Provenance and History/Archiving

H.V. Jagadish

University of Michigan

Dynamic Systems

- System state changes over time
- Databases are subject to updates
- A user can customize her provenance request with regard to the updates

~~Four~~ Three Papers

- [M] Zhang + Jagadish (Mich)
  - "Dynamic Data Systems"
- [O] Gawlick + Radhakrishnan (Oracle)
  - "Fine Grain Using Temporal"
- [E] Halpin + Cheney (Edinb/www)
  - "Dynamic SPARQL Updates Named Graphs"
- [P] Zhou et al (Penn)
  - "Time Aware Distributed"

Edinburgh paper [E]

- RDF graphs change over time.
- Assume copy-paste paradigm for change.
- Use SPARQL to query for provenance that is change-aware.

Oracle Paper [O]

- Oracle provides the temporal data and version management capability required to support dynamic provenance.

PROVENANCE IN DYNAMIC SYSTEMS

Jing Zhang H.V. Jagadish

EECS Department
University of Michigan, Ann Arbor

Outline

- Introduction
- Lost Source Provenance
- Negative Provenance
- Non-Answer Provenance
- Conclusion

Lost source provenance

- Removing data from database causes (part of) the provenance missing from database.
- Retrieval of the lost provenance when the user asks for it.

Lost Source Provenance – Data Structures

- Information on previous database states
- Provenance log: recording the manipulations including queries and modifications
- Shadow table: archiving the removed tuples from the regular tables
- Extra attributes to the regular tables: referring to the manipulation that introduced the tuples into the database

Lost Source Provenance - Data Structure Example

ISBN	Title	Author	Since
0007208642	1940s Omnibus	A. Christie	0
0002310198	After the Funeral	A. Christie	0
0553380168	A Brief History of Time	S.W. Hawking	0
0742627098	Adventures of Gerard	A.C. Doyle	0

Book

ISBN	Price	Since
0007208642	9	0
0002310198	12	0
0553380168	11	2
0742627098	25	0

Price

ISBN	Price	Begin	End
0553380168	10	0	2

Price Shadow Table

Title	Price	Since
1940s Omnibus	9	1
A Brief History of Time	10	1

Result

ID	timestamp	user	sqlStatement
1	2009-08-01 01:00:00	Alice	Query
2	2009-08-02 11:00:00	Bob	Update

Provenance Log

Lost Source Provenance - Extended Tracing Queries

- When retrieving provenance, the tracing query uses a view that is created on the regular tables and the shadow tables and that represents the state of the regular tables at the time of the execution of the original query
- $$\{s : \langle B_1, ..., B_l \rangle \mid (S(s) \wedge s.since < id) \vee \exists s' (S_{sh}(s') \wedge s'.begin < id \wedge s'.end >= id \wedge s'.B_1 = s.B_1 \wedge ... \wedge s'.B_l = s_k.B_l)\}$$
- Where B_i are attribute names, S is the regular table, S_{sh} is the shadow table, *s* and *s'* are tuples, *id* is the entry ID in provenance log

Comparison With Temporal Database Based Approach

- Temporal database
  - Support the query over database at any previous time point
- Potential performance issues
  - Existing database applications on regular databases
  - The complicated timestamping mechanism potentially hinders the database performance
  - The time concept is not really needed to retrieve the lost provenance, since the relative ordering of queries and updates is the one that matters

Outline

- Introduction
- Lost Source Provenance
- Negative Provenance
- Non-Answer Provenance
- Conclusion

Negative provenance

- Adding new data to database turns answer to non-answer
- Validation of previous answer and retrieval of negative provenance in the new data as the explanation of invalidated previous answer

Negative Provenance - Validation Of Previously Derived Tuples

- Modifications to databases have potential effect on previously derived tuples
- View maintenance is one way to determine the effects
  - –Not sufficient for explanation of invalidated previous result tuples
  - –Not efficient for updating the complete result set when only a small fraction of result tuples are in question

Explanation for Invalidation of Previously Derived Tuples

- Regular provenance: its presence is required to derive a given result tuple
- Negative provenance: its absence is required to derive a given result tuple
  - –Intuitively, when certain tuples are added to the database, a given derived tuple (answer) will become invalid (non-answer)
  - –E.g., suppose a table `T(A,B)` and a query `SELECT max(A), B FROM T GROUP BY B`. If (1, 2) is in the result, the insertion of (2,2) will invalidate it.

Negative Provenance - Definition

- Suppose $t \in Q(\{T_1,....,T_n\})$. Then, $\{T_1^{'},...,T_n^{'}\}$ is a negative derivation of t according to Q if they satisfy the following requirements:

  1. $T_i^{'}$ is of the same schema as T_i
  2. $t \notin Q(\{T_1 \cup T_1^{'},....,T_n \cup T_n^{1}\})$
  3. $\neg \exists T_i^{''} \subset T_i^{'}$ such that $\{T_1 \cup T_1^{'},...,T_i \cup T_i^{''},...,T_n \cup T_n^{'}\}$

  satisfies the above two requirements. The union of all the negative derivations is called the negative provenance.

- Comparison
  - Regular provenance is within the database and consists of finite number of tuples
  - Negative provenance is outside the database and may consist of infinite number of tuples

Negative Provenance - Representation

- Due to the fact that negative provenance is both outside the database and can be of infinite tuples, the representation of it through listing all the tuples is not practical
- However, negative provenance can be detected within the delta tables that are inserted or removed from the original database

Outline

- Introduction
- Lost Source Provenance
- Negative Provenance
- Non-Answer Provenance
- Conclusion

Non-answer provenance

- Expected answer does not show up in the result
  - Changes to source database to produce the expected answer
  - Operations in the derivation that caused the lack of answers
- Construction of the changes and the detection of the causing operations

Non-Answer Provenance

- An expected answer does not show up in the result
- The reason lies either in the source data or in the operations in the derivation
- Proper changes to the source data can lead to the answer being produced as expected[1]
- Find out the responsible operations in the derivations that caused the lack of expected answer[2]

[1] On the provenance of non-answers to queries over extracted data. J. Huang, T. Chen, A. Doan, and J. F. Naughton. VLDB '08
[2] Why not? A. Chapman and H. V. Jagadish. SIGMOD '09

Conclusion

- Most database systems are subject to updates and thus provenance is subject to removal, modifications, etc.
- Users have their customized requests for provenance, which can usually be expressed some kind of filtering conditions on the complete provenance
- Related provenance problems include
  - Finding the lost source provenance
  - Detecting the negative provenance and explaining the answer turning non-answer
  - Computing customized provenance without involving the unrequested part
  - Explaining the non-answer

Fine Grain Provenance

- Data changes. Have to deal with change. Only a rare database that has no updates.
- Probably don’t have the same issue with workflow provenance (process or data set or large grain provenance).

- Session should not be called Provenance +History/Archiving
- History is fundamental to provenance.

Provenance QType

- Where
  - Data item derived from is updated.
- How
  - Log of actions that affect a specific datum.
- Why
- When
  - Time/event when X became true

Versioning Solves Most Things

- Treat each version (of both source and derived data) as an independent object.
- Maintain provenance of each version of derived, keeping track of version for each source datum referenced.
- Margo Seltzer said this too.
- Don’t really need temporal DB for this.
  - Unless it is Oracle

Causality Needs Notion of Time

- Basic fine grain provenance is enough for determining causality as defined by Suciu. (But not influence, and other new functions).
- Large number of variables can kill patient.
- Cause of death is the specific variable whose value changed – others are considered exogenous.
- Need history to do this!!