Logical Data Expiration

David Toman

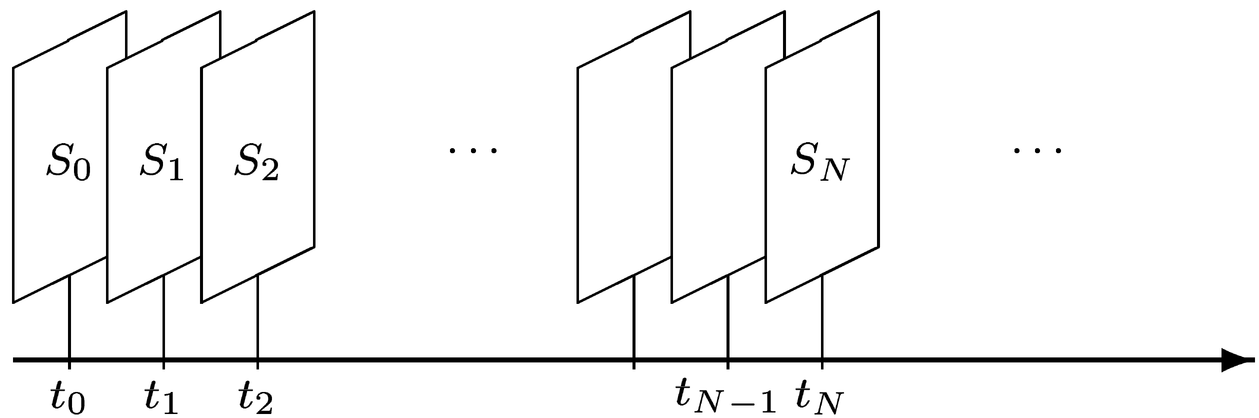
School of Computer Science

University of Waterloo

List of Slides

Data Evolution and Histories

Changes of data can be captured (conceptually) by *histories*:



- states S_i describe *system state*
- transitions $S_i \to S_{i+1}$ represent system evolution

  $\Rightarrow$ append only histories (new states appear at the end)

Data Access and Queries

Data is accessed using *queries*

- simple value look-ups vs. complex query languages
- current state only vs. access to *past states*

  ⇒ analysis of data warehouse evolution

  ⇒ enforcement of dynamic/temporal integrity constraints

  ⇒ monitoring applications

Expiration

1. *Policy*-driven expiration
2. *Query*-driven (logical) expiration

> The data to be removed (expired) is determined by
> **the (class of) queries** we are allowed to ask
> in **all possible extensions** of a history

Examples

- Record keeping/business rules:

  $\Rightarrow$ tax forms must be kept 5 years back

- Dynamic integrity constraints:

  $\Rightarrow$ don't hire anyone you've fired in the past

- Caching policies

  $\Rightarrow$ what data should be moved to backup storage?

- Moving window queries, etc. . .

Outline of the Talk

- Temporal Database Primer
- Expiration Operators

  $\Rightarrow$ How good is an expiration operator?
- Administrative Approaches to Expiration
- Query-driven Expiration

  $\Rightarrow$ Temporal Logic and Materialized Views

  $\Rightarrow$ First-order Queries and Partial Evaluation
- Space Limits for Expiration Operators
- Infinite Extensions of Histories and Potential Answers

Temporal Databases and Histories

System states: Relational structures (fixed schema)

Time: discrete (integer-like) $\{0, \ldots, N, \ldots\}$

1. Snapshot Temporal Database:

   $\Rightarrow$ time-indexed sequence of relational structures

   $\Rightarrow$ History, Kripke structure

2. Timestamp Temporal Database:

   $\Rightarrow$ time-indexed tuples (i.e., additional temporal attribute)

   $\Rightarrow$ **append**-only: $H; D_{N+1}$

Choices 1 and 2 equivalent [Chomicki and Toman, 1998]

Example

Information about TA and courses by semester:

0	$\{(\text{John}, \text{CS448})\}$
1	$\{(\text{John}, \text{CS448}), (\text{Sue}, \text{CS234})\}$
2	$\{(\text{John}, \text{CS448})\}$
3	$\{(\text{Sue}, \text{CS234})\}$

Temporal Queries

Queries: first-order formulas (over a fixed schema)

1. Temporal logic (FOTL)

   ⇒ modal (temporal) connectives

   ⇒ implicit references to time

2. Temporal Relational Calculus (2-FOL):

   ⇒ temporal variables/attributes/quantifiers

   ⇒ explicit access to time and ordering of time

Proposition 1 ([Abiteboul et al., 1996, Toman and Niwinski, 199

FOTL cannot express all 2-FOL queries.

