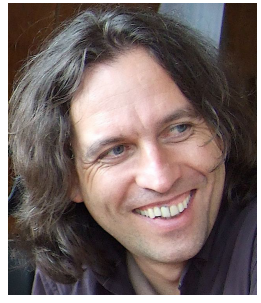
Parallel-Correctness and Transferability for Conjunctive Queries

Tom J. Ameloot[1] Gaetano Geck[2] Bas Ketsman[1]
Frank Neven[1] Thomas Schwentick[2]







[1] Hasselt University [2] Dortmund University

Big Data

"Too large for one server"

Several systems: Hadoop, Spark, . . . many others

Common Strategy

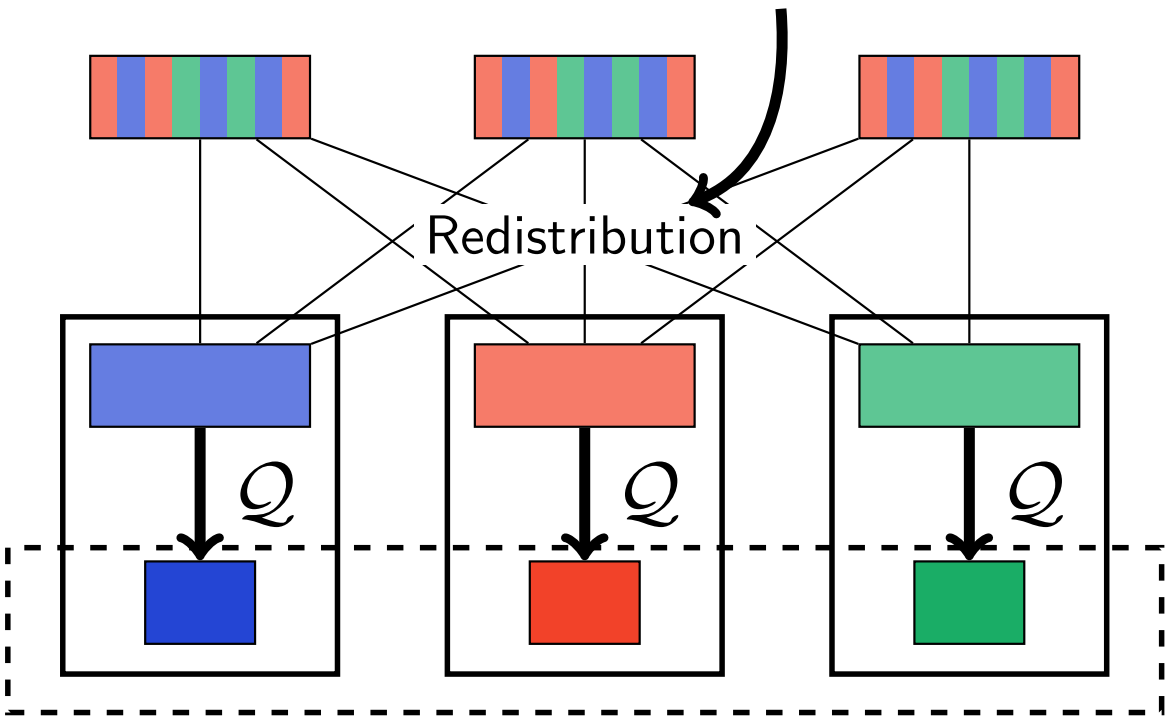
- Data is distributed
- Query evaluation:
  Multiple rounds with reshuffling

Simple Evaluation Algorithm

1-Round MPC model [Koutris & Suciu 2011]

Input = query $\mathcal{Q}$

Modeled by a distribution policy $\boldsymbol{P}$



Output = union of output at each server

Main Problems

Semantical correctness:

When is the simple algorithm correct on a distribution policy?

Parallel-Correctness

Multiple-query optimization:

Which queries allow to reuse the distribution obtained for another query?

