

What is a relevant source?

Relevance may mean different things to

- different users
- the same user when posing different queries

Relevance depends on

- the query Q posed by the user
  - a source might be relevant to Q_1 and not to Q_2
- the database instance I against which Q is executed
  - a source might be relevant to Q in I_1 and not in I_2

Definition (attempt #1)

Relevant Source. A source s is relevant to a query Q iff a **single update** to s **could change** the result of Q. ◇

Definition (attempt #1)

Relevant Source. A source s is relevant to a query Q iff a **single update** to s **could change** the result of Q. ◇

The definition above does not address important scenarios...

- Which sources participate in the derivation of a result tuple?
  - non-relevant sources are considered by attempt #1
- What results to expect when multiple updates to one source, single updates to multiple sources, or both are made?
  - relevant sources are not considered by attempt #1

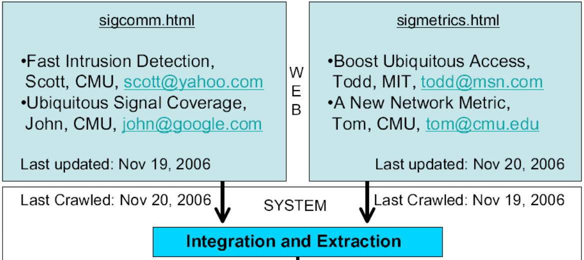
Definition (attempt #1)

Relevant Source. A source s is relevant to a query Q iff a **single update** to s **could change** the result of Q. ◇

The definition above does not address important scenarios...

- Which sources participate in the derivation of a result tuple?
  - non-relevant sources are considered by attempt #1
- What results to expect when multiple updates to one source, single updates to multiple sources, or both are made?
  - relevant sources are not considered by attempt #1

Let's illustrate this with an example!



Authors

source	authorName	authorOrg	authorEmail
sigcomm.html	John	CMU	John@google.com
sigcomm.html	Scott	CMU	Scott@yahoo.com

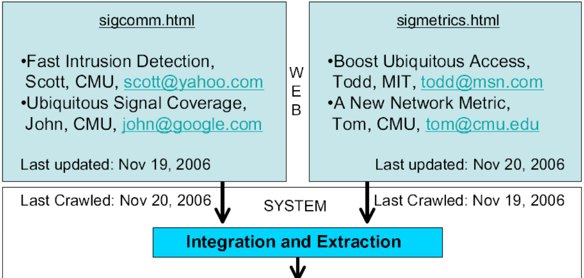
Papers

source	paperAuthor	paperTitle
sigcomm.html	John	Ubiquitous Signal Coverage
sigcomm.html	Scott	Fast Intrusion Detection

schema: {Authors, Papers}
sources: $\{S_1, S_2\}$

S_1 = sigcomm (synced)
S_2 = sigmetrics (not synced)

notice: source column



Authors

source	authorName	authorOrg	authorEmail
sigcomm.html	John	CMU	John@google.com
sigcomm.html	Scott	CMU	Scott@yahoo.com

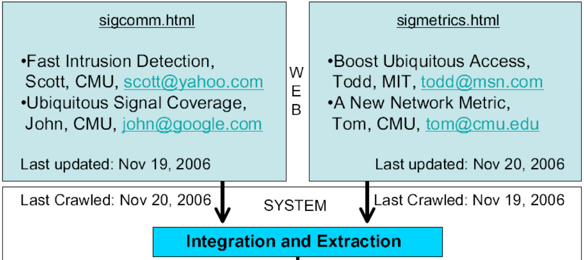
Papers

source	paperAuthor	paperTitle
sigcomm.html	John	Ubiquitous Signal Coverage
sigcomm.html	Scott	Fast Intrusion Detection

assume: join on source
Q: papers by MIT authors with ‘Ubiquitous’ in title
$A = \emptyset$

attempt #1:
$\rightarrow S_1$ is relevant (requires one update)
$\rightarrow S_2$ is not relevant (requires two updates)

misleading: A will change if S_2 is crawled!



