Weaker Forms of Monotonicity for Declarative Networking: a more fine-grained answer to the CALM-conjecture

Tom J Ameloot[1], Bas Ketsman[1], Frank Neven[1] and Daniel Zinn[2]

[1] Hasselt University & transnational University of Limburg [2] LogicBlox, Inc

universiteit hasselt

Introduction

- Declarative Networking: Datalog based languages for parallel and distributed computing
- Cloud-computing: Setting with asynchronous communication via messages which can be arbitrarily delayed but not lost
- CALM-conjecture: A query has a coordination-free and eventually consistent execution strategy iff the query is monotone

[Hellerstein, 2010]

CALM = Consistency And Logical Monotonicity

CALM-conjecture

- [Ameloot, Neven, Van den Bussche, 2011]: TRUE
  for a setting where nodes have **no** information about the distribution of facts
- [Zinn, Green, Ludäscher, 2012]: FALSE
  for settings where nodes **have** information about the distribution of facts
- **TRUE when also refining montonicity**

Summary

$\mathcal{F}_0$	=	$\mathcal{M}$	=	wILOG($\neq$)	$\supsetneq$	Datalog($\neq$)
$\subsetneq$		$\subsetneq$		$\subsetneq$		$\subsetneq$
$\mathcal{F}_1$	=	$\mathcal{M}_{distinct}$	=	SP-wILOG	$\supsetneq$	SP-Datalog
$\subsetneq$		$\subsetneq$		$\subsetneq$		$\subsetneq$
$\mathcal{F}_2$	=	$\mathcal{M}_{disjoint}$	=	semicon-wILOG$^\neg$	$\supsetneq$	semicon-Datalog$^\neg$
Coordination freeness		Monotonicity		Datalog + value invention		Datalog

CALM [Ameloot, Neven, Van den Bussche, 2011]

A query has a coordination-free and eventually consistent execution strategy
iff
the query is monotone

Theorem
$\mathcal{F}_0 = \mathcal{M}$

CALM Revision 1

A query has a coordination-free and eventually consistent execution strategy under distribution policies
iff
the query is domain-distinct-monotone

Theorem
$\mathcal{F}_1 = \mathcal{M}_{distinct}$

CALM Revision 2

A query has a coordination-free and eventually consistent execution strategy under domain-guided distribution policies
iff
the query is domain-disjoint-monotone

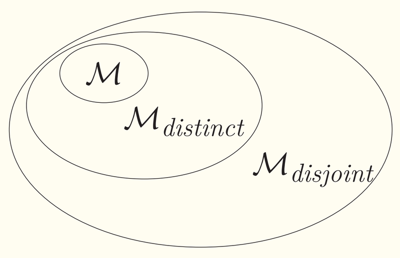
Theorem
$\mathcal{F}_2 = \mathcal{M}_{disjoint}$

Monotonicity

Definition
A query Q is monotone if $Q(\mathbf{I}) \subseteq Q(\mathbf{I} \cup \mathbf{J})$ for all database instances $\mathbf{I}$ and $\mathbf{J}$.

Notation
$\mathcal{M}$: class of monotone queries

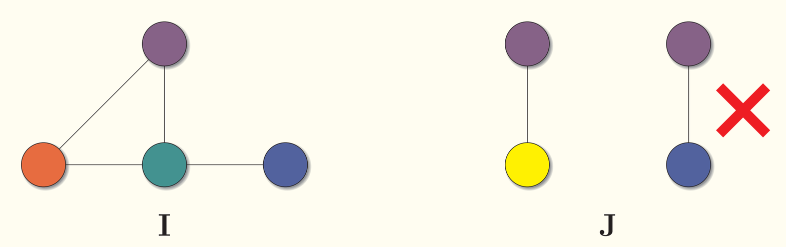
Example
- Select triangles in a graph $\in \mathcal{M}$
- Select open triangles in a graph $\notin \mathcal{M}$



Domain-distinct-monotonicity

- A fact $\mathbf{f}$ is domain distinct from instance $\mathbf{I}$ when $adom(\mathbf{f}) \not\subseteq adom(\mathbf{I})$.
- An instance $\mathbf{J}$ is domain distinct from instance $\mathbf{I}$ when every fact $\mathbf{f} \in \mathbf{J}$ is domain distinct from $\mathbf{I}$.