Transferability

Formal framework for reasoning about correctness of query evaluation and optimization in a distributed setting

Outline

1. Definitions
2. Parallel-Correctness
3. Transferability
4. Lowering the Complexity
5. Conclusion & Future Work

Definitions

Database schema

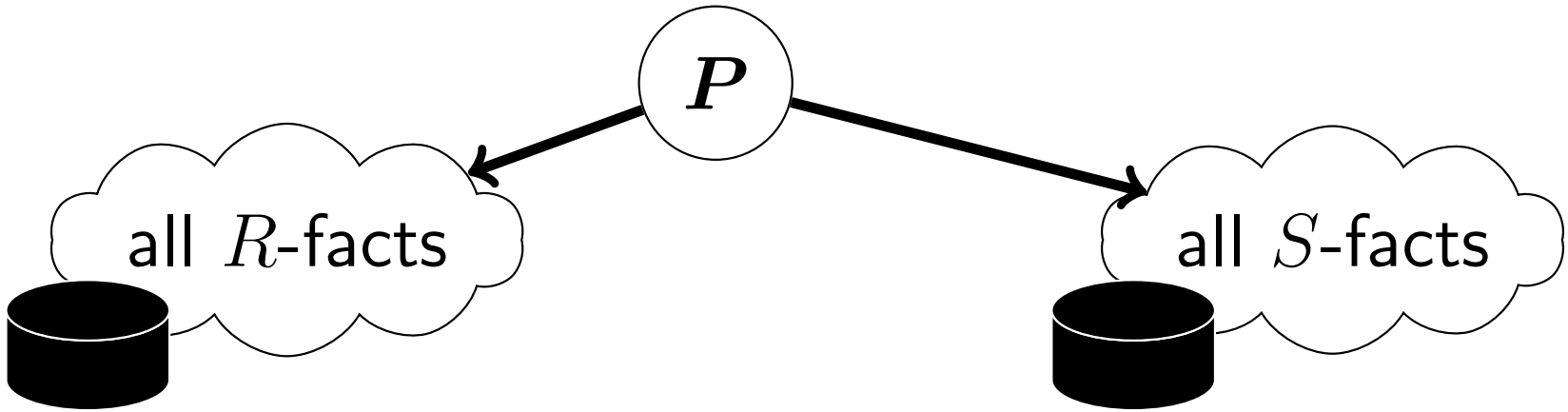
Infinite set of data values

Instance I is a finite set of facts $R(d_1, \ldots, d_n)$

Conjunctive Query: $T(\bar{x}) \leftarrow R_1(\bar{y}_1), \ldots, R_m(\bar{y}_m)$

