Foundations of Declarative Data Analysis Using Limit Datalog Programs

Mark Kaminski, Bernardo Cuenca Grau,
Egor V. Kostylev, Boris Motik, and Ian Horrocks

Department of Computer Science, University of Oxford

DeLBP 2017

Data Analytics

- identifying patterns or trends in raw data:
  market predictions, spot production bottlenecks, …
- gaining importance in research and business
- major challenge: heterogeneous data
  - collected from different sources
  - no uniform data format

State of the Art

- custom-made imperative data processing code

State of the Art

- custom-made imperative data processing code
- labour-intensive
- requires deep technical understanding
- error-prone

Declarative Analytics

Alvaro et al. 2010, Markl 2014, Seo et al. 2015, Shkapsky et al. 2016

- describe what to compute rather than how
- delegate low-level details to the query engine
- improve speed and cost of code development

Declarative Analytics

Alvaro et al. 2010, Markl 2014, Seo et al. 2015, Shkapsky et al. 2016

- describe what to compute rather than how
- delegate low-level details to the query engine
- improve speed and cost of code development
- query language: recursive rules + arithmetic

Loo et al. 2009, Alvaro et al. 2010, Eisner & Filardo 2011, Chin et al. 2015, Seo et al. 2015, Wang et al. 2015, Shkapsky et al. 2016

Example

cost of the cheapest route from London to Melbourne

$\text{flight}(x,y,c) \rightarrow \text{route}(x,y,c)$
$\text{route}(x,z,c_1) \wedge \text{flight}(z,y,c_2) \rightarrow \text{route}(x,y,c_1+c_2)$
$m = \min\{\, c \mid \text{route}(x,y,c) \,\} \rightarrow \text{cheapest\_route}(x,y,m)$

Example

cost of the cheapest route from London to Melbourne

$\text{flight}(x,y,c) \rightarrow \text{route}(x,y,c)$
$\text{route}(x,z,c_1) \wedge \text{flight}(z,y,c_2) \rightarrow \text{route}(x,y,c_1{+}c_2)$
$m = \min\{\, c \mid \text{route}(x,y,c) \,\} \rightarrow \text{cheapest\_route}(x,y,m)$

$\text{cheapest\_route}(\text{London, Melbourne}, x)?$

Challenges

- datalog + arithmetic undecidable see Dantsin et al. 2011
- no universally agreed-on semantics for aggregation
- proposals in the literature suffer from
  - high complexity / undecidability Van Gelder 1993, Ross & Sagiv 1997, Greco 1999, Mazuran et al. 2013
  - limited expressivity Mumick et al. 1990, Consens & Mendelzon 1993, Greco 1999, Faber et al. 2011
  - unnatural syntactic restrictions Ross & Sagiv 1997

Our Goal

unifying formal foundation for declarative analytics

- generalise existing approaches
- natural syntax and semantics
- sufficient expressive power
- theoretically understood computational properties
- amenable to efficient implementation

Overview

- $\text{datalog}_{\mathbb{Z}}$
- decidability
- tractability

Datalog$_{\mathbb{Z}}$

- positive datalog extended with integer arithmetic
- example rule

  $A(x) \wedge B(x,y,m) \wedge C(y,z,n) \wedge (m+1 \leq 2 \cdot n) \rightarrow D(y,z,m+n)$

Datalog$_{\mathbb{Z}}$

- positive datalog extended with integer arithmetic

- example rule

$A(x) \wedge B(x,y,m) \wedge C(y,z,n) \wedge (m+1 \leq 2 \cdot n) \rightarrow D(y,z,m+n)$

ordinary
datalog
atoms

Datalog$_{\mathbb{Z}}$

- positive datalog extended with integer arithmetic

- example rule

$$A(x) \wedge B(x,y,m) \wedge C(y,z,n) \wedge (m+1 \leq 2 \cdot n) \rightarrow D(y,z,m+n)$$

numeric
atoms

Datalog$_{\mathbb{Z}}$

- positive datalog extended with integer arithmetic

- example rule

  $A(x) \wedge B(x,y,m) \wedge C(y,z,n) \wedge (m+1 \leq 2\cdot n) \rightarrow D(y,z,m+n)$

one numeric argument per atom

Datalog$_{\mathbb{Z}}$

- positive datalog extended with integer arithmetic

- example rule

  A(x) ∧ B(x,y,m) ∧ C(y,z,n) ∧ (m+1 ≤ 2·n) → D(y,z,m+n)

comparison
atoms

Datalog$_{\mathbb{Z}}$

- positive datalog extended with integer arithmetic

- example rule

  $A(x) \wedge B(x,y,m) \wedge C(y,z,n) \wedge (m+1 \leq 2\cdot n) \rightarrow D(y,z,m+n)$

