Modeling Datalog Fact Assertion and Retraction in Linear Logic

Edmund S. L. Lam and Iliano Cervesato
Carnegie Mellon University, Qatar

1. Introducing Datalog and Deductive Databases

A Logic Programming Language for Deductive Databases.

- An Example: Graph relation, let E be Edge and P be Path,

\[ P = \{ r_1 : P(x, y) \rightarrow E(x, y), r_2 : P(x, z) \rightarrow E(x, y), P(y, z) \} \]

- Assertion of new facts:

\[ E(2, 3), P(2, 3), E(3, 4), P(3, 4) \]
\[ \Rightarrow E(2, 3), P(2, 3), E(3, 4), P(3, 4), P(4, 2) \]
\[ \Rightarrow E(2, 3), P(2, 3), E(3, 4), P(4, 3), P(4, 2) \]

- Retraction of facts:

\[ P(2, 3), E(3, 4), P(4, 4), P(4, 2) \]
\[ \Rightarrow E(2, 3), P(4, 3), P(4, 2) \]
\[ \Rightarrow P(4, 3), P(4, 2) \]

- Over recent ten years, Datalog has been applied to new domains, e.g.:
  - Implementing network protocols [GWL01, LCG+06]
  - Distributed ensemble programming [ARLG+09]
  - Deductive spreadsheets [Cer07]
  - Main challenge and focus so far:
    - Maintaining recursive views in presence of assertion and retraction.
    - Efficient algorithms and implementations are well-known [ARLG+09, CARG+12, GMS93, LCG+06]

2. Traditional Logical Interpretation of Datalog

- First order logic interpretation:

\[ P = \{ r_1 : \forall x, y, E(x, y) \Rightarrow P(x, y), r_2 : \forall x, y, z, E(x, y) \land P(y, z) \Rightarrow P(x, z) \} \]

- Assertion: Forward chain application of implications, until saturation, e.g. adding of new base fact \( E(3, 4) \):

\[ P, E(2, 3), P(2, 3), E(3, 4), P(3, 4), P(4, 4) \Rightarrow C \]
\[ P, E(2, 3), P(2, 3), E(3, 4), P(4, 4), P(4, 2) \Rightarrow C \]

- But what about retraction? E.g. removal of fact \( E(2, 3) \):

\[ P, E(2, 3), P(2, 3), E(3, 4), P(4, 4), P(4, 2) \Rightarrow C \]

3. Our Objective

- To define a logical specification of Datalog that supports assertion and retraction internally.
- Our Solution: Define a Linear Logic [Gir87] Interpretation of Datalog.
- Linear logic because:
  - Assumptions can grow or shrink as inference rules apply.
  - Facts are not permanent truths, but can be retracted (consumed)

4. Linear Logic Interpretation of Datalog

Example: Linear logic interpretation (simplified) of the Graph program:

\[ P = \{ r_1 : P(x, y) \rightarrow E(x, y), r_2 : P(x, z) \rightarrow E(x, y), P(y, z) \} \]

- Absorption rules:

\[ A_F = \{ E(x, y) \Rightarrow (E(x, y) \rightarrow E(x, y)), P(x, y) \Rightarrow P(x, y) \rightarrow E(x, y) \} \]

- Program interpretation denoted as:

\[ P = \forall x, y, z. \overline{P}(x, y, z) \]

5. Datalog Assertion in Linear Logic Interpretation

- Two-sided intuitionistic linear logic sequent calculus, \( \Lambda^\text{obs} : \Delta \rightarrow C \)
- Assumption, e.g. adding of new base fact \( E(3, 4) \):

\[ P, E(2, 3), P(2, 3), E(3, 4), P(3, 4), P(4, 4) \Rightarrow C \]

- Similar to traditional logic interpretation, Datalog assertions map to forward chaining fragment of Linear Logic proof search.
- Key difference: Inference of new facts leaves behind “bookkeeping” information:
  - Specifically retraction rules \( R_{\Delta(3,3)}, z_{\Delta(3,3)}, \ldots \)
  - Act as “cookie crumbs” that guides retraction

6. Datalog Retraction in Linear Logic Interpretation

- Retraction, e.g. removal of fact \( E(2, 3) \):

\[ P, E(2, 3), P(2, 3), E(3, 4), P(4, 4), P(4, 2) \Rightarrow C \]

7. Completeness and Soundness Results

- Define \( \Delta \Rightarrow^+ \Delta' \) as an abstract state transition system that computes inference closures of Datalog states \( \Delta \).
- We define this, based on linear logic proof search:

\[ \Delta \not\subseteq \Delta' \Rightarrow \Delta' \Rightarrow \Delta \Rightarrow \Delta' \Rightarrow \Delta' \Rightarrow \Delta' \rightarrow \text{Quiescent}(\Delta', (\Delta, \Delta')) \text{ (Infer)} \]
\[ \Delta \subseteq \Delta' \Rightarrow \Delta' \Rightarrow \Delta \Rightarrow \Delta' \Rightarrow \Delta' \rightarrow \text{Quiescent}(\Delta', (\Delta, \Delta')) \text{ (Retract)} \]

- Technical hurdles that we had to overcome to achieve this:
  - Trivial non-termination in assertions.
  - In-exhaustive retraction.
- Correctness and Soundness of assertion and retraction: Given a Datalog Program \( P \), for reachable states \( \Delta_1, \Delta_2, \Delta_3, \Delta_4 \) such that \( \Delta_1 \Rightarrow \Delta_2 \Rightarrow \Delta_3 \rightarrow \Delta_4 \rightarrow \Delta_4 \) then we have the following:

\[ \Delta_1 \Rightarrow \Delta_2 \Rightarrow \Delta_3 \Rightarrow \Delta_4 \Rightarrow \Delta_4 \text{ if } \Delta_4 \approx P(\Delta_4) \]

- See our PPDP’12 paper or tech report (CMU-CS-12-126) for details.

8. Contributions and Future Works

- So why do we need a linear logic interpretation of Datalog?
- We’ve got a few reasons:
  - Provide a refined logical understanding of Datalog assertion and retraction, hence we can prove properties of Datalog programs via theorem provers (e.g. CLF).
  - Provide an operational semantics of Datalog style assertion and retraction based on higher order, forward chaining multiset rewrite rules.
  - Provide a cleaner and more theoretically well-founded way of implementing and reasoning about modern extensions of Datalog (e.g. Meld [ARLG+09], Dedalus [AMC+09], Distributed Datalog [NLS11]).
- Future Works:
  - Implementation of Datalog based on higher order multiset rewritings.
  - Refine our linear logic interpretation.

- Funded by the Qatar National Research Fund as project NPRP 09-6671-100 (Effective Programming for Large Distributed Ensembles)
- In proceedings of 14th International Symposium on Principles and Practice of Declarative Programming (PPDP’12)

http://www.qatar.cmu.edu/~sllam/