Example

Students who TA'ed at least one class twice:

- in (past) FOTL:

$$\{x : \blacklozenge(\exists y.\mathrm{TA}(x,y) \wedge \bullet\blacklozenge\mathrm{TA}(x,y))\}$$

- in 2-FOL:

$$\{x : \exists t_1, t_2 . t_1 < t_2 \wedge \exists y.\mathrm{TA}(t_1,x,y) \wedge \mathrm{TA}(t_2,x,y)\}$$

Finite vs. Infinite Histories

Semantics of queries defined w.r.t:

1. current (finite) history

   $\Rightarrow$ query evaluation on a finite temporal database

2. a completion of current history

   $\Rightarrow$ hypothetical reasoning

Expiration Operator

- provides an *inductive definition*

$$
\begin{aligned}
\mathcal{E}(\langle\ \rangle) &= 0^{\mathcal{E}} && \text{(initial state)}\\
\mathcal{E}(H;D) &= \Delta^{\mathcal{E}}(\mathcal{E}(H),D) && \text{(extension maintenance)}
\end{aligned}
$$

  for an induced operator on histories, and

- maintains the following invariant:

$$Q(H) = Q^{\mathcal{E}}(\mathcal{E}(H)) \qquad \text{(answer preservation)}$$

Examples

- the *identity* operator:

$$0^{\mathcal{E}_{\rm id}} = \langle\ \rangle \qquad\qquad Q^{\mathcal{E}_{\rm id}} = Q$$
$$\Delta^{\mathcal{E}_{\rm id}} = \lambda H \lambda S.H; S$$

- the *current* operator:

$$0^{\mathcal{E}_{\rm now}} = \langle\ \rangle \qquad\qquad Q^{\mathcal{E}_{\rm now}} = Q$$
$$\Delta^{\mathcal{E}_{\rm now}} = \lambda H \lambda S.\langle S\rangle$$

Note that the supported query languages are *different*. . .

More Examples

- *compression* based operator:

$$0^{\mathcal{E}_{\text{compress}}} = \text{compress}(\langle\ \rangle)$$
$$\Delta^{\mathcal{E}_{\text{compress}}} = \lambda H \lambda S.\text{compress}(\text{decompress}(H); S)$$
$$Q^{\mathcal{E}_{\text{compress}}} = \lambda H.Q(\text{decompress}(H))$$

$\Rightarrow$ compress and decompress are *lossless*.

$\Rightarrow$ accounts for *interval encoding* of temporal databases.

Expiration vs. Queries Revisited

1. Given an *expiration operator*
   - for what class of queries it preserves answers?

     $\Rightarrow$ can these be characterized syntactically?

2. Given a *fixed set* of temporal queries:
   - is there an expiration operator that maintains answers to these queries?

     $\Rightarrow$ that minimizes $|\mathcal{E}(H)|$?

     $\Rightarrow$ can it be found algorithmically?
   - what query language can we formulate the queries?

How Good is an Expiration Operator?

What is the space needed by $\mathcal{E}(H)$ in terms of

1. size of the original history, $|H|$,
2. length of H (number of states, $|\mathbf{dom}_T|$),
3. the size of the *active data domain* of H (number of constants that have appeared in H, $|\mathbf{dom}_D|$),
4. size of the queries.

Goal: make the size of $\mathcal{E}(H)$ independent of length of H.

$\Rightarrow$ **bounded expiration operator**

Example

Proposition 2 $\mathcal{E}_{\mathrm{now}}$ *is bounded.*

Proposition 3 *Let* $\mathrm{compress}$ *and* $\mathrm{decompress}$ *define lossless compression scheme. Then* $\mathcal{E}_{\mathrm{compress}}$ *cannot be bounded.*

... how about $\mathcal{E}_Q$ for a temporal query Q?

Finite Histories

Query answers defined with respect to a finite history

$$\langle D_0, D_1, \ldots, D_n \rangle$$

$\Rightarrow$ active domain semantics.

Administrative Approaches

query-independent expiration policies.

- characterize queries whose answers are not affected, or
- detect attempts to access the *missing data* at run-time.

Most common approach: *history truncation* or *cutoff point*

1. policies based on fixed *absolute cutoff point*, or
2. policies based on *now-relative cutoff point*.

A generalization of the $\mathcal{E}^{\mathrm{id}}$ and the $\mathcal{E}^{\mathrm{now}}$ operators

Vacuuming

[Jensen, 1995]:

- $\rho(R) : e$ (a *remove* specification), and
- $\kappa(R) : e$ (a *keep* specification).

R is a temporal relation; e a selection condition

$\Rightarrow$ a special constant symbol *now*

Query Driven Expiration

Proposition 4 *Finite relational structures can be completely characterized by first-order queries.*

GOAL: an expiration operator for a fixed query Q.

- query language for Q?

  $\Rightarrow$ Past FOTL (and variants)

  $\Rightarrow$ Future FOTL

  $\Rightarrow$ 2-FOL

Proposition 5 *Optimal expiration operator is not possible.*

$\Rightarrow$ we try for a *bounded expiration operator*.

Query Driven Approaches

1. Removal of “old” states (expiration)

$\Rightarrow$ removes a **subset** of existing states

$\Rightarrow$ no other changes (maps a history to another history)

2. Auxiliary (non-temporal) view maintenance

$\Rightarrow$ maintains **auxiliary** relations

$\Rightarrow$ maps a history to *a single extended state*

3. Specialization of queries

$\Rightarrow$ specializes a given query w.r.t. the known prefix H.

Past Temporal Logic

- Syntax: First-order logic past temporal operators

$$Q ::= R(\mathbf{x}) \mid F \mid Q \wedge Q \mid \neg Q \mid \exists x.Q \mid \bullet Q \mid Q \ \mathbf{since}\ Q$$

$\Rightarrow$ queries over unbounded past: $\{x : \blacklozenge R(x)\}$

- Semantics:

$$Q(H) = \{\theta : H, \theta, n \models Q\}$$

where n is the *last* time instant in H.