- $P \models A(\mathbf{a})$ if $\forall I$: $I \models P$ implies $I \models A(\mathbf{a})$

Datalog$_{\mathbb{Z}}$

- positive datalog extended with integer arithmetic

- example rule

  $A(x) \wedge B(x,y,m) \wedge C(y,z,n) \wedge (m+1 \leq 2 \cdot n) \rightarrow D(y,z,m+n)$

- $P \models A(\mathbf{a})$ if $\forall I\colon I \models P$ implies $I \models A(\mathbf{a})$

two-sorted
FO interpretation
with integers

Datalog$_{\mathbb{Z}}$

- positive datalog extended with integer arithmetic

- example rule

  $\mathrm{A}(x) \wedge \mathrm{B}(x,y,m) \wedge \mathrm{C}(y,z,n) \wedge (m+1 \leq 2 \cdot n) \rightarrow \mathrm{D}(y,z,m+n)$

- $P \models \mathrm{A}(\mathbf{a})$ if $\forall I$: $I \models P$ implies $I \models \mathrm{A}(\mathbf{a})$

- $P \models \mathrm{A}(\mathbf{a})$ iff $\mathrm{A}(\mathbf{a}) \in \mathrm{T}_P^{\infty}(\varnothing)$ $\quad$ T_P immed. cons. operator

Datalog$_{\mathbb{Z}}$

- positive datalog extended with integer arithmetic
- example rule

  $A(x) \wedge B(x,y,m) \wedge C(y,z,n) \wedge (m+1 \leq 2 \cdot n) \rightarrow D(y,z,m+n)$

- $P \models A(\mathbf{a})$ if $\forall I$: $I \models P$ implies $I \models A(\mathbf{a})$
- $P \models A(\mathbf{a})$ iff $A(\mathbf{a}) \in T_P^{\infty}(\varnothing)$ $\quad$ T_P immed. cons. operator
- undecidable even when + is the only operator

Limit Predicates

- keep only the minimal/maximal numeric value
- restrict interpretations to satisfy

  $A(\mathbf{x},m) \wedge (m \leq n) \rightarrow A(\mathbf{x},n)$ for A a min predicate

  $B(\mathbf{x},m) \wedge (n \leq m) \rightarrow B(\mathbf{x},n)$ for B a max predicate

Limit Predicates

- keep only the minimal/maximal numeric value
- restrict interpretations to satisfy

  $\mathrm{A}(\mathbf{x},m) \wedge (m \leq n) \rightarrow \mathrm{A}(\mathbf{x},n)$ for A a min predicate

  $\mathrm{B}(\mathbf{x},m) \wedge (n \leq m) \rightarrow \mathrm{B}(\mathbf{x},n)$ for B a max predicate

- limit datalog$_{\mathbb{Z}}$: all numeric predicates in rule heads limit predicates

Example

cheapest route from London to Melbourne?

flight(x,y,c) → route(x,y,c)
route(x,z,c_1) ∧ flight(z,y,c_2) → route(x,y,c_1+c_2)

route a min predicate

flight		
London	Dubai	500
Dubai	Melbourne	500
London	Melbourne	1500

Example

cheapest route from London to Melbourne?

flight(x,y,c) → route(x,y,c)
route(x,z,c_1) ∧ flight(z,y,c_2) → route(x,y,c_1+c_2)

route a min predicate

flight		
London	Dubai	500
Dubai	Melbourne	500
London	Melbourne	1500

route		
London	Melbourne	1000
London	Melbourne	1500
...	...	...

Example

cheapest route from London to Melbourne?

$\text{flight}(x,y,c) \rightarrow \text{route}(x,y,c)$
$\text{route}(x,z,c_1) \wedge \text{flight}(z,y,c_2) \rightarrow \text{route}(x,y,c_1+c_2)$

route a min predicate

flight		
London	Dubai	500
Dubai	Melbourne	500
London	Melbourne	1500

route		
London	Melbourne	1000

Example

cheapest route from London to Melbourne?