Distribution Policies

Network $\mathcal{N}$ is a finite set of nodes

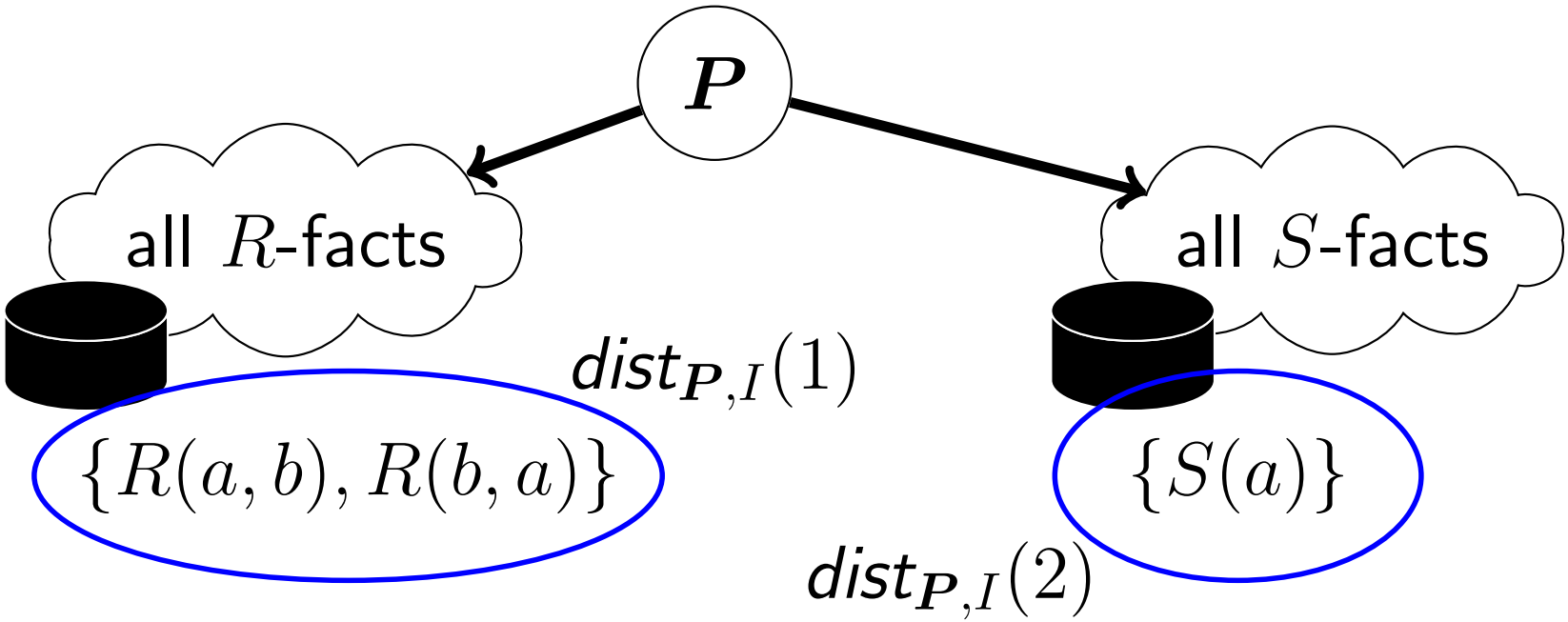


Definition

A distribution policy $\boldsymbol{P}$ is a total function mapping facts (over **dom**) to sets of nodes in $\mathcal{N}$

Distribution Policies

Network $\mathcal{N}$ is a finite set of nodes



$=$ distribution of I based on $\boldsymbol{P}$

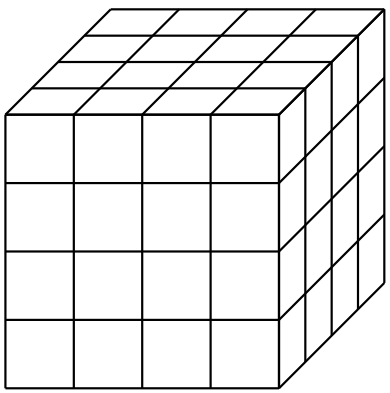
Instance $I = \{R(a,b), R(b,a), S(a)\}$

Hypercube

- Invented in the context of Datalog evaluation
  [Ganguli, Silberschatz & Tsur 1990]
- Described in Map-Reduce context
  [Afrati & Ullman 2010]
- Intensively studied
  [Beame, Koutris & Suciu 2014]

Algorithm:

- Reshuffling based on structure of $\mathcal{Q}$



Partitioning of complete valuations over servers in instance independent way through hashing of domain values

Simple Evaluation Algorithm

Input $=$ query $\mathcal{Q}$

Step 1: distribute data over servers w.r.t. $\boldsymbol{P}$

Step 2: evaluate $\mathcal{Q}$ at each server

Parallel-Correctness

Definition

$\mathcal{Q}$ is parallel-correct on I w.r.t. $\boldsymbol{P}$, iff

$$\mathcal{Q}(I) = \bigcup_{\kappa \in \mathcal{N}} \mathcal{Q}(\mathit{dist}_{\boldsymbol{P},I}(\kappa))$$

$\supseteq$ by monotonicity

Definition (w.r.t. all instances)

$\mathcal{Q}$ is parallel-correct w.r.t. $\boldsymbol{P}$ iff
$\mathcal{Q}$ is parallel-correct w.r.t. $\boldsymbol{P}$ on every I

Parallel-Correctness
Sufficient Condition

(C0) for every valuation V for $\mathcal{Q}$,

$$\bigcap_{f \in V(\text{body}_{\mathcal{Q}})} \boldsymbol{P}(f) \neq \emptyset.$$