Unfolding and Materialized Views

- Crux of the approach:

$$Q_1 \;\mathbf{since}\; Q_2 \quad\equiv\quad Q_1 \wedge (\bullet Q_2 \vee \bullet(Q_1 \;\mathbf{since}\; Q_2))$$

$\Rightarrow$ make an auxiliary view for each temporal subformula

$\Rightarrow$ use recurrent definitions of PastTL connectives to maintain the views.

α	R^0_α	R^n_α
$\bullet Q$	false	Q^{n-1}
$Q_1 \;\mathbf{since}\; Q_2$	false	$Q_1^n \wedge (Q_2^{n-1} \vee R_\alpha^{n-1})$

Example

- Query:

  Students who have TA'ed at least one class twice.

$$\{x : \blacklozenge(\exists y.\mathrm{TA}(x,y) \wedge \CIRCLE\blacklozenge\mathrm{TA}(x,y))\}$$

- Temporal subqueries:

  $\alpha_1 = \blacklozenge\mathrm{TA}(x,y)$ and

  $\alpha_2 = \CIRCLE\blacklozenge\mathrm{TA}(x,y)$ and

  $\alpha_3 = \blacklozenge\exists y.\mathrm{TA}(x,y) \wedge \blacklozenge\mathrm{TA}(x,y)$.

Example (cont.)

inductive maintenance of views:

$R_{\alpha_1}(x,y)$	$R_{\alpha_2}(x,y)$	$R_{\alpha_3}($
$\{(\text{John}, \text{CS448})\}$	$\{\}$	$\{\}$
$\{(\text{John}, \text{CS448}), (\text{Sue}, \text{CS234})\}$	$\{(\text{John}, \text{CS448})\}$	$\{$Johr
$\{(\text{John}, \text{CS448}), (\text{Sue}, \text{CS234})\}$	$\{(\text{John}, \text{CS448}), (\text{Sue}, \text{CS234})\}$	$\{$Johr
$\{(\text{John}, \text{CS448}), (\text{Sue}, \text{CS234})\}$	$\{(\text{John}, \text{CS448}), (\text{Sue}, \text{CS234})\}$	$\{$John, S

Space Utilization

- $Q = \blacklozenge(p(x_1) \wedge \ldots \wedge p(x_k))$
- $H = \langle \{a_1\}, \{a_2\}, \{a_3\}, \ldots, \{a_n\} \rangle$.

For $\alpha = \blacklozenge(p(x_1) \wedge \ldots \wedge p(x_k))$:

$$|R_\alpha| = (n-1)^k$$

... the same holds for every prefix of H.

Full details: [Chomicki, 1995], subsumes approaches based on TRA [Yang and Widom, 1998, Yang and Widom, 2000].

Adding Fixpoints

- Syntax: $Q ::= R(\mathbf{x}) \mid F \mid Q \wedge Q \mid \neg Q \mid \exists x.Q \mid \bullet Q \mid \mu X.Q$.
- Unfolding of fixpoint: $\mu X.Q \equiv Q(\mu X.Q)$
- Inductive maintenance of auxiliary relation:

α	R^0_α	R^n_α
$\bullet Q$	false	Q^{n-1}
$\mu X.Q$	Q^0	Q^n

Example

- Query: *students who TA'ed in "even" terms*

$$\mu X.\exists y.\mathrm{TA}(x,y) \vee \bullet\bullet X$$

- Temporal subformulas:

$\alpha_1 = \mu X.\exists y.\mathrm{TA}(x,y) \vee \bullet\bullet X$, $\alpha_2 = \bullet\bullet X$, and $\alpha_3 = \bullet X$.

	$R_{\alpha_1}(x)$	$R_{\alpha_2}(x)$	$R_{\alpha_3}(x)$
0	$\{\mathrm{John}\}$	$\{\}$	$\{\}$
1	$\{\mathrm{John}, \mathrm{Sue}\}$	$\{\}$	$\{\mathrm{John}\}$
2	$\{\mathrm{John}\}$	$\{\mathrm{John}\}$	$\{\mathrm{John}, \mathrm{Sue}\}$
3	$\{\mathrm{John}, \mathrm{Sue}\}$	$\{\mathrm{John}, \mathrm{Sue}\}$	$\{\mathrm{John}\}$

Metric Temporal Logic

- access to *real time* time instants

  $\Rightarrow$ a clk constant in each state (current *real* time)

  $\Rightarrow$ not part of the active data domain

- additional temporal operators

$$Q ::= \ldots \mid \mathbf{since}_{\sim c} \mid \bullet_{\sim c}$$

  $\Rightarrow$ semantics respects $\sim c$ distances

- materialized views now contain *distance* values

  $\Rightarrow$ bounded by c

  $\Rightarrow$ bounded expiration if $\mathrm{clk}^i - \mathrm{clk}^{i-1} \geq \epsilon > 0$

Future Temporal Logic

- Syntax: $Q ::= R(\mathbf{x}) \mid F \mid Q \wedge Q \mid \neg Q \mid \exists x.Q \mid \bigcirc Q \mid Q \;\mathbf{until}\; Q$
- Semantics:

$$Q(H) = \{\theta : H, \theta, 0 \models Q\}$$

where 0 is the *first* time instant in H.