Authors

source	authorName	authorOrg	authorEmail
sigcomm.html	John	CMU	John@google.com
sigcomm.html	Scott	CMU	Scott@yahoo.com

Papers

source	paperAuthor	paperTitle
sigcomm.html	John	Ubiquitous Signal Coverage
sigcomm.html	Scott	Fast Intrusion Detection

assume: join on source
Q: names of CMU authors
$A = \{$John, Scott$\}$

fact: Scott studies @MIT
fact: S_1 has faulty data!

attempt #1:
$\rightarrow$ S_1 and S_2 are relevant (each requires one update)

misleading: user wants to know that data in her answer came from S_1 so she can request a source update

What sources impact query Q, no matter how many updates?

What sources impact query Q, no matter how many updates?

Proposal: K-Relevant Sources

- Intuition: source s is k-relevant if potential updates to k relations, with at least one update from s, cause the results of Q to change

What sources impact query Q, no matter how many updates?

Proposal: K-Relevant Sources

- Intuition: source s is k-relevant if potential updates to k relations, with at least one update from s, cause the results of Q to change

Two kinds of relevance

- 0-Relevance or Lineage Relevance
  - a tuple from s participates in the derivation of a result tuple
- Update Relevance
  - updates to s could cause it to contribute to the derivation of a result tuple

What sources impact query Q, no matter how many updates?

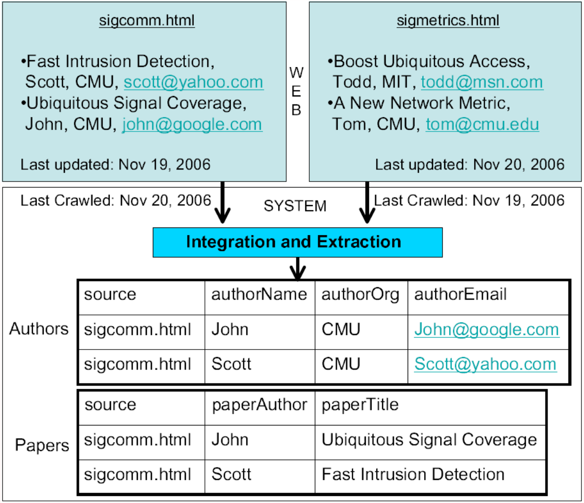
Proposal: K-Relevant Sources

- Intuition: source s is k-relevant if potential updates to k relations, with at least one update from s, cause the results of Q to change

Two kinds of relevance

- 0-Relevance or Lineage Relevance
  - a tuple from s participates in the derivation of a result tuple
- Update Relevance
  - updates to s could cause it to contribute to the derivation of a result tuple

The set of k-relevant sources for Q is a function of Q and the database instance I.



Authors

source	authorName	authorOrg	authorEmail
sigcomm.html	John	CMU	John@google.com
sigcomm.html	Scott	CMU	Scott@yahoo.com

Papers

source	paperAuthor	paperTitle
sigcomm.html	John	Ubiquitous Signal Coverage
sigcomm.html	Scott	Fast Intrusion Detection

schema: {Authors, Papers}
sources: $\{S_1, S_2\}$

S_1 = sigcomm (synced)
S_2 = sigmetrics (not synced)

Q: papers by MIT authors

SELECT
P.paperTitle
FROM
Authors A, Papers P
WHERE
A.authorOrg='MIT' **AND**
A.source = P.source **AND**
A.authorName = P.paperAuthor

WEB

sigcomm.html

- Fast Intrusion Detection, Scott, CMU, scott@yahoo.com
- Ubiquitous Signal Coverage, John, CMU, john@google.com

Last updated: Nov 19, 2006

sigmetrics.html

- Boost Ubiquitous Access, Todd, MIT, todd@msn.com
- A New Network Metric, Tom, CMU, tom@cmu.edu

Last updated: Nov 20, 2006

SYSTEM

Last Crawled: Nov 20, 2006

Last Crawled: Nov 19, 2006

Integration and Extraction

Authors

source	authorName	authorOrg	authorEmail
sigcomm.html	John	CMU	John@google.com
sigcomm.html	Scott	CMU	Scott@yahoo.com