Example



Definition
A query Q is domain-distinct-monotone if $Q(\mathbf{I}) \subseteq Q(\mathbf{I} \cup \mathbf{J})$ for all $\mathbf{I}$ and $\mathbf{J}$ for which $\mathbf{J}$ is domain distinct from $\mathbf{I}$.

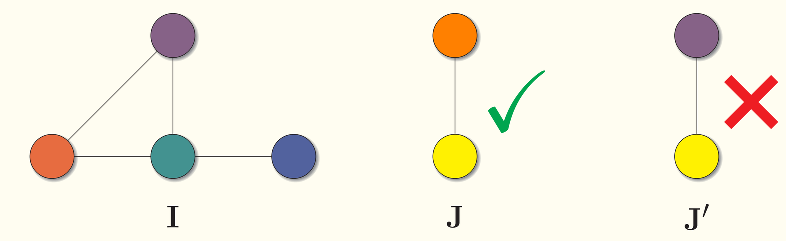
Notation
$\mathcal{M}_{distinct}$: class of domain-distinct-monotone queries

Remark
$\mathcal{M}_{distinct}$: class of queries preserved under extensions

Domain-disjoint-monotonicity

- An instance $\mathbf{J}$ is domain disjoint from instance $\mathbf{I}$ when $adom(\mathbf{I}) \cap adom(\mathbf{J}) = \emptyset$.

Example

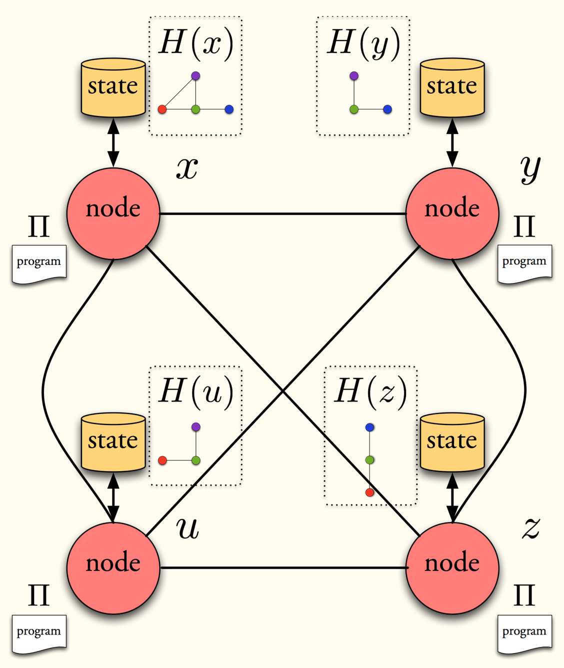


Definition
A query Q is domain-disjoint-monotone if $Q(\mathbf{I}) \subseteq Q(\mathbf{I} \cup \mathbf{J})$ for all $\mathbf{I}$ and $\mathbf{J}$ for which $\mathbf{J}$ is domain disjoint from $\mathbf{I}$.

Notation
$\mathcal{M}_{disjoint}$: class of domain-disjoint-monotone queries

Relational Transducer Networks

- Network $\mathcal{N} = \{x, y, u, z\}$
- Transducer Π
- messages can be arbitrarily delayed but never get lost



Semantics defined in terms of runs over a transition system

Definition
A transducer Π computes a query Q if
- for all networks $\mathcal{N}$, ← Network independent
- for all databases $\mathbf{I}$, ← Data distribution independent
- for all horizontal distributions H, and
- for every run of Π,