⇒ but still active domain semantics

- Unfolding rule (similarly to PastTL):

$$Q_1 \;\mathbf{until}\; Q_2 \equiv Q_1 \wedge (\bigcirc Q_2 \vee \bigcirc(Q_1 \;\mathbf{until}\; Q_2))$$

⇒ but now we need to represent a formula with *holes* to be substituted when the history is extended.

Biquantified Formulas

- Automata-based approach

  $\Rightarrow$ designed in the *propositional* setting

  $\Rightarrow$ interleaving quantifiers and temporal connectives?

- [Lipeck and Saake, 1987, Lipeck et al., 1994]

  $\Rightarrow$ restrictions to Future FOTL syntax: 3 layers

  1. FO formulas (evaluated in a *state*,
  2. TL(FO) formulas (temporal operators on top of (1),
  3. Universal quantifiers on top of (2)

- bounded expiration based on an automaton for (2) implemented by *triggers*.

Two-sorted First-order Language

- Temporal Relational Calculus (2-FOL)

$$L ::= R(t, \mathbf{x}) | x = x' | t < t' | t = t' | L \wedge L | L \wedge \neg L | L \vee L | \exists x.L | \exists t.L$$

where $R(t, \mathbf{x})$ is true in H iff $R(\mathbf{x})$ is true in D_t

Does a **bounded expiration operator** exist for 2-FOL?

$\Rightarrow$ conjectured that it does NOT exist

NOTE: 2-FOL queries with *unbounded answers*

cannot have bounded expiration operator

$\Rightarrow$ consider only *bounded queries*

Expiration Revisited

Idea: remove those states that

1. do not contribute to query answer (due to $\wedge$)
2. contribute duplicate information (due to $\exists$)

Easy for a fixed history:

$\Rightarrow$ compute answer to Q bottom-up

$\Rightarrow$ propagate “back” to remove redundant data

Handling History Extensions

Atomic formulas:

- $\left[\begin{smallmatrix} x \\ a \end{smallmatrix}\right] \equiv \begin{cases} x = a & a \in \mathbf{dom}_D \\ \forall a \in \mathbf{dom}_D . x \neq a & a = \bullet \end{cases}$
- $\left[\begin{smallmatrix} t \\ s \end{smallmatrix}\right] \equiv \begin{cases} t = s & s \in \mathbf{dom}_T \\ t > \text{maxtime}(\mathbf{dom}_T) & s = \bullet \end{cases}$

Specialization of base relations *and their extensions*:

$$R(t, \mathbf{x}) \equiv \left(\bigvee_{\mathbf{a} \in R_{D_s}} \text{true}\left[\begin{smallmatrix} t\mathbf{x} \\ s\mathbf{a} \end{smallmatrix}\right] \right) \vee \left(\bigvee_{\mathbf{a} \in \mathbf{dom}_D \cup \{\bullet\}} R(t, \mathbf{x})\left[\begin{smallmatrix} t\mathbf{x} \\ \bullet\mathbf{a} \end{smallmatrix}\right] \right)$$