$\mathrm{flight}(x,y,c) \rightarrow \mathrm{route}(x,y,c)$

$\mathrm{route}(x,z,c_1) \wedge \mathrm{flight}(z,y,c_2) \rightarrow \mathrm{route}(x,y,c_1+c_2)$

$\cancel{m = \min\{\, c \mid \mathrm{route}(x,y,c) \,\} \rightarrow \mathrm{cheapest\_route}(x,y,m)}$

flight		
London	Dubai	500
Dubai	Melbourne	500
London	Melbourne	1500

route		
London	Melbourne	1000

Pseudo-Interpretations

- Herbrand interpretations J
- for each min/max predicate A and constants **a** store only the minimal/maximal $k \in \mathbb{Z}$ s.t. $J \models A(\mathbf{a}, k)$

Pseudo-Interpretations

- Herbrand interpretations J
- for each min/max predicate A and constants **a** store only the minimal/maximal $k \in \mathbb{Z}$ s.t. $J \models A(\mathbf{a}, k)$
- each limit datalog$_{\mathbb{Z}}$ program P has a pseudo-model J with $|J| \leq |P|$

Limit Linearity

- limit datalog$_{\mathbb{Z}}$ undecidable: consider *P* as follows

$$\rightarrow A(0)$$

$$A(x_1) \wedge \ldots \wedge A(x_n) \wedge p(x_1,\ldots,x_n)=0 \rightarrow B$$

$P \models B$ iff $p(x_1,\ldots,x_n)=0$ has non-negative integer solution

Limit Linearity

- limit datalog$_{\mathbb{Z}}$ undecidable: consider P as follows

$$\rightarrow A(0)$$

$$A(x_1) \wedge \ldots \wedge A(x_n) \wedge p(x_1,\ldots,x_n)=0 \rightarrow B$$

$P \models B$ iff $p(x_1,\ldots,x_n)=0$ has non-negative integer solution

- limit linearity:
  disallow multiplication between limit variables

Limit Linearity

- limit datalog$_{\mathbb{Z}}$ undecidable: consider P as follows

$$\rightarrow A(0)$$

$$A(x_1) \wedge \ldots \wedge A(x_n) \wedge p(x_1,\ldots,x_n)=0 \rightarrow B$$

$P \models B$ iff $p(x_1,\ldots,x_n)=0$ has non-negative integer solution

- limit linearity:
disallow multiplication between limit variables

$A(x) \wedge B(y) \rightarrow C(x \cdot y)$ not limit linear

Limit Linearity

- limit datalog$_{\mathbb{Z}}$ undecidable: consider P as follows

$$\rightarrow A(0)$$

$$A(x_1) \wedge \ldots \wedge A(x_n) \wedge p(x_1,\ldots,x_n)=0 \rightarrow B$$

$P \models$ B iff $p(x_1,\ldots,x_n)=0$ has non-negative integer solution

- limit linearity:
  disallow multiplication between limit variables

$A(x) \wedge B(y) \rightarrow C(x \cdot y)$ limit linear

Limit-Linear Datalog$_{\mathbb{Z}}$

- fact entailment coNEXPTIME-complete and coNP-complete in data complexity
- upper bounds (data complexity)
  - fact entailment reducible to Presburger validity

    $A(x) \rightarrow B(x+1) \ \leadsto\ \forall x.\mathit{def}_A \wedge (x \leq \mathit{val}_A) \rightarrow \mathit{def}_B \wedge (x+1 \leq \mathit{val}_B)$

  - magnitude of integers in countermodels exponentially bounded — using Chistikov & Haase 2016
  - NP guess-and-check procedure for non-entailment

Limit-Linear Datalog$_{\mathbb{Z}}$

- lower bounds: reduction from square tiling

Limit-Linear Datalog$_{\mathbb{Z}}$

- lower bounds: reduction from square tiling

Square Tiling

input: finite set T of tiles
horizontal compatibility relation $H \subseteq T \times T$
vertical compatibility relation $V \subseteq T \times T$
number N

problem: is there a function $N \times N \rightarrow T$
satisfying H and V (tiling)?