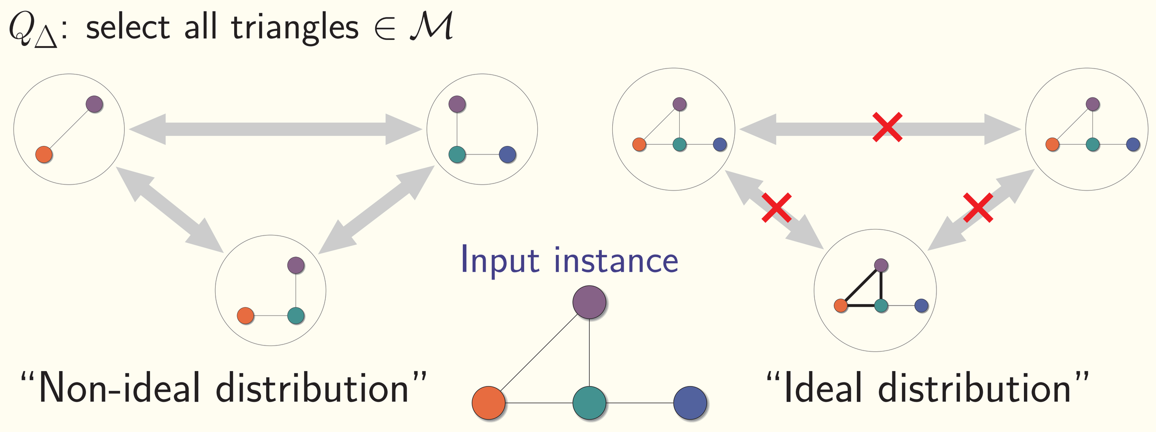
$$out(\Pi) = Q(\mathbf{I}).$$

Consistency requirement

Definition
Π is coordination-free if for all inputs $\mathbf{I}$ there is a distribution on which Π computes $Q(\mathbf{I})$ without having to do communication.

Goal: separate data-communication from coordination-communication

Example
Q_Δ: select all triangles $\in \mathcal{M}$



Definition
$\mathcal{F}_0$ = set of queries which are distributedly computed by coordination-free transducers.

Positive Datalog with inequalities

Datalog($\neq$) $\subsetneq$ wILOG($\neq$) = $\mathcal{M}$

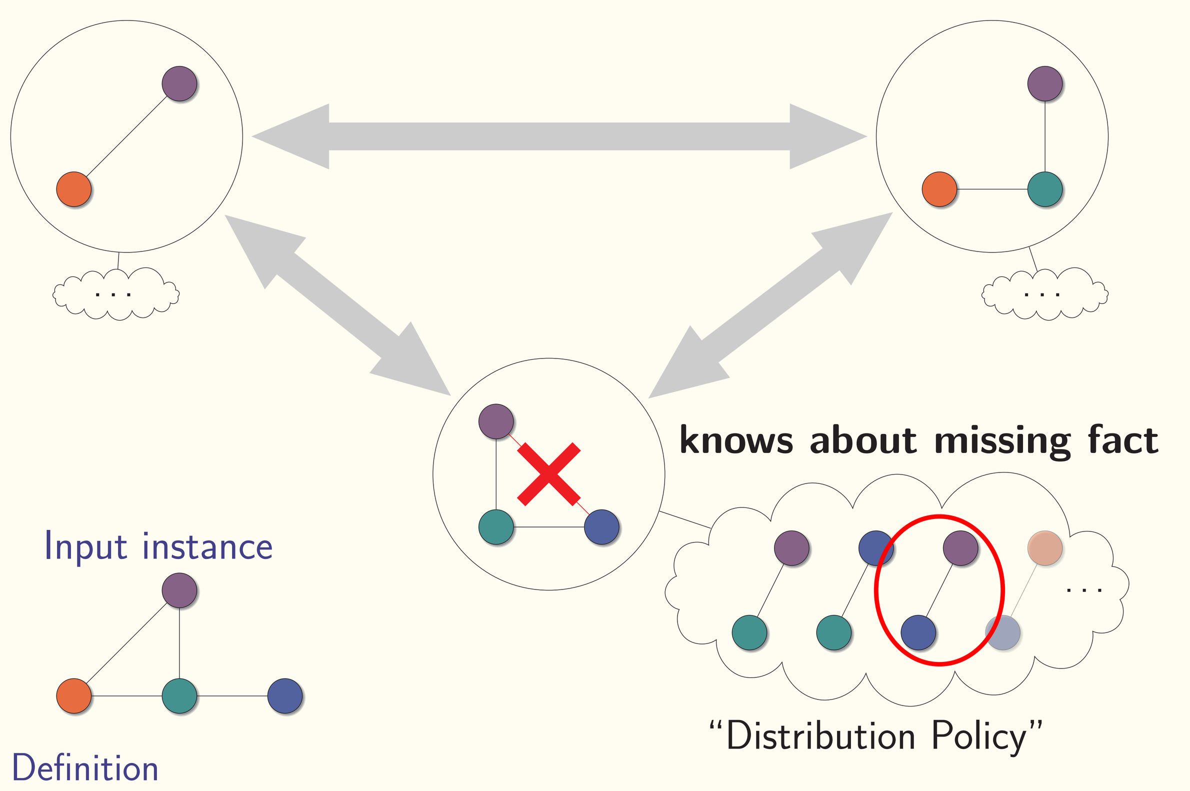
- Datalog($\neq$) $\subsetneq \mathcal{M} \cap$ PTIME