Papers

source	paperAuthor	paperTitle
sigcomm.html	John	Ubiquitous Signal Coverage
sigcomm.html	Scott	Fast Intrusion Detection

Q: papers by MIT authors
$A = \emptyset$

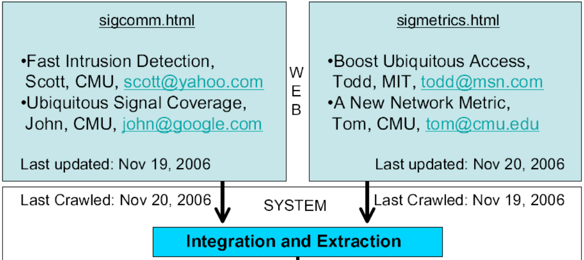
$0 - relevance$

Authors from S_1:
$\rightarrow$ no contribution

Papers from S_1:
$\rightarrow$ no contribution

Authors from S_2:
$\rightarrow$ no contribution

Papers from S_2:
$\rightarrow$ no contribution



Authors

source	authorName	authorOrg	authorEmail
sigcomm.html	John	CMU	John@google.com
sigcomm.html	Scott	CMU	Scott@yahoo.com

Papers

source	paperAuthor	paperTitle
sigcomm.html	John	Ubiquitous Signal Coverage
sigcomm.html	Scott	Fast Intrusion Detection

Q: papers by MIT authors
$A = \emptyset$

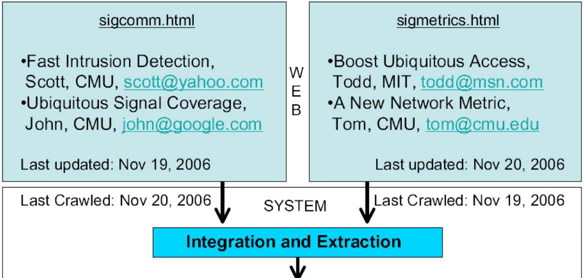
$1 - relevance$

Authors from S_1:
$\rightarrow$ change 'CMU' to 'MIT'

Papers from S_1:
$\rightarrow$ broken selection

Authors from S_2:
$\rightarrow$ no Papers to join

Papers from S_2:
$\rightarrow$ no Authors to join



Authors

source	authorName	authorOrg	authorEmail
sigcomm.html	John	CMU	John@google.com
sigcomm.html	Scott	CMU	Scott@yahoo.com

Papers

source	paperAuthor	paperTitle
sigcomm.html	John	Ubiquitous Signal Coverage
sigcomm.html	Scott	Fast Intrusion Detection

Q: papers by MIT authors
$A = \emptyset$

$2 - relevance$

for any source:
$\rightarrow$ add tuple τ_1 to Authors
$\rightarrow$ add tuple τ_2 to Papers
$\rightarrow$ τ_1 and τ_2 satisfy in Q

- Scenario
  - Central database schema, $\{R_1, \ldots, R_N\}$
  - Corresponding database instance, I
  - Multiple data sources contributing data to I
  - Conjunctive SPJ query Q
- Some relations can be updated by sources
  - such relations are denoted updated
  - $R_i.c_s^i$ identifies the source tuple $\tau \in R_i$ originates from
  - $\tau \in R_i$ can only be updated by the source in $R_i.c_s^i$
- Some relations cannot be updated by sources
  - such relations are denoted static
  - special static table H contains known sources (column $H.c_s$)

- Every Q references $R_1, \ldots, R_m, \ldots, R_n$, where
  - R_i can be updated, for $1 \leq i \leq m$
  - R_i is static, for $m < i \leq n$
  - aliases in Q are considered distinct relations
- Tuples in the current database instance are denoted real
- Tuples that could be inserted in the current database instance are denoted potential

Definition

K-Relevant Source. A source s is k-relevant for Q **via** R_i, denoted $s \in S^k_{R_i}$, if there exists at most k updated relations including R_i, such that there is:

- a potential tuple from s in R_i if $k > 0$,
- a potential tuple from any source for the other $k-1$ relations if $k > 1$, and
- a real tuple for each remaining relation such that they join to satisfy Q.