Intuition: Facts required by a valuation meet at some node

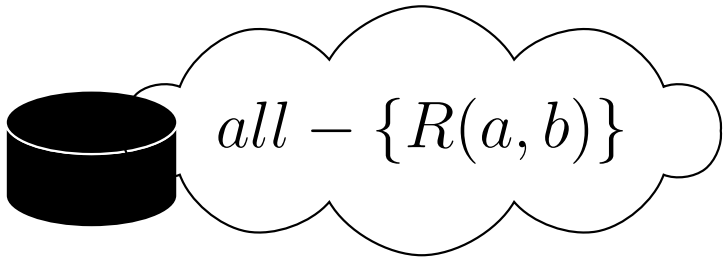
Lemma

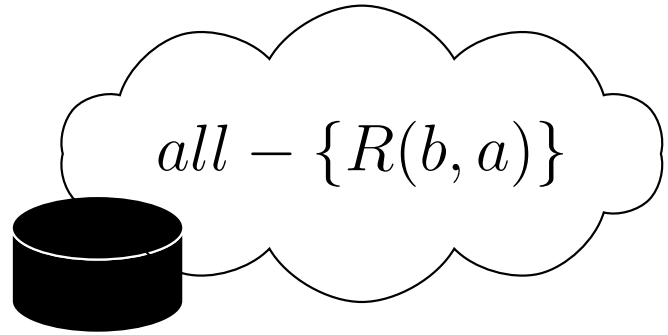
(C0) implies $\mathcal{Q}$ parallel-correct w.r.t. $\boldsymbol{P}$.

Not necessary

(C0) not Necessary
Example

Distribution policy P





Query $\mathcal{Q}$: $T(x,z) \leftarrow R(x,y), R(y,z), R(x,x)$

$V = \{x, z \to a, y \to b\}$

Requires:

$R(a,b)$ $R(b,a)$ $R(a,a)$

Derives:

$T(a,a)$

Do not meet

$V' = \{x, y, z \to a\}$

Requires:

$R(a,a)$

Derives:

$T(a,a)$

$\supsetneq$ $=$

Parallel-Correctness
Characterization

Lemma

$\mathcal{Q}$ is parallel-correct w.r.t. $\boldsymbol{P}$ iff

(C1) for every minimal valuation V for $\mathcal{Q}$,

$$\bigcap_{\boldsymbol{f} \in V(\text{body}_{\mathcal{Q}})} \boldsymbol{P}(\boldsymbol{f}) \neq \emptyset.$$

Definition

V is minimal if no V' exists, where
$V'(\text{head}_{\mathcal{Q}}) = V(\text{head}_{\mathcal{Q}})$, $V'(\text{body}_{\mathcal{Q}}) \subsetneq V(\text{body}_{\mathcal{Q}})$.

Parallel-Correctness
Example

Query $\mathcal{Q}$: $T(x,z) \leftarrow R(x,y), R(y,z), R(x,x)$

$V = \{x, z \to a, y \to b\}$ Requires: $\boxed{R(a,b) \quad R(b,a) \quad R(a,a)}$ Derives: $\boxed{T(a,a)}$

$\supsetneq$

$V' = \{x, y, z \to a\}$ Requires: $\boxed{R(a,a)}$ Derives: $\boxed{T(a,a)}$

$=$

Notice: $\mathcal{Q}$ is minimal CQ

CQ is minimal iff injective valuations are minimal

Proposition

Testing whether a valuation is minimal is coNP-complete.

Parallel-Correctness
Complexity

Theorem

Deciding whether $\mathcal{Q}$ is parallel-correct w.r.t. $\boldsymbol{P}$ is Π_2^P-complete.

