Database interrogation using conjunctive queries

Michał Bielecki
Warsaw, Poland

Jan Van den Bussche
Limburg, Belgium

Database interrogation

Client accesses server using query language L

By posing **sequence** of queries, client can obtain answer to queries not expressible in L

1. **database-independent:** fixed sequence
2. **adaptive:** next query depends on answer to previous one

Client-server computing, communication aspects of database querying

Security

Information integration

Conjunctive queries

We study case where $L =$ conjunctive queries and variations:

$$
\begin{array}{ll}
\mathsf{CQ}: & (x) \leftarrow R(x,y), R(y,z) \\
\mathsf{CQ}(\neq): & (x) \leftarrow R(x,y), R(y,z),\ x \neq y \\
\mathsf{CQ}(\neg): & (x) \leftarrow R(x,y),\ \neg S(y) \\
\mathsf{qfCQ}: & (x,y,z) \leftarrow R(x,y), R(y,z)
\end{array}
$$

qfFO: unions of qfCQ($\neg$) queries

The basic question

What queries can be computed by database interrogation using conjunctive queries?

1. database-independent

2. adaptive

Problem: by downloading entire database, client can compute everything locally!

$$(x, y) \leftarrow R(x, y)$$

Refined questions

What queries can be computed by database interrogation using

- **boolean** *conjunctive queries?*

- **counts of** *conjunctive queries?*

Results for database-independent clients

Notation:

- $L_1 \leqslant L_2$ means that any L_1-query can be derived by a database-independent client using L_2-queries

- $L_1 = L_2$ if $L_1 \leqslant L_2$ and $L_2 \leqslant L_1$

- $L_1 < L_2$ if $L_1 \leqslant L_2$ but not $L_2 \leqslant L_1$

Theorem:

$$\#\mathrm{qfCQ} = \#\mathrm{qfFO} < \#\mathrm{CQ} < \#\mathrm{CQ}(\neq)$$

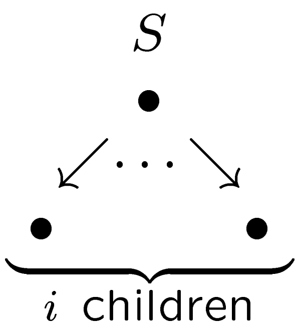
$\#\mathrm{CQ}(\neq) < \#\mathrm{CQ}(\neg)$ is conjectured but open

Proof of #qfCQ $<$ #CQ

We show that $\#Q$ is not derivable from database-independent sequence of #qfCQ queries, with:

$$Q(x) \leftarrow S(x), R(x, y)$$

Use disjoint unions of **stars** G_i:



Database A_n: $\binom{n}{i}$ copies of G_i for even $i \leqslant n$
B_n: odd

For any $j < n$, number of embedded G_j's is same in A_n and B_n, so indistinguishable using #qfCQ

However, $\#Q(A_n) \neq \#Q(B_n)$

The adaptive model

Reflective relational machine [APV]: Turing machine communicating with database as an oracle

RRMs using boolean FO-queries can reconstruct isomorphic copy of database

Theorem: Already possible using just CQ($\neq$) queries

Corollary: An adaptive client using #qfCQ queries can compute **any** computable natural number query

Contrast database-independent clients!

RRMs with boolean CQs

Can compute only queries closed under CQ-equivalence.

All?

No: "R is CQ-equivalent to a simple cycle" is not computable by RRM(CQ)

Nonetheless, various non-trivial queries computable:

- length of shortest cycle
- length of longest chain

Characterize power of RRM(CQ)?

Proof "CQ-equivalent to simple cycle" not in RRM(CQ)

Consider RRM(CQ) running on a 2-cycle

Let n be largest number of vars used in run

Add a p-cycle, $p > n$ and odd:

- Indistinguishable from 2-cycle using n-variable CQs

- Not equivalent to a simple cycle

Conclusion

Study communication aspects of database querying!

- this work
- query rewriting, information integration
- Web browsing
- peer-to-peer