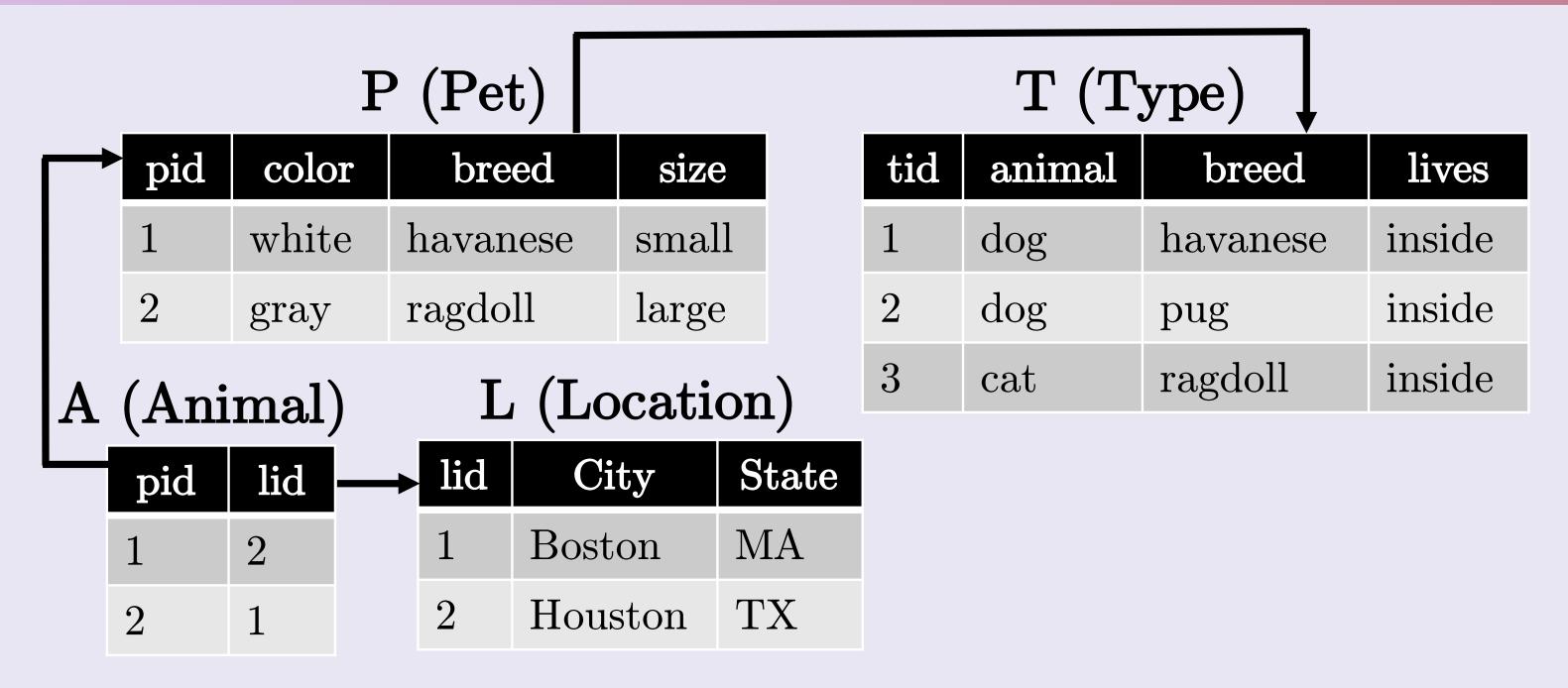
Accelerating query evaluation using multi-query prediction



Anna Fariha Tony Ohmann Yuriy Brun Alexandra Meliou {afariha, ohmann, brun, ameli}@cs.umass.edu

Motivation: Faster query evaluation via prediction



Timestamp	Query
6:00:00	SELECT DISTINCT animal FROM T
6:00:02	SELECT DISTINCT breed FROM T WHERE animal = 'dog'
6:00:03	SELECT * FROM P WHERE breed='pug'
?	What is the next query?

Existing approaches:

- Caching common queries can accelerate evaluation but ignores temporal information captured by sequences of queries.
- Temporal query information has been used for for auto completion [3] and query recommendation [2] but not for improving query evaluation efficiency.

Observations:

- There is redundancy in the patterns of queries users and applications submit.
- If the DBMS can partially predict upcoming queries, it can precompute queries and improve evaluation efficiency by rewriting queries to use precomputed results.

QUERYPREDICT approach Time Use Synoptic [1] to infer a probabilistic FSM model of past sequences of queries. past users' Infer query prediction FSM query workloads > Find paths in the prediction FSM to locate user's current possible states prediction model from recent queries. Compute path probability from current user's Predict upcoming queries current possible states to next states to recent queries predict user's likely next states. list of predicted queries Abstract common parts of the likely Find common query using next queries and precompute results multi-query optimization [5] that benefit evaluation speed of those queries. predicted common query Materialize view for predicted common query view > This rewritten query is equivalent to the original query but performs faster. DBMS Rewrite actual query using current user's

Example of accelerating query evaluation

Predicted common query: SELECT * FROM P,A,L WHERE P.pid = A.pid AND A.lid = L.lid ORDER BY P.breed

Materialized view: CREATE view V as SELECT * FROM P,A,L WHERE P.pid = A.pid AND A.lid = L.lid ORDER BY P.breed

Actual query: SELECT * FROM P,A,L WHERE P.breed = 'lab' AND P.pid = A.pid AND A.lid = L.lid AND state = 'MA'

Rewritten query with materialized view: SELECT * FROM V WHERE breed = 'lab' AND state = 'MA'

References

next query

[1] I. Beschastnikh, Y. Brun, S. Schneider, M. Sloan, and M. Ernst. Leveraging existing instrumentation to automatically infer invariant-constrained models. In ESEC/FSE, 2011.

rewritten query

- [2] G. Chatzopoulou, M. Eirinaki, and N. Polyzotis. Query recommendations for interactive database exploration. In SSDBM, 2009.
- [3] N. Khoussainova, Y. Kwon, M. Balazinska, and D. Suciu. Snipsuggest: Context-aware autocompletion for sql. In VLDB, 2010.
- [4] R. Pottinger and A. Y. Levy. A scalable algorithm for answering queries using views. In VLDB, 2000.

materialized views [4]

[5] P. Roy, S. Seshadri, S. Sudarshan, and S. Bhobe. Efficient and extensible algorithms for multi query optimization. In SIGMOD, 2000.