Limit-Linear Datalog$_{\mathbb{Z}}$

- lower bounds: reduction from square tiling
  - interpret each $N^2 \cdot \lceil \log_2|T| \rceil$-bit number n as a candidate tiling; initialise n with 0

Limit-Linear Datalog$_{\mathbb{Z}}$

- lower bounds: reduction from square tiling
  - interpret each $N^2 \cdot \lceil \log_2 |T| \rceil$-bit number *n* as a candidate tiling; initialise *n* with 0
  - if *n* not a tiling, increase *n*

Limit-Linear Datalog$_{\mathbb{Z}}$

- lower bounds: reduction from square tiling
  - interpret each $N^2 \cdot \lceil \log_2 |T| \rceil$-bit number *n* as a candidate tiling; initialise *n* with 0
  - if *n* not a tiling, increase *n*
  - if $n > 2^{N^2 \cdot \lceil \log_2 |T| \rceil} - 1$, return ‘noSolution’

Limit-Linear Datalog$_{\mathbb{Z}}$

- lower bounds: reduction from square tiling
  - interpret each $N^2 \cdot \lceil \log_2 |T| \rceil$-bit number n as a candidate tiling; initialise n with 0
  - if n not a tiling, increase n
  - if $n > 2^{N^2 \cdot \lceil \log_2 |T| \rceil} - 1$, return ‘noSolution’
  - $P \models$ noSolution iff no tiling exists

Tractability

(in data complexity)

- stability: rules “strictly monotone”

Tractability

(in data complexity)

- stability: rules “strictly monotone”

- example

  $A(m) \wedge (m \leq 10) \rightarrow B(m)$

Tractability

(in data complexity)

- stability: rules "strictly monotone"

- example
  $A(m) \wedge (m \leq 10) \rightarrow B(m)$ not stable

A
10
15

Tractability

(in data complexity)

- stability: rules "strictly monotone"
- example

  $A(m) \wedge (m \leq 10) \rightarrow B(m)$ not stable

  $A(m) \rightarrow B(m)$ stable

A
10
15

Tractability

(in data complexity)

- stability: rules "strictly monotone"

- example

  A(m) ∧ (m ≤ 10) → B(m) not stable

  A(m) → B(m) stable

A
10
15

- fact entailment for stable limit-linear datalog$_{\mathbb{Z}}$
  EXPTIME-complete and PTIME-complete w.r.t. data

Tractability

(in data complexity)

- stability: rules "strictly monotone"

- example

  $A(m) \wedge (m \leq 10) \rightarrow B(m)$ not stable

  $A(m) \rightarrow B(m)$ stable

A
10
15

- fact entailment for stable limit-linear datalog$_{\mathbb{Z}}$
  EXPTIME-complete and PTIME-complete w.r.t. data

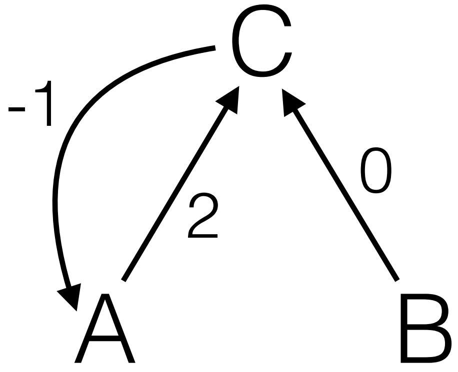
  - lower bounds: datalog

Tractability: Upper Bounds

- P and J induce a value propagation graph $G_{P,J}$

J = { A(0), B(2), C(0) }   A,B,C max

$A(x) \wedge B(y) \rightarrow C(x+y)$
$C(x) \rightarrow A(x\text{-}1)$
$B(x) \wedge (x>5) \rightarrow B(x+1)$

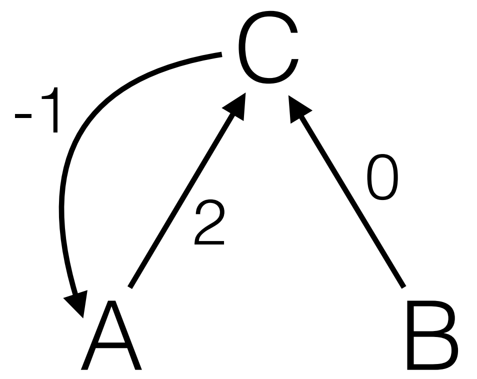


Tractability: Upper Bounds

- P and J induce a value propagation graph $G_{P,J}$

J = { A(0), B(2), C(0) }   A,B,C max

$\boxed{\text{A}}(x) \wedge \text{B}(y) \rightarrow \boxed{\text{C}}(x+y)$
$\text{C}(x) \rightarrow \text{A}(x\text{-}1)$
$\text{B}(x) \wedge (x>5) \rightarrow \text{B}(x+1)$