Proof:

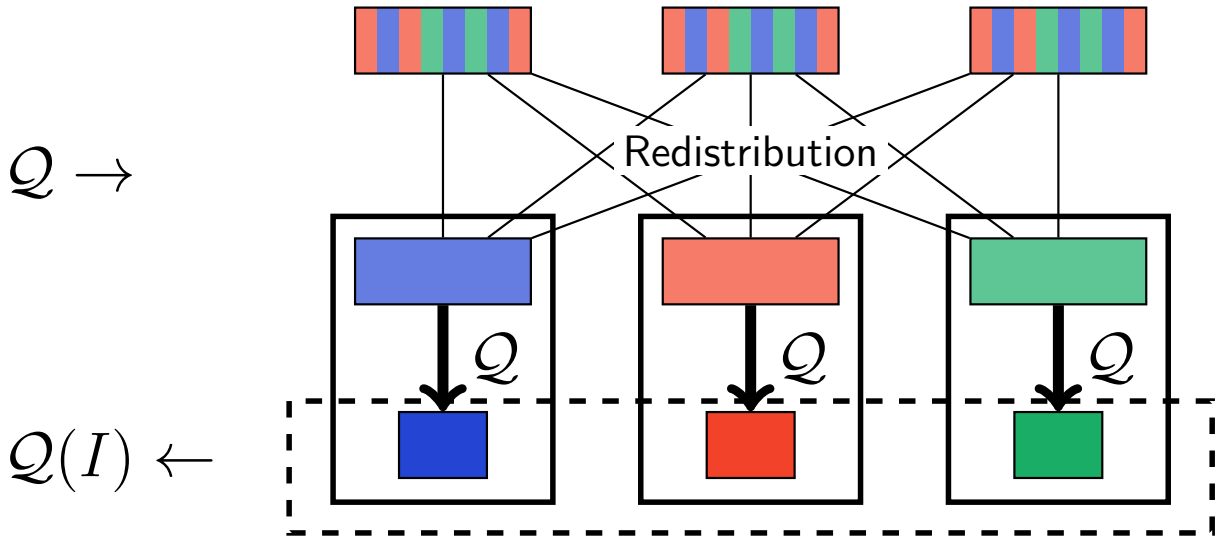
- Lower bound: Reduction from Π_2-QBF
- Upper bound: Characterization

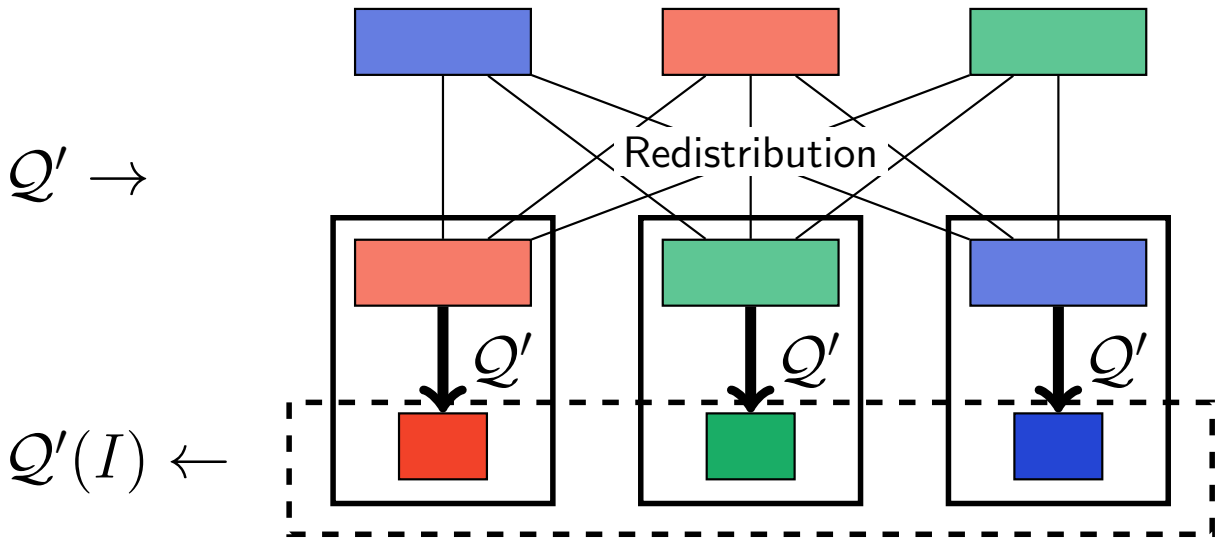
  but, requires proper formalization of P