$\Rightarrow$ disjoint union

$\Rightarrow$ depends **only** on the future extensions

University of Waterloo

Query Specialization

$$
\mathsf{PE}_H(Q) = \begin{cases}
\{\mathsf{true}[\substack{t\mathbf{x}\\ s\mathbf{a}}] : R(s,\mathbf{a}) \in D\} & \\
\qquad \cup \{R(t,\mathbf{x})[\substack{t\mathbf{x}\\ \bullet\mathbf{a}}] : \mathbf{a} \in (\mathbf{dom}_D \cup \{\bullet\})^{|\mathbf{x}|}\} & Q \equiv R(t,\mathbf{x}) \\
\{Q_1'[\substack{\mathbf{x}\\ \mathbf{a}}] : Q_1'[\substack{\mathbf{x}\\ \mathbf{a}}] \in \mathsf{PE}_H(Q_1), \models [\substack{\mathbf{x}\\ \mathbf{a}}] \wedge F\} & Q \equiv Q_1 \wedge F \\
\{Q_1' \wedge Q_2'[\substack{\mathbf{xy}\\ \mathbf{ab}}] : Q_1'[\substack{\mathbf{x}\\ \mathbf{a}}] \in \mathsf{PE}_H(Q_1), Q_2'[\substack{\mathbf{y}\\ \mathbf{b}}] \in \mathsf{PE}_H(Q_2), \models [\substack{\mathbf{xy}\\ \mathbf{ab}}]\} & Q \equiv Q_1 \wedge Q_2 \\
\{(\exists y. \bigvee_{Q_1'[\substack{\mathbf{x}y\\ \mathbf{a}b}] \in \mathsf{PE}_H(Q_1)} Q_1')[\substack{\mathbf{x}\\ \mathbf{a}}] : \exists b. Q_1''[\substack{\mathbf{x}y\\ \mathbf{a}b}] \in \mathsf{PE}_H(Q_1)\} & Q \equiv \exists y. Q_1 \\
\{(\exists t. \bigvee_{Q_1'[\substack{\mathbf{x}t\\ \mathbf{a}s}] \in \mathsf{PE}_H(Q_1)} Q_1')[\substack{\mathbf{x}\\ \mathbf{a}}] : \exists s. Q_1''[\substack{\mathbf{x}t\\ \mathbf{a}s}] \in \mathsf{PE}_H(Q_1)\} & Q \equiv \exists t. Q_1 \\
\{Q_1' \wedge \neg Q_2'[\substack{\mathbf{x}\\ \mathbf{a}}] : Q_1'[\substack{\mathbf{x}\\ \mathbf{a}}] \in \mathsf{PE}_H(Q_1), Q_2'[\substack{\mathbf{x}\\ \mathbf{a}}] \in \mathsf{PE}_H(Q_2)\} & \\
\qquad \cup \{Q_1'[\substack{\mathbf{x}\\ \mathbf{a}}] : Q_1'[\substack{\mathbf{x}\\ \mathbf{a}}] \in \mathsf{PE}_H(Q_1), Q_2'[\substack{\mathbf{x}\\ \mathbf{a}}] \notin \mathsf{PE}_H(Q_2)\} & Q \equiv Q_1 \wedge \neg Q \\
\{Q_1' \vee Q_2'[\substack{\mathbf{x}\\ \mathbf{a}}] : Q_1' \in \mathsf{PE}_H(Q_1)[\substack{\mathbf{x}\\ \mathbf{a}}], Q_2'[\substack{\mathbf{x}\\ \mathbf{a}}] \in \mathsf{PE}_H(Q_2)\} & \\
\qquad \cup \{Q_1'[\substack{\mathbf{x}\\ \mathbf{a}}] : Q_1'[\substack{\mathbf{x}\\ \mathbf{a}}] \in \mathsf{PE}_H(Q_1), Q_2'[\substack{\mathbf{x}\\ \mathbf{a}}] \notin \mathsf{PE}_H(Q_2)\} & \\
\qquad \cup \{Q_2'[\substack{\mathbf{x}\\ \mathbf{a}}] : Q_1'[\substack{\mathbf{x}\\ \mathbf{a}}] \notin \mathsf{PE}_H(Q_1), Q_2'[\substack{\mathbf{x}\\ \mathbf{a}}] \in \mathsf{PE}_H(Q_2)\} & Q \equiv Q_1 \vee Q_2
\end{cases}
$$

University of Waterloo

Example

The PE$_H$ operator applied on the subquery

$$\exists t_1, t_2. \boxed{t_1 < t_2 \land \exists y.\mathrm{TA}(t_1, x, y) \land \mathrm{TA}(t_2, x, y)}$$

yields the following set of formulas:

$$\mathrm{true}\ \left[\begin{smallmatrix} t_1 & t_2 & x \\ 1 & 2 & \mathrm{John} \end{smallmatrix}\right]$$

$$\mathrm{true}\ \left[\begin{smallmatrix} t_1 & t_2 & x \\ 1 & 3 & \mathrm{John} \end{smallmatrix}\right]$$

$$\mathrm{true}\ \left[\begin{smallmatrix} t_1 & t_2 & x \\ 2 & 3 & \mathrm{John} \end{smallmatrix}\right]$$

$$\mathrm{TA}(x, \mathrm{CS448}, t_2)\ \left[\begin{smallmatrix} t_1 & t_2 & x \\ 1 & \bullet & \mathrm{John} \end{smallmatrix}\right]$$

$$\mathrm{TA}(x, \mathrm{CS448}, t_2)\ \left[\begin{smallmatrix} t_1 & t_2 & x \\ 2 & \bullet & \mathrm{John} \end{smallmatrix}\right]$$

$$\mathrm{TA}(x, \mathrm{CS448}, t_2)\ \left[\begin{smallmatrix} t_1 & t_2 & x \\ 3 & \bullet & \mathrm{John} \end{smallmatrix}\right]$$

$$\mathrm{true}\ \left[\begin{smallmatrix} t_1 & t_2 & x \\ 2 & 4 & \mathrm{Sue} \end{smallmatrix}\right]$$

$$\mathrm{TA}(x, \mathrm{CS234}, t_2)\ \left[\begin{smallmatrix} t_1 & t_2 & x \\ 2 & \bullet & \mathrm{Sue} \end{smallmatrix}\right]$$

$$\mathrm{TA}(x, \mathrm{CS234}, t_2)\ \left[\begin{smallmatrix} t_1 & t_2 & x \\ 4 & \bullet & \mathrm{Sue} \end{smallmatrix}\right]$$

Duplicate Information Removal

Modify the PE_H for quantification over time:

$$(\exists t. \bigvee_{\substack{Q_1'[{\mathbf{x}t \atop \mathbf{a}s}] \in \mathsf{PE}_H(Q_1) \\ s \in \mathsf{TB}_{\mathbf{a}}(t)}} Q_1')[{\mathbf{x} \atop \mathbf{a}}] \text{ where } Q_1''[{\mathbf{x}t \atop \mathbf{a}s}] \in \mathsf{PE}_H(Q_1) \text{ for some } s$$

$\longleftarrow$ what is this??