[Afrati, Cosmadakis, Yannakakis, 1994]

- wILOG($\neq$) = $\mathcal{M}$

[Cabibbo,1998]

Policy-aware Transducers [Zinn, Green, Ludäscher, 2012]



Definition
- A distribution policy $\mathbf{P}$ for σ and $\mathcal{N}$ is a total function from $facts(\sigma)$ to the power set of $\mathcal{N}$ (where $\mathbf{P}(\mathbf{f}) \neq \emptyset$, for every fact $\mathbf{f}$).
- A policy-aware transducer is a transducer with access to $\mathbf{P}$ restricted to its active domain.

Intuition
- every node is responsible for a predetermined set of facts
- every fact $\mathbf{f}$ in the global instance appears on all nodes responsible for $\mathbf{f}$

Definition
A policy-aware transducer Π computes a query Q if
- for all networks $\mathcal{N}$,
- for all databases $\mathbf{I}$,
- for all distribution policies $\mathbf{P}$, and
- for every run of Π,

$$out(\Pi) = Q(\mathbf{I}).$$

Definition
$\mathcal{F}_1$ = set of queries which are distributedly computed by policy-aware coordination-free transducers.

Proof of $\mathcal{M}_{distinct} \subseteq \mathcal{F}_1$

Observation
Let $\mathbf{I}$ be an instance, $C \subseteq adom(\mathbf{I})$.

Induced instance: $\mathbf{I}_{|C} = \{\mathbf{f} \in \mathbf{I} \mid adom(\mathbf{f}) \subseteq C\}$

By domain-distinct-monotonicity: $Q(\mathbf{I}_{|C}) \subseteq Q(\mathbf{I})$

C is complete at node x when x knows for every fact $\mathbf{f}$ with $adom(\mathbf{f}) \subseteq C$ whether $\mathbf{f} \in \mathbf{I}$ or $\mathbf{f} \notin \mathbf{I}$.

Algorithm
- broadcast all positive and negative facts
- Evaluate query on complete sets