Outline

1. Definitions
2. Parallel-Correctness
3. Transferability
4. Lowering the Complexity
5. Conclusion & Future Work

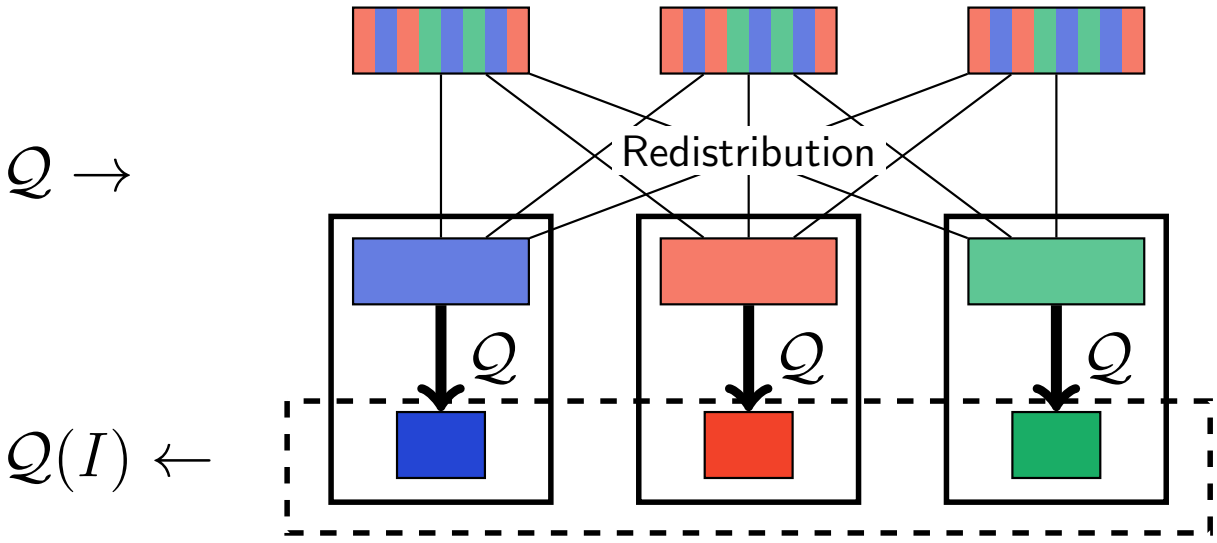
Computing Multiple Queries



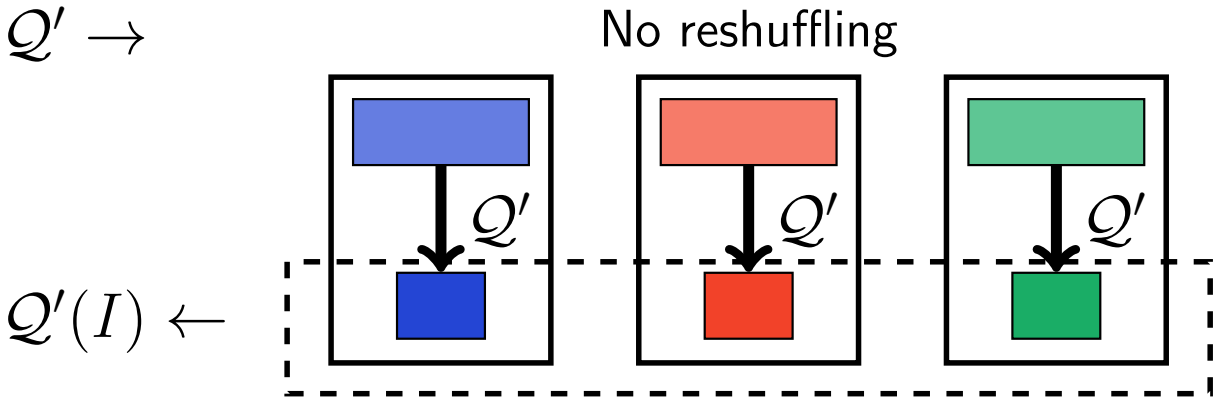


...