Definition 1: Let $Q_1[{\mathbf{x}t \atop \mathbf{a}s_1}], Q_2[{\mathbf{x}t \atop \mathbf{a}s_2}] \in \mathsf{PE}_H(Q)$ for $s_1 \neq s_2$.

We define $[{\mathbf{x}t \atop \mathbf{a}s_1}] \sim_Q^D [{\mathbf{x}t \atop \mathbf{a}s_2}]$ iff for any extension D' of D

$$(\mathbf{a}, s_1) \in Q(D; D') \iff (\mathbf{a}, s_2) \in Q(D; D')$$

Definition 2: $\mathsf{TB}_{\mathbf{a}}(t)$ is the set of representatives of the $[{\mathbf{x}t \atop \mathbf{a}s_1}] \sim_Q^D [{\mathbf{x}t \atop \mathbf{a}s_2}]$ equivalence classes [min in time order].

University of Waterloo

Equivalence in Extensions

$Q = R$: $[^{\mathbf{x}}_{\mathbf{a}_1}] \sim^D_Q [^{\mathbf{x}}_{\mathbf{a}_2}] \iff ((Q_1 = Q_2 = \text{ true}) \vee (\mathbf{a}_1 = \mathbf{a}_2))$

$Q = Q' \wedge F$: $[^{\mathbf{x}}_{\mathbf{a}_1}] \sim^D_Q [^{\mathbf{x}}_{\mathbf{a}_2}] \iff$
$\left([^{\mathbf{x}}_{\mathbf{a}_1}] \sim^D_{Q'} [^{\mathbf{x}}_{\mathbf{a}_2}] \text{ where } [^{\mathbf{x}}_{\mathbf{a}_1}] \wedge F \text{ and } [^{\mathbf{x}}_{\mathbf{a}_2}] \wedge F \text{ are satisfiable}\right)$,

$Q = \exists y.Q'$: Let $S_1 = \{b : Q'_1[^{y\mathbf{x}}_{b\mathbf{a}_1}] \in \mathsf{PE}_H(Q')\}$,
$S_2 = \{b : Q'_2[^{y\mathbf{x}}_{b\mathbf{a}_2}] \in \mathsf{PE}_H(Q')\}$. $[^{\mathbf{x}}_{\mathbf{a}_1}] \sim^D_Q [^{\mathbf{x}}_{\mathbf{a}_2}] \iff$
$\left((\forall b \in S_1 \exists c \in S_2.[^{y\mathbf{x}}_{b\mathbf{a}_1}] \sim^D_{Q'} [^{y\mathbf{x}}_{c\mathbf{a}_2}]) \wedge (\forall c \in S_2 \exists b \in S_1.[^{y\mathbf{x}}_{b\mathbf{a}_1}] \sim^D_{Q'} [^{y\mathbf{x}}_{c\mathbf{a}_2}])\right)$,

$Q = \exists t.Q'$: Let $S_1 = \{s : Q'_1[^{t\mathbf{x}}_{s\mathbf{a}_1}] \in \mathsf{PE}_H(Q')\}$,
$S_2 = \{s : Q'_2[^{t\mathbf{x}}_{s\mathbf{a}_2}] \in \mathsf{PE}_H(Q')\}$. $[^{\mathbf{x}}_{\mathbf{a}_1}] \sim^D_Q [^{\mathbf{x}}_{\mathbf{a}_2}] \iff$
$\left((\forall r \in S_1 \exists s \in S_2.[^{t\mathbf{x}}_{r\mathbf{a}_1}] \sim^D_{Q'} [^{t\mathbf{x}}_{s\mathbf{a}_2}]) \wedge (\forall r \in S_2 \exists s \in S_1.[^{t\mathbf{x}}_{r\mathbf{a}_1}] \sim^D_{Q'} [^{t\mathbf{x}}_{s\mathbf{a}_2}])\right)$,

$Q = Q' \wedge Q''$: $[^{\mathbf{x}}_{\mathbf{a}_1}] \sim^D_Q [^{\mathbf{x}}_{\mathbf{a}_2}] \iff$
$\left([^{\mathbf{x}'}_{\mathbf{a}'_1}] \sim^D_{Q'} [^{\mathbf{x}'}_{\mathbf{a}'_2}] \wedge [^{\mathbf{x}''}_{\mathbf{a}''_1}] \sim^D_{Q''} [^{\mathbf{x}''}_{\mathbf{a}''_2}], \text{ for } [^{\mathbf{x}}_{\mathbf{a}_i}] = [^{\mathbf{x}'\mathbf{x}''}_{\mathbf{a}'_i\mathbf{a}''_i}] \text{ satisfiable}\right)$,

$Q = Q' \wedge \neg Q''$ or $Q = Q' \vee Q''$:
$[^{\mathbf{x}}_{\mathbf{a}_1}] \sim^D_Q [^{\mathbf{x}}_{\mathbf{a}_2}] \iff \left([^{\mathbf{x}}_{\mathbf{a}_1}] \sim^D_{Q'} [^{\mathbf{x}}_{\mathbf{a}_2}] \wedge [^{\mathbf{x}}_{\mathbf{a}_1}] \sim^D_{Q''} [^{\mathbf{x}}_{\mathbf{a}_2}]\right)$.