A source s is k-relevant for Q, denoted $s \in S^k_{all}$, if there exists a relation R_i such that s is k-relevant for Q **via** R_i. Notice that $S^k_{all} = \bigcup S^k_{R_i}$. ◇

Intuition

- $s \in S_R^0 \Rightarrow \exists$ real tuple in R from s
- $s \in S_R^1 \Rightarrow \exists$ potential tuple in R from s
- $s \in S_R^k \Rightarrow \exists$ potential tuples in R from s, and other $k-1$ relations from any source
- $s \in S_R^m \Rightarrow \exists$ potential tuples in R from s, and other $m-1$ relations from any source
- Notation: $S_R^m \equiv S_R^\infty$

An update may change both the result of a query and its set of relevant sources- e.g., an update from S_2 to Authors (example 2)!

An update may change both the result of a query and its set of relevant sources- e.g., an update from S_2 to Authors (example 2)!

Question

How can we use source relevance information to know which source updates may impact a query?

If $s \in S_{all}^{\infty}$

- Tuples from s might contribute to the result of Q
- Updates from s might change the result of Q
- Hence, s might be relevant to Q

If $s \notin S_{all}^{\infty}$

- No tuples from s can contribute to the result of Q
- No updates from s can change the result of Q
- Hence, s is irrelevant to Q

If $s \in S_R^\infty$

- Tuples in R from s might contribute to the result of Q
- Updates from s to R might change the result of Q
- Hence, s might be relevant to Q via R

If $s \notin S_R^\infty$

- No tuples in R from s can contribute to the result of Q
- No updates from s to R can change the result of Q
- Hence, s is irrelevant to Q via R

If $s \in S_R^k$

- Tuples in R from s might contribute to the result of Q
- Updates from s to R might change the result of Q
- Hence, s might be k-relevant to Q via R

If $s \notin S_R^k$

- No tuples in R from s can contribute to the result of Q
- No updates from s to R can change the result of Q...
  - unless more than k relation updates occur
  - in which case the result of Q might change if $s \in S_R^{k+1}$
- Hence, s might be k-relevant to Q via R

Lemma 3.2 (monotonicity)

$S_R^{k-1} \subseteq S_R^k$, for $k \geq 1$.

Intuition

- $s \in S_R^{k-1} \Rightarrow \exists$ **at most** $k-1$ updated relations including R from s
- By definition, $s \in S_R^k$
- If $s' \in S_R^k$ with **exactly** k updated relations including R from s', then $s' \notin S_R^{k-1}$

Corollary 3.3

$s \in S_R^k - S_R^{k-1}, k \geq 1 \Rightarrow$ updates from s to R do not change results of Q until **at least** $k-1$ other relations are updated.

Intuition

- $s \in S_R^k \Rightarrow$ one update from s to R, and **at most k-1** other updates change results of Q
- $s \notin S_R^{k-1} \Rightarrow$ one update from s to R, and **at most k-2** other updates do not change results of Q
- The claim follows

Corollary 3.3

$s \in S_R^k - S_R^{k-1}, k \geq 1 \Rightarrow$ updates from s to R do not change results of Q until **at least** $k-1$ other relations are updated.

Consequence

- If updates are made to one relation (say, R_i), only updates to sources in $S_{R_i}^1$ can change the results of Q
- If updates are made to two relations (say, R_i, R_j), only updates to sources in $S_{R_i}^2$ and $S_{R_j}^2$ can change the results of Q, and so on

Corollary 3.4

S_R^∞ never changes as a result of updates.

Intuition (no proof in the text)

- Current instances of updated relations do not play a role in computing S_R^∞. Only predicates (joins and selections), and constraints on source columns affect S_R^∞.

Corollary 3.5

S_R^k never changes as a result of updates to R alone, for $k > 0$.

Intuition (no proof in the text)

- For $k = 1$, any $s \in S_R^1$ can always contribute potential tuples for Q. For $k > 1$, since none of the other $k - 1$ updated relations are modified, they still can contribute $k - 1$ potential tuples to join with a potantial tuple from R and derive results for Q.