Computing Multiple Queries



When can $\mathcal{Q}'$ be evaluated on distribution used for $\mathcal{Q}$?



...

Transferability

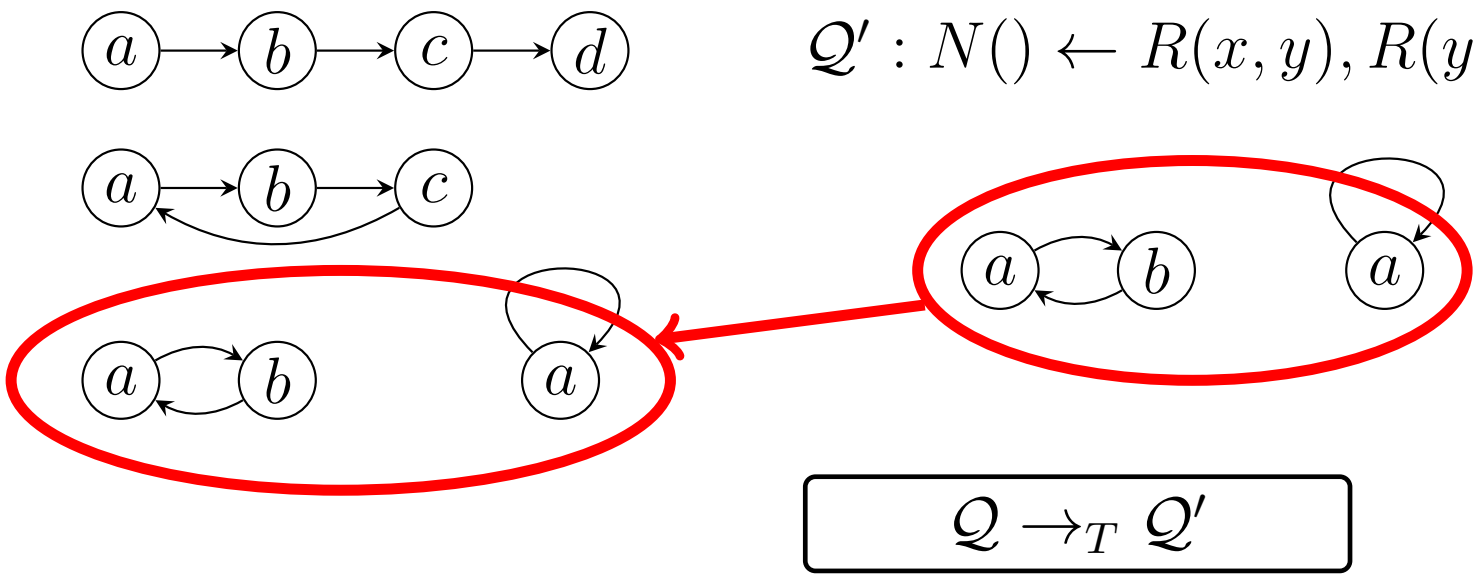
Definition

$\mathcal{Q} \rightarrow_T \mathcal{Q}'$ iff $\mathcal{Q}'$ is parallel-correct on every $\boldsymbol{P}$ where $\mathcal{Q}$ is parallel-correct on

Example

$\mathcal{Q} : T() \leftarrow R(x, y), R(y, z), R(z, w)$

$\mathcal{Q}' : N() \leftarrow R(x, y), R(y, x)$



$\mathcal{Q} \rightarrow_T \mathcal{Q}'$

Transferability
Characterization & Complexity

Lemma

$\mathcal{Q} \rightarrow_T \mathcal{Q}'$ iff

(C2) for every minimal valuation V' for $\mathcal{Q}'$ there is a minimal valuation V for $\mathcal{Q}$, s.t.

$$V'(\text{body}_{\mathcal{Q}}) \subseteq V(\text{body}_{\mathcal{Q}}).$$

Based on query structure alone, not on distribution policies

Transferability
Characterization & Complexity

Lemma

$\mathcal{Q} \to_T \mathcal{Q}'$ iff

(C2) for every minimal valuation V' for $\mathcal{Q}'$ there is a minimal valuation V for $\mathcal{Q}$, s.t.

$$V'(\text{body}_{\mathcal{Q}}) \subseteq V(\text{body}_{\mathcal{Q}}).$$

Theorem

Deciding $\mathcal{Q} \to_T \mathcal{Q}'$ is Π_3^P-complete.