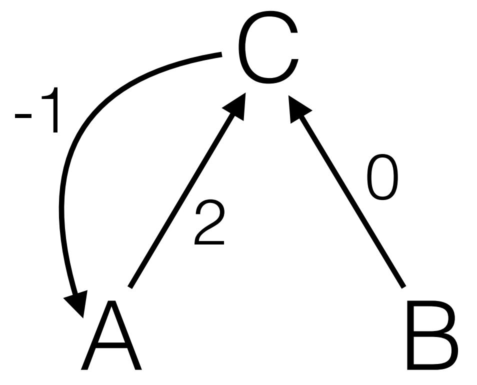


Tractability: Upper Bounds

- *P* and *J* induce a value propagation graph $G_{P,J}$

J = { A(0), B(2), C(0) }   A,B,C max

$\boxed{\mathrm{A}}(0) \wedge \mathrm{B}(2) \rightarrow \boxed{\mathrm{C}}(0+2)$
$\mathrm{C}(x) \rightarrow \mathrm{A}(x\text{-}1)$
$\mathrm{B}(x) \wedge (x>5) \rightarrow \mathrm{B}(x+1)$



Tractability: Upper Bounds

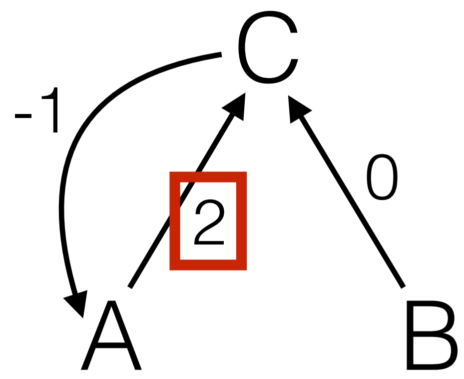
- P and J induce a value propagation graph $G_{P,J}$

J = { A(0), B(2), C(0) }   A,B,C max

$\boxed{\text{A}}(0) \wedge \text{B}(2) \rightarrow \boxed{\text{C}}(0+2)$ 2-0 = 2

$\text{C}(x) \rightarrow \text{A}(x\text{-}1)$

$\text{B}(x) \wedge (x>5) \rightarrow \text{B}(x+1)$



Tractability: Upper Bounds

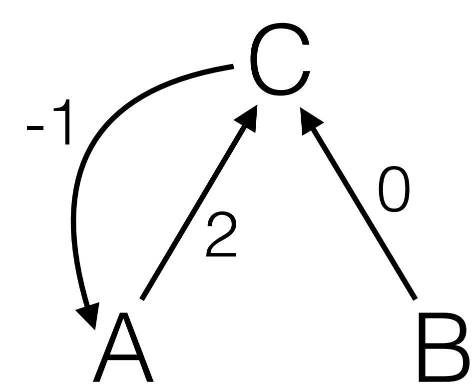
- P and J induce a value propagation graph $G_{P,J}$

J = { A(0), B(2), C(0) }   A,B,C max

$A(0) \wedge \boxed{B}(2) \rightarrow \boxed{C}(0+2)$ 2-0 = 2

$C(x) \rightarrow A(x\text{-}1)$

$B(x) \wedge (x>5) \rightarrow B(x+1)$



Tractability: Upper Bounds

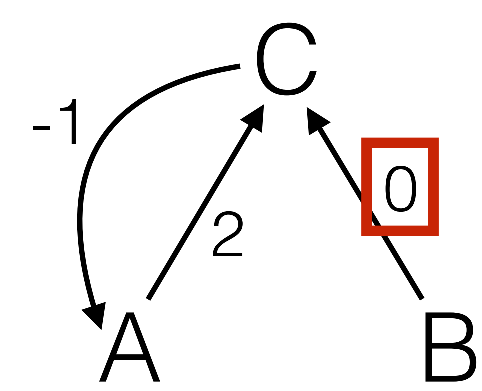
- P and J induce a value propagation graph $G_{P,J}$