Theorem (3.6)

If an update is made to R_j from $s \notin S^{\infty}_{R_j} \Rightarrow S^{k}_{R_i}$ is unchanged for $i \neq j$ and all $k \geq 0$.

Proof Sketch

- assume $s \notin S^{k}_{R_i}(Q, I)$ and after the update to R_j from $s_1 \notin S^{\infty}_{R_j}(Q, I)$, $s \in S^{k}_{R_i}(Q, I')$
- w.r.t. I, there are potential tuples for R_i from s, for R_j from s_1, and for k-1 other relations, and real tuples for each of the remaining relations
- then, $s_1 \in S^{k+1}_{R_i} \Longrightarrow s_1 \in S^{\infty}_{R_i}$- contradiction!
- the case for $s_1 \in S^{k}_{R_i}$ also yields a contradiction (omitted)

Intuition for 0-Relevance:

- Modify Q to include source columns for all R_i
- Load results into temporary table T
- Query sources from T
- Filter source columns and output answers to Q
- Piggyback!

INPUT: I, $Q = \pi_F(\sigma_E(Rels))$
OUTPUT: query result A, $S^0_{R_i}$ for $1 \leq i \leq m$

0-RELEVANCE:

1: $T \leftarrow \pi_{F,R_1.c^1_s,\ldots,R_m.c^m_s}(\sigma_E(Rels))$
2: $S^0_{R_i} \leftarrow \pi_{R_i.c^i_s}(T)$ for $1 \leq i \leq m$
3: $A \leftarrow \pi_F(T)$
4: $S^0_{all} \leftarrow \bigcup_{1 \leq i \leq m} S^0_{R_i}$

Intuition for ∞-Relevance:

- Compute all implicit constrains on source column
- Append integrity and domain constrains as selections (when possible)
- Deal with general dependencies- static lookup tables
  - similar to source columns, hence omitted from presentation
- Notation

  P_s^i, J_s^i, P_o^i, $P_s^{i'}$, and $J_s^{i'}$
- Observation

  $S_{R_i}^{\infty}(Q, I) = \pi_{c_s}(\sigma_{P_s^{i'}}(H))$

INPUT: I, $Q = \pi_F(\sigma_{E \wedge P_s^1 \wedge \cdots \wedge P_s^m}(Rels))$
OUTPUT: $S^{\infty}_{R_i}$ for $1 \leq i \leq m$

∞-RELEVANCE:

for all $1 \leq i \leq m$ **do**
    Replace $R_i.c_s^i$ with $H.c_s$ in P_s^i to get $P_s^{i'}$
    $S^{\infty}_{R_i} \leftarrow \pi_{c_s}(\sigma_{P_s^{i'}}(H))$
end for
$S^{\infty}_{all} \leftarrow \bigcup_{1 \leq i \leq m} S^{\infty}_{R_i}$

Intuition for K-Relevance:

- Compute all implicit joins on source column, e.g.,
  - $R.c_s = S.c_1 \wedge S.c_1 = T.c_1 \Rightarrow R.c_s = T.c_1$
  - if $R.c_s = T.c_1$ is not inferred, S_R^k might be affected
- Assume
  - potential tuples for R_i, and $R_{j_1}, \ldots, R_{j_{k-1}}$
  - real tuples for $R_{j_k}, \ldots, R_{j_{m-1}}$
- For some combination of updated relations, if

  $$s \in \pi_{c_s}(\sigma_{P_s^{i\prime} \wedge J_s^{i\prime} \wedge P_o^i}(H \times \textstyle\prod_{k \leq l \leq (m-1)} R_{j_l}))$$

  then $s \in S_{R_i}^k$
- Combinatorially expensive (!) as $k \rightarrow m/2$
  - optimizations necessary!