- Lower bound: Reduction from Π_3-QBF
- Upper bound: Characterization

Outline

1. Definitions
2. Parallel-Correctness
3. Transferability
4. Lowering the Complexity
5. Conclusion & Future Work

Strongly Minimal CQs

Definition

A CQ is strongly minimal if all its valuations are minimal

- Full-CQs
  $T(x, y) \leftarrow R(x, y), R(x, x)$
- CQs without self-joins
  $T() \leftarrow R(x, y), S(x, x)$
- Hybrids
  $T(y) \leftarrow R(x, y), R(x, x), R(z, x), S(z)$

A minimal CQ is not always strongly minimal

Strongly Minimal CQs

Lemma

Deciding whether $\mathcal{Q}$ is strongly minimal is coNP-complete

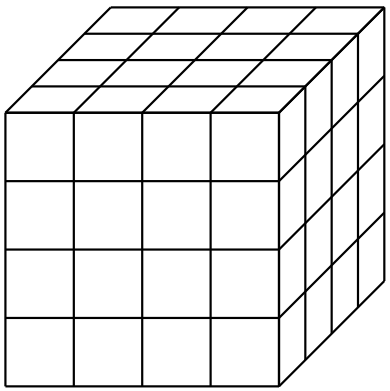
Theorem

Deciding $\mathcal{Q} \rightarrow_T \mathcal{Q}'$ is NP-complete for strongly minimal $\mathcal{Q}$

Hypercube

Algorithm:

- Reshuffling based on structure of $\mathcal{Q}$



Partitioning of complete valuations over servers in instance independent way through hashing of domain values

$\mathcal{H}(\mathcal{Q}) =$ family of Hypercube policies for $\mathcal{Q}$.

Definition

$\mathcal{Q} \rightarrow_H \mathcal{Q}'$ iff
$\mathcal{Q}'$ is parallel-correct w.r.t. every $\boldsymbol{P} \in \mathcal{H}(\mathcal{Q})$.

Hypercube

Two properties:

- $\mathcal{Q}$-**generous:** for every valuation facts meet on some node ($\forall \boldsymbol{P} \in \mathcal{H}(\mathcal{Q})$)
- $\mathcal{Q}$-**scattered:** there is a policy scattering facts in such a way that no facts meet by coincidence ($\forall I$)

Theorem

Deciding whether $\mathcal{Q} \rightarrow_H \mathcal{Q}'$ is NP-complete

(also when $\mathcal{Q}$ or $\mathcal{Q}'$ is acyclic)

Related Concepts

Containment

$\mathcal{Q} \subseteq \mathcal{Q}'$

Lemma

Containment and transferability are incomparable

Determinacy (Data-Integration)

$\mathcal{Q}'(I) = \mathcal{Q}'(J)$ implies $\mathcal{Q}(I) = \mathcal{Q}(J)$, for every I, J

Lemma

Determinacy and transferability are incomparable

Summary

Formal framework for reasoning about correctness of query evaluation and optimization in a distributed setting

Main concepts:

- Parallel-correctness
- Transferability

Independent of expression mechanism

Future Work

Expression Formalism for distribution policies

- Other than Hypercube?

Distribution policy for set of queries

- Given CQ: which distribution policy?

Hypercube

- Given set of CQs: which distribution policy?

Open question

Future Work

Tractable Results

- Other classes of queries?
- Other families of distribution policies?

More expressive classes of queries

- This work: CQs
- FO: undecidable
- initial results: UCQs, CQs with negation