Example

$$\left\{ \begin{bmatrix} t_1 & t_2 & x \\ 1 & 2 & \text{John} \end{bmatrix}, \begin{bmatrix} t_1 & t_2 & x \\ 1 & 3 & \text{John} \end{bmatrix}, \begin{bmatrix} t_1 & t_2 & x \\ 2 & 3 & \text{John} \end{bmatrix} \right\},$$

$$\left\{ \begin{bmatrix} t_1 & t_2 & x \\ 1 & \bullet & \text{John} \end{bmatrix}, \begin{bmatrix} t_1 & t_2 & x \\ 2 & \bullet & \text{John} \end{bmatrix}, \begin{bmatrix} t_1 & t_2 & x \\ 3 & \bullet & \text{John} \end{bmatrix} \right\},$$

$$\left\{ \begin{bmatrix} t_1 & t_2 & x \\ 2 & 4 & \text{Sue} \end{bmatrix} \right\},$$

$$\left\{ \begin{bmatrix} t_1 & t_2 & x \\ 2 & \bullet & \text{Sue} \end{bmatrix}, \begin{bmatrix} t_1 & t_2 & x \\ 4 & \bullet & \text{Sue} \end{bmatrix} \right\}$$

Residual History Reconstruction

- *Specialization-based* expiration:

$$\begin{aligned} Q(H) &= \text{PE}_H(Q)(\emptyset) \\ \text{PE}_{H;H'}(Q) &\equiv \text{PE}_{H'}(\text{PE}_H(Q)) \end{aligned}$$

- We use $\text{PE}_H(Q)$ to *construct* $\mathcal{E}(H)$

$\Rightarrow$ for each temporal variable t_i we define a unary relation

$$T_i(t) = \bigcup_{\mathbf{a}} \text{TB}_{\mathbf{a}}(t).$$

$\Rightarrow$ each quantifier $\exists t_i.Q'$ in Q is restricted to $T_i(t)$

$\Rightarrow$ a state $D_j \in H$ is expired if $j \notin \bigcup_i T_i$.

University of Waterloo

Example

$$\mathsf{TB}_{2,\mathrm{John}}(t_1) = \{1\} \qquad \mathsf{TB}_{4,\mathrm{Sue}}(t_1) = \{2\}$$

$$\mathsf{TB}_{3,\mathrm{John}}(t_1) = \{1\} \qquad \mathsf{TB}_{\bullet,\mathrm{Sue}}(t_1) = \{2\}$$

$$\mathsf{TB}_{\bullet,\mathrm{John}}(t_1) = \{1\}$$

$\Rightarrow$ is sufficient to keep only states $1 \text{ and } 2$ as valuations for the variable t_1.

Properties of the Expiration Operator

- $Q(H; H') = Q(\mathcal{E}_Q(H); H')$

  for all H, H' histories and Q FO query

- $|\,\mathcal{E}_Q(H)| \leq f(|\mathbf{dom}_D|, |Q|)$,

  f is an exponential tower in number of nested $\exists$.

- $|\,\mathcal{E}_Q(H)| \leq |H| + |\mathbf{dom}_T||Q|$

... and can be implemented by FO queries/updates.

Space: Lower Bound

Example: $\exists t_1, t_2 . t_1 < t_2 \wedge \forall x . R(t_1, x) \iff R(t_2, x)$

- Potentially we need to keep all states for which R contains distinct subsets of $\mathbf{dom}_D$

  $\Rightarrow$ potentially all subsets of $\mathbf{dom}_D$
  $\Rightarrow$ therefore any expired history is exponential in $|\mathbf{dom}_D|$.

- extended to *sequences of states* yields more exponents.

Limits of Bounded Encoding

Clearly, this cannot work for all possible queries:

Example 1: Query $\{t : R(t)\}$.

$\Rightarrow$ answer $\sim |\mathbf{dom}_T H|$

Example 2: Query $\{t : R(t) \wedge \forall t'.R(t') \rightarrow t \geq t'\}$.

$\Rightarrow$ answer $\sim \log(|\mathbf{dom}_T H|)$

Counting

Example: “is the number of states containing a
greater that the number of states containing b?”

$\Rightarrow$ we need $\Omega(\log(|\mathbf{dom}_T|))$ space to represent counter(s)
$\Rightarrow$ $\mathcal{O}(\log(|\mathbf{dom}_T|))$ is sufficient.

Conjecture: we can use the above technique (but remember counts of the expired values) to answer queries with counting

$\Rightarrow$ $\mathcal{E}_Q(H)| \leq POLY(\log(|\mathbf{dom}_T|))$

Duplicates

Example (in SQL-style syntax):

```
( select '1'
  from   R
  where  R.x='a' ) except all ( select '1'
                                from   R
                                where  R.x='b' )
```

is nonempty if and only if the number of states containing a
is greater that the number of states containing b

$\Rightarrow$ just like counting . . .