$J = \{\ A(0), B(2), C(0)\ \}$ A,B,C max

$A(0) \wedge \boxed{B}(2) \rightarrow \boxed{C}(0+2)$ 2-2 = 0

$C(x) \rightarrow A(x\text{-}1)$

$B(x) \wedge (x>5) \rightarrow B(x+1)$

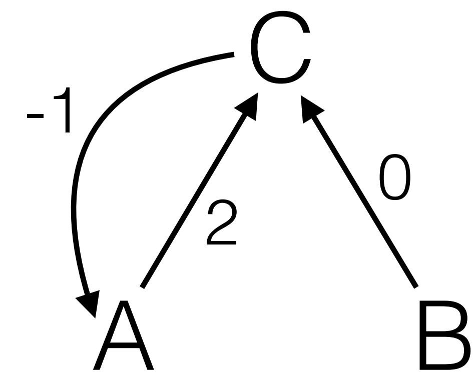


Tractability: Upper Bounds

- P and J induce a value propagation graph $G_{P,J}$

$J = \{$ A(0), B(2), C(0) $\}$ A,B,C max

$A(x) \wedge B(y) \to C(x+y)$
$\boxed{C}(x) \to \boxed{A}(x\text{-}1)$
$B(x) \wedge (x>5) \to B(x+1)$

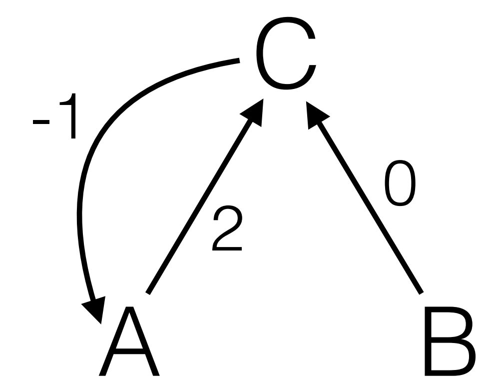


Tractability: Upper Bounds

- P and J induce a value propagation graph $G_{P,J}$

J = { A(0), B(2), C(0) }   A,B,C max

$A(x) \wedge B(y) \rightarrow C(x+y)$
$\boxed{C}(0) \rightarrow \boxed{A}(0-1)$
$B(x) \wedge (x>5) \rightarrow B(x+1)$



Tractability: Upper Bounds

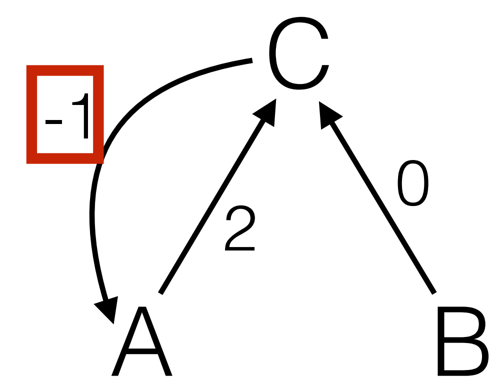
- P and J induce a value propagation graph $G_{P,J}$

$J = \{\ A(0), B(2), C(0)\ \}$ A,B,C max

$A(x) \wedge B(y) \rightarrow C(x+y)$

$\boxed{C}(0) \rightarrow \boxed{A}(0\text{-}1)$ -1-0 = -1

$B(x) \wedge (x>5) \rightarrow B(x+1)$

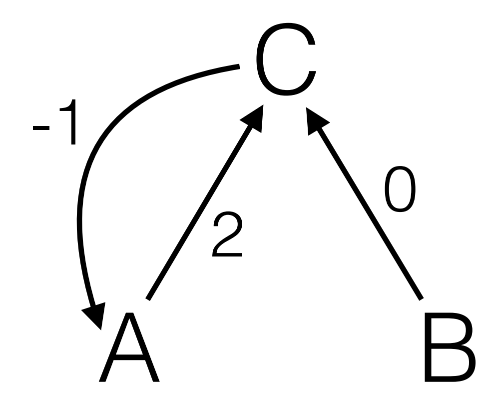


Tractability: Upper Bounds

- P and J induce a value propagation graph $G_{P,J}$

J = { A(0), B(2), C(0) }   A,B,C max

A(x) ∧ B(y) → C(x+y)
C(x) → A(x-1)
B(x) ∧ (x>5) → B(x+1)