INPUT: I, $Q = \pi_F(\sigma_{P_s^1 \wedge J_s^1 \wedge P_o^1 \wedge \cdots \wedge P_s^m \wedge J_s^m \wedge P_o^m}(Rels))$
OUTPUT: $S^k_{R_i}$ for $1 \leq i \leq m$

K-RELEVANCE:

for all $1 \leq i \leq m$ **do**
  Replace $R_i.c_s^i$ with $H.c_s$ in P_s^i to get $P_s^{i'}$
  for all choice of updated relations $R_{j_k}, \ldots, R_{j_{m-1}}$ (except R_i) **do**
    Replace $R_i.c_s^i$ with $H.c_s$ in J_s^i to get $J_s^{i'}$
    Evaluate $\pi_{c_s}(\sigma_{P_s^{i'} \wedge J_s^{i'} \wedge P_o^i}(H \times \prod_{k \leq l \leq (m-1)} R_{j_l}))$
    Union the result to $S^k_{R_i}$ and continue
  end for
end for
$S^k_{all} \leftarrow \bigcup_{1 \leq i \leq m} S^k_{R_i}$

Optimizations for K-Relevance:

- Alternate computation for large h and small k
  - monotonicity: $S_R^k = S_R^h \Rightarrow S_R^k = S_R^j$ for all $k \leq j \leq h$
- (Theorem 4.1) For some combination $R_{j_1}, \ldots, R_{j_{k-1}}$
  - if no R_{j_k} joins with R_i, and
  - $\sigma_{P_o^i}(\prod_{k \leq l \leq (m-1)} R_{j_l}) \neq \emptyset$, then

    $s \in \pi_{c_s}(\sigma_{P_s^{i\prime} \wedge P_o^i}(H \times \prod_{k \leq l \leq (m-1)} R_{j_l}))$

    $\Rightarrow s \in \pi_{c_s}(\sigma_{P_s^{i\prime}}(H \times \sigma_{P_o^i}(\prod_{k \leq l \leq (m-1)} R_{j_l})))$

    $\Rightarrow s \in \pi_{c_s}(\sigma_{P_s^{i\prime}}(H))$

    $\Rightarrow s \in S_{R_i}^{\infty}$

Intuition for K-Relevance Maintenance:

- Commonly asked query
  - materialize relevant sources for query
  - maintain incrementally
  - use of relevance information improves on existing algorithms
- Apply Corollary 3.5 and Theorem 3.6 to skip updates
- Rely on existing algorithms
  - incremental strategy
  - Counting Algorithm (Gupta et al) as "black box"

INPUT: updates U, materialized result for $S^k_{R_i}$ for $1 \leq i, k \leq m$
OUTPUT: new results for $S^k_{R_i}$ for $1 \leq i, k \leq m$

K-RELEVANCE MAINTENANCE:

for all $u \in U$ **do**
  ignore if u updates R_j but *source*$(u) \notin S^{\infty}_{R_j}$ // by thm 3.6
end for
Return if $U = \emptyset$
for all $1 \leq i \leq m$ **do**
  for all $u \in U$ **do**
    ignore if u updates R_i // by cor 3.5
  end for
  Continue if no updates left for $S^k_{R_i}$
  for all $1 \leq k \leq m$ **do**
    $\Delta \leftarrow$ COUNTING ALGORITHM(Q^k_i, U) // assume black box
    apply Δ to materialized result of $S^k_{R_i}$
  end for
end for

INPUT: updates U, materialized result for Q, Q', $S^k_{R_i}$ for $1 \leq i, k \leq m$
OUTPUT: new results for Q, Q', $S^k_{R_i}$ for $1 \leq i, k \leq m$

0-RELEVANCE MAINTENANCE:

$k \leftarrow \#$ relations all updates are made to
for all $u \in U$ **do**
    ignore if u updates R_j but *source*$(u) \notin S^\infty_{R_j}$ // by thm 3.6
    buffer if u updates R_j but *source*$(u) \notin S^k_{R_j}$
end for
Return if $U = \emptyset$
$\Delta \leftarrow$ COUNTING ALGORITHM(Q', U) // assume black box
apply Δ to materialized result of Q'
for all $\tau \in \Delta$ and $1 \leq i \leq m$ **do**
    $\tau =$ INS: add *count*(τ) to tuples in $S^0_{R_i}$ and S^0_{all} with same source
    $\tau =$ DEL: sub *count*(τ) from tuples in $S^0_{R_i}$ and S^0_{all} with same source
    project original fields from τ and apply to materialized result of Q
end for