Retroactive Updates

Example:

while $\exists t.R(t,a) \wedge \exists t.R(t,b)$ **do**   **{** while both a and b exist in R **}**
    delete $R(t,a)$
        where $\forall t'.R(t',a) \supset t' > t$;   **{** delete (chronologically) first a
    delete $R(t,b)$
        where $\forall t'.R(t',b) \supset t' > t$;   **{** delete (chronologically) first b
return $\exists t.R(t,a)$ **{** return true if R contains an a

$\Rightarrow$ we need $\Omega(\log(|\mathbf{dom}_T|))$ space to represent counter(s)

$\Rightarrow$ just like for counting . . .

Infinite Histories

Definition 6 *Let H be a finite history, Q a query (in an appropriate query language), and θ a substitution.*

- *θ is a potential answer for Q with respect to H if there is an infinite completion H' of H such that $H', \theta \models Q$.*
- *θ is a certain answer for Q with respect to H if for all infinite completions H' of H we have $H', \theta \models Q$.*

The notion of *potential answer* is a direct generalization of the notion of *potential constraint satisfaction* [Chomicki, 1995].

Infinite Histories (cont.)

Proposition 7 ([Gabbay et al., 1994]) *The satisfaction problem for two dimensional propositional temporal logic over natural numbers-based time domain is not decidable.*

Proposition 8 ([Chomicki, 1995]) *For past formulas potential constraint satisfaction is undecidable.*

Proposition 9 ([Chomicki and Niwinski, 1995]) *For biquantified formulas with no internal quantifiers (called* universal*), potential constraint satisfaction is decidable (in exponential time). For biquantified formulas with a single internal quantifier, potential constraint satisfaction is undecidable.*

Related Issues

- garbage collection in programming languages
- temporal/dynamic integrity constraint enforcement
- model checking
- materialized view maintenance

  $\Rightarrow$ self-maintainable views and expiration for SAGAs

Open Problems

FutureTL. expiration operator for full FutureTL

$\Rightarrow$ combined Past-Future TL?

Fixpoints in 2-FOL.

Rich Temporal Domains. more than linear $\leq$

$\Rightarrow$ constraint database techniques? [Libkin et al., 2000]

Space Bounds For Aggregate Queries.

$\Rightarrow$ a weaker bound, e.g., $|\mathcal{E}H| \in O(\log(|\mathbf{dom}_T H|))$?

Query Languages with Decidable Potential Answers.

$\Rightarrow$ Optimal Expiration Operators?

Acknowledgment

- Part of this research was done while visiting

  BRICS

  Centre for Basic Research in Computer Science
  funded by the Danish National Science Foundation.

- The research was supported by the National Sciences and Engineering Research Council of Canada (NSERC).

- The material (pending revisions) will appear in

  J. Chomicki, G. Saake, and R. van der Mayden:
  Logics for Emerging Applications of Databases, Springer '03.

`http://db.uwaterloo.ca/~david/book-lead.ps`

References

[Abiteboul et al., 1996] Abiteboul, S., Herr, L., and Van den Bussche, J. (1996). Temporal Versus First-Order Logic to Query Temporal Databases. In *ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, pages 49–57.

[Chomicki, 1995] Chomicki, J. (1995). Efficient Checking of Temporal Integrity Constraints Using Bounded History Encoding. *TODS*, 20(2):149–186.

[Chomicki and Niwinski, 1995] Chomicki, J. and Niwinski, D. (1995). On the Feasibility of Checking Temporal Integrity Constraints. *Journal of Computer and System Sciences*, 51(3):523–535.

[Chomicki and Toman, 1998] Chomicki, J. and Toman, D. (1998). Temporal Logic in Information Systems. In Chomicki, J. and Saake, G., editors, *Logics for Databases and Information Systems*, pages 31–70. Kluwer.

[Gabbay et al., 1994] Gabbay, D. M., Hodkinson, I. M., and Reynolds, M. (1994). *Temporal Logic: Mathematical Foundations and Computational Aspects*. Oxford University Press.

[Jensen, 1995] Jensen, C. S. (1995). Vacuuming. In Snodgrass, R. T., editor, *The TSQL2 Temporal Query Language*, pages 447–460.

[Libkin et al., 2000] Libkin, L., Kuper, G., and Paredaens, J., editors (2000). *Constraint Databases*. Springer.

[Lipeck et al., 1994] Lipeck, U. W., Gertz, M., and Saake, G. (1994). Transitional Monitoring of Dynamic Integrity Constraints. *IEEE Data Engineering Bulletin*.

[Lipeck and Saake, 1987] Lipeck, U. W. and Saake, G. (1987). Monitoring Dynamic Integrity Constraints Based on Temporal Logic. *Information Systems*, 12(3):255–269.

[Toman and Niwinski, 1996] Toman, D. and Niwinski, D. (1996). First-Order Queries over Temporal Databases Inexpressible in Temporal Logic. In *Advances in Database Technology, EDBT'96*, volume 1057, pages 307–324. Springer.

[Yang and Widom, 1998] Yang, J. and Widom, J. (1998). Maintaining Temporal Views over Non-Temporal Information Sources for Data Warehousing. In *Advances in Database Technology, EDBT'98*, pages 389–403.

[Yang and Widom, 2000] Yang, J. and Widom, J. (2000). Temporal View Self-Maintenance. In *Advances in Database Technology, EDBT'00*, pages 395–412.