Semi-positive Datalog

SP-Datalog $\subsetneq$ SP-wILOG = $\mathcal{M}_{distinct}$

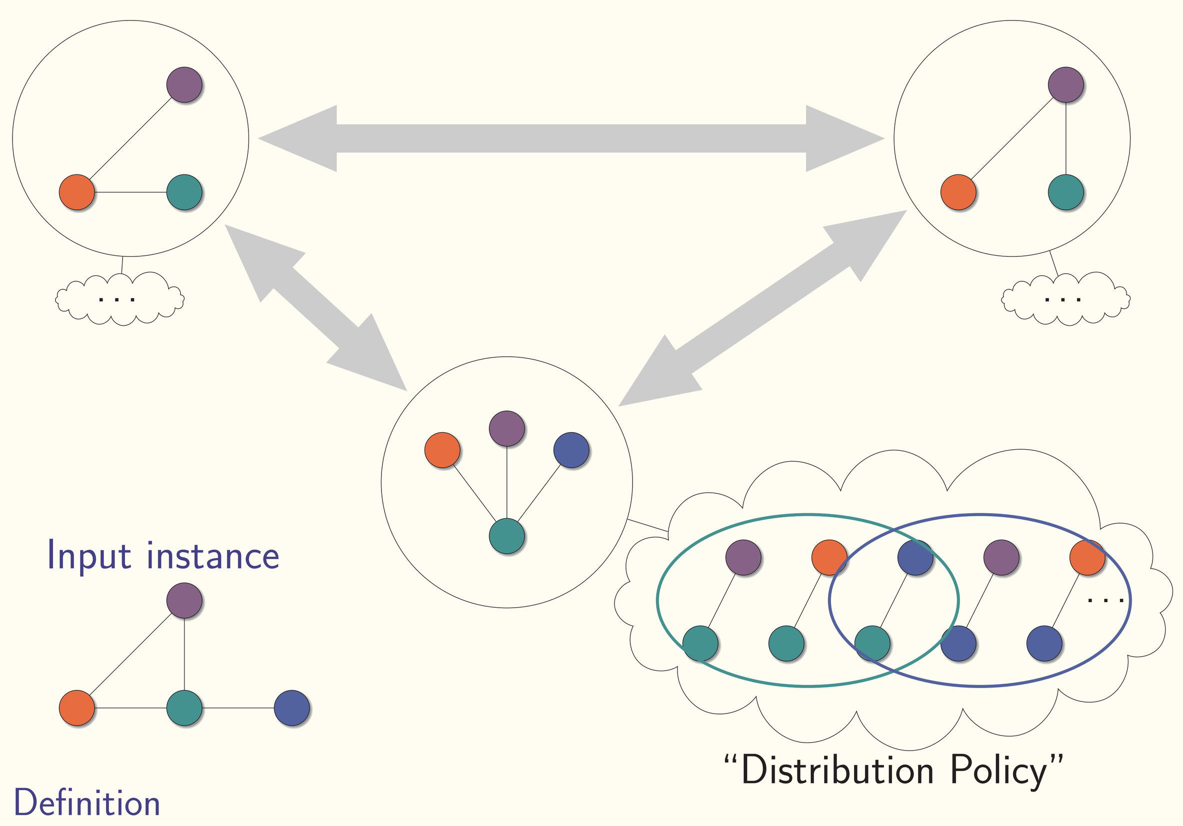
- SP-Datalog $\subsetneq \mathcal{M}_{distinct} \cap$ PTIME

[Afrati, Cosmadakis, Yannakakis, 1994]

- SP-wILOG = $\mathcal{M}_{distinct}$

[Cabibbo,1998]

Domain-guided Policies [Zinn, Green, Ludäscher, 2012]



Definition
- A domain assignment α for $\mathcal{N}$ is a total function from **dom** to the power set of $\mathcal{N}$.
- A distribution policy $\mathbf{P}$ is domain-guided if there is a domain assignment α where

$$\mathbf{P}(R(a_1, \ldots, a_k)) = \bigcup_{i=1}^{k} \alpha(a_i).$$

Intuition
- every node is responsible for a predetermined set of domain elements
- every fact $R(a_1, .., a_k)$ in the global instance appears on all nodes responsible for at least one value a_i (with $i \in \{1, \ldots, k\}$)

Definition
$\mathcal{F}_2$ = queries which are distributedly computed under domain-guided distribution policies by policy-aware coordination-free transducers.

Semiconnected Datalog

semicon-Datalog$^\neg$ $\subsetneq$ semicon-wILOG$^\neg$ = $\mathcal{M}_{disjoint}$

Connected Rules
$O(x, y, z) \leftarrow E(x, y), E(y, z), E(z, x)$ is connected
$O(x, y, z) \leftarrow E(x, y), E(z, z)$ is not connected

Definition
A stratified-Datalog program is semi-connected if all rules are connected except (possibly) those of the last stratum.

Example

$$TC(x, y) \leftarrow E(x, y)$$
$$TC(x, y) \leftarrow E(x, z), TC(z, y)$$
$$O(x, y) \leftarrow \neg TC(x, y), x \neq y$$

Conclusion and Future Work

Conclusion
- Coordination-free evaluation = (refined) monotonicity
- Introduction of (semi-)connected Datalog

Can we put the CALM-conjecture to rest?

Future Work
- Other settings / other distribution policies?
- Coordination-free + efficient evaluation?

[1] <firstname>.<lastname>@uhasselt.be [2] <firstname>.<lastname>@logicblox.com