Three Main Goals

- Compare costs of computing and maintaining k-relevant sources with cost of original query
- Understand the dependence of this overhead on the number of sources and size of data
- Gain insight on effectiveness of proposed algorithms in limiting relevant sources

Schema:

1: Authors(sourceId, confName, confYear, name, org, position, email)
2: Papers(sourceId, confName, confYear, authorName, authorOrg, title)
3: Students(sourceId, name, org, advsr, prog, yr)
4: ConfSourceFD(confName, confYear, sourceId)
5: StudentSourceFD(name, org, sourceId)
6: AllSources(id, url)

Schema:

1: Authors(sourceId, confName, confYear, name, org, position, email)
2: Papers(sourceId, confName, confYear, authorName, authorOrg, title)
3: Students(sourceId, name, org, advsr, prog, yr)
4: ConfSourceFD(confName, confYear, sourceId)
5: StudentSourceFD(name, org, sourceId)
6: AllSources(id, url)

Observations

- Underline indicates B-tree indices for fields
- Red indicates static tables
- ConfSourceFD models (confName, confYear) → sourceId
- StudentSourceFD models (name, org) → sourceId

Data:

- Conferences held for 30 years
- 100 author organizations, 10 students and 5 professors each
- I_1: 500 conferences/year and 200 papers/conference
- I_2: 5000 conferences/year and 200 papers/conference
- I_3: 500 conferences/year and 400 papers/conference
- Missing simulated data for conferences and students

Maintenance:

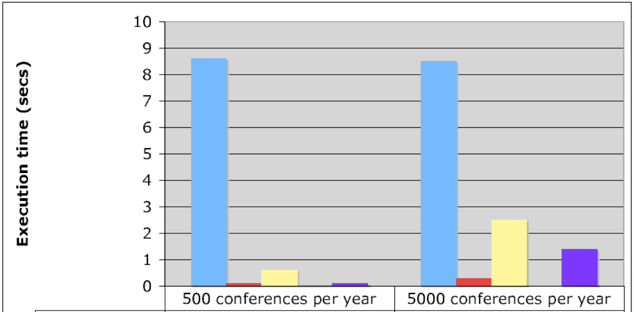
- Updates simulated with delta tables for Authors and Papers
  - one pair for 0-relevant sources
    $\Longrightarrow$ worst case maintanence of Q and its k-relevant sources
  - one pair for non ∞-relevant sources
    $\Longrightarrow$ ∞-relevant sources need no maintanence

Environment and Procedure Details:

- Intel Pentium 2.4GHz, 512MB RAM
- Tao Linux (RHEL 3.0 based)
- PostgreSQL 8.1.5 with default settings
  - Each query ran 11 times
  - Results are averaged over last 10 runs (warmed up cache)
  - Pre- and post-processing costs ignored (e.g., SQL parsing, result merging, etc)

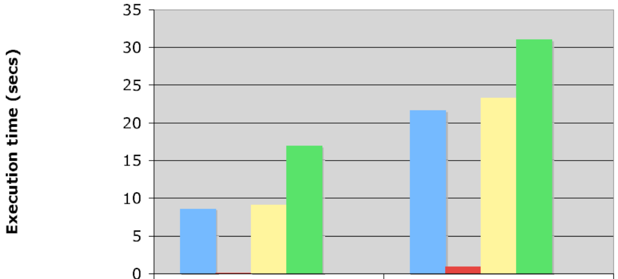
Query Q:

```
SELECT A.name, A.org, P.title, S.prog, S.yr, S.advsr
FROM Authors AS A, Papers AS P, Student AS S
WHERE A.confName IN [list of 10 conf types]
AND A.position = 'student'
AND A.name = P.authorName
AND A.org = P.authorOrg
AND A.confName = P.confName
AND A.confYear = P.confYear
AND A.name = S.name
AND A.org = S.org
```



