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

<table>
<thead>
<tr>
<th>P (Pet)</th>
<th>T (Type)</th>
</tr>
</thead>
<tbody>
<tr>
<td>pid</td>
<td>color</td>
</tr>
<tr>
<td>1</td>
<td>white</td>
</tr>
<tr>
<td>2</td>
<td>gray</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>A (Animal)</th>
<th>L (Location)</th>
</tr>
</thead>
<tbody>
<tr>
<td>pid</td>
<td>lid</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>

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

- past users’ query workloads
- current user’s recent queries
- current user’s next query

Infer query prediction FSM

- Predict upcoming queries
- Find common query using multi-query optimization [5]
- Materialize view for predicted common query
- Rewrite actual query using materialized views [4]

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 ORDER BY P.breed
```

Rewritten query with materialized view:

```
SELECT * FROM V WHERE breed = 'lab' AND state = 'MA'
```

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 ORDER BY P.breed
```

Rewritten query with materialized view:

```
SELECT * FROM V WHERE breed = 'lab' AND state = 'MA'
```

References