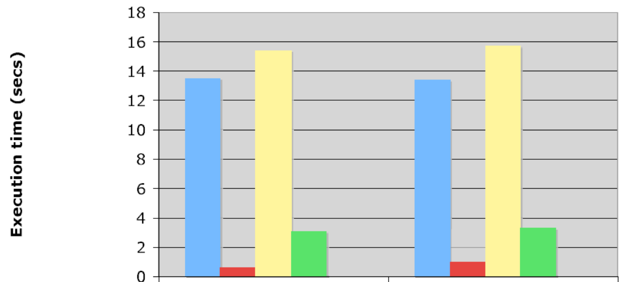
	500 conferences per year	5000 conferences per year
original query	8.6	8.5
0-relevant	0.1	0.3
1-relevant	0.6	2.5
2-relevant	0	0
3-relevant	0.1	1.4

[K-Relevance via Authors varying conferences/year] For 2-relevant sources, the optimization from Theorem 4.1 was used.



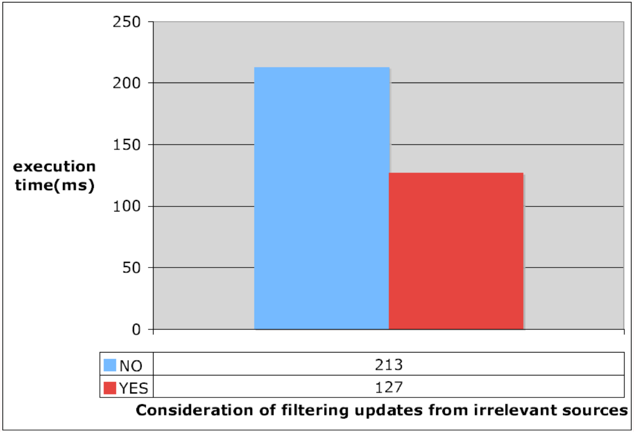
	200 papers per conf	400 papers per conf
original query	8.6	21.6
0-relevant	0.1	0.9
1-relevant	9.1	23.3
2-relevant	16.9	31
3-relevant	0.01	0.02

[K-Relevance via Students varying papers/conference] Notice that the numbers for 1- and 2- relevant grow at roughly the same rate as the numbers for Q

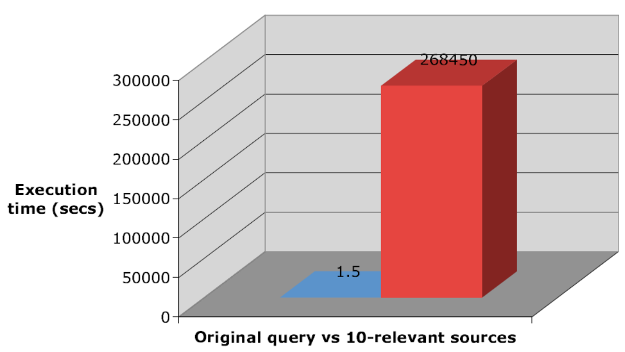


	500 conferences per year	5000 conferences per year
original query	13.5	13.4
0-relevant	0.6	1
1-relevant	15.4	15.7
2-relevant	3.1	3.3
3-relevant	0	0

[K-Relevance maintenance via Students for updates from 0-relevant sources] At worst, as bad as computing k-relevant sources for Q.



[Impact of filtering of updates from non ∞-relevant sources] Reduces the amount of updated data a maintenance query has to access.



original query	1.5
10-relevant	268450

[Computationally expensive scenario for k-relevant sources] 20-way self-join on Students table. High costs is due to the number of ways to choose 10 out of 19 updated relations when no optimizations are used.

Summary

- Definition of k-relevance
- Lemma, Corollaries, and Theorem establishing relationships between sets of relevant sources
- Efficient algorithms for computing some 0-relevant, and ∞-relevant sources
- A not so efficient algorithm for computing k-relevant sources, with optimizations to address many of the inefficient cases
- Algorithms for maintainence of materialized 0-relevant and k-relevant sources
- Experimental results

Future directions

- Data independent source relevance
- Relevance from the user's perspective
- More efficient algorithms for S_R^k

Jiansheng Huang and Jeffrey F. Naughton. K-Relevance: A Spectrum of Relevance for Data Sources Impacting a Query. ACM SIGMOD, Beijing, China, June 2007.