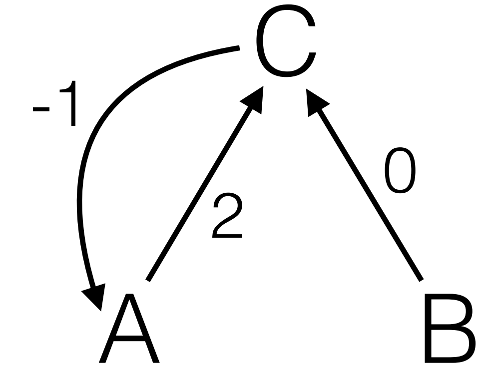


Tractability: Upper Bounds

- *P* and *J* induce a value propagation graph $G_{P,J}$

  $J = \{ A(0), B(2), C(0) \}$ A,B,C max

  $A(x) \wedge B(y) \to C(x+y)$
  $C(x) \to A(x-1)$
  $B(x) \wedge (x>5) \to B(x+1)$



- all atoms on a positive-weight cycle of $G_{P,J}$ 'diverge' in $T_P^\infty(J)$ if *P* stable

Upper Bounds ctd.

- algorithm for computing $\mathrm{T}_P^\infty(\emptyset)$:
  starting with $J := \emptyset$ iterate
  - for each atom on a positive-weight cycle in $G_{P,J}$, set numeric argument in J to '∞'
  - $J := \mathrm{T}_P(J)$

Upper Bounds ctd.

- algorithm for computing $\mathrm{T}_P^\infty(\emptyset)$:
  starting with $J := \emptyset$ iterate
  - for each atom on a positive-weight cycle in $G_{P,J}$, set numeric argument in J to '∞'
  - $J := \mathrm{T}_P(J)$
- computation converges in polynomial time w.r.t. maximal size of $G_{P,J}$
  - polynomial in data complexity
  - exponential in combined complexity

Stable Datalog$_{\mathbb{Z}}$

- captures useful analytic tasks
- same complexity as for datalog:
  EXPTIME-complete and PTIME-complete w.r.t. data

Stable Datalog$_{\mathbb{Z}}$

- captures useful analytic tasks
- same complexity as for datalog:
  EXPTIME-complete and PTIME-complete w.r.t. data
- semantic stability undecidable
- syntactic sufficient condition: type consistency
  - checkable in LOGSPACE

Future Work

- non-monotonic extension (work in progress)
- aggregation operators
- multiplication between limit variables, division, reals
- connections to existing approaches
- scalable implementation
- applications

Thank you!

References

Alvaro et al. 2010 Alvaro, P.; Condie, T.; Conway, N.; Elmeleegy, K.; Hellerstein, J. M.; Sears, R.: BOOM analytics: exploring data-centric, declarative programming for the cloud. EuroSys 2010

Chin et al. 2015 Chin, B.; von Dincklage, D.; Ercegovac, V.; Hawkins, P.; Miller, M. S.; Och, F. J.; Olston, C.; Pereira, F: Yedalog: Exploring knowledge at scale. SNAPL 2015

Chistikov & Haase 2016 Chistikov, D.; Haase, C.: The taming of the semi-linear set. ICALP 2016

Consens & Mendelzon 1993 Consens, M. P.; Mendelzon, A. O.: Low complexity aggregation in GraphLog and Datalog. Theor. Comput. Sci. 116, 1993

Dantsin et al. 2001 Dantsin, E.; Eiter, T.; Gottlob, G.; Voronkov, A.: Complexity and expressive power of logic programming. ACM Comput. Surv. 33, 2001

Eisner & Filardo 2011 Eisner, J.; Filardo, N. W.: Dyna: Extending datalog for modern AI. Datalog 2011

Faber et al. 2011 Wolfgang Faber, Gerald Pfeifer, Nicola Leone: Semantics and complexity of recursive aggregates in answer set programming. Artif. Intell. 175, 2011

Greco 1999 Greco, S.: Dynamic Programming in Datalog with Aggregates. IEEE TKDE 11, 1999

Loo et al. 2009 Loo, B. T.; Connie, T.; Garofalakis, M. N.; Gay, D. E.; Hellerstein, J. M.; Maniatis, P.; Ramakrishnan, R.; Roscoe, T.; Stoica, I.: Declarative networking. Commun. ACM 52, 2009

References

Markl 2014 Markl, V.: Breaking the chains: On declarative data analysis and data independence in the big data era. PVLDB 7, 2014

Mazuran et al. 2013 Mazuran, M.; Serra, E.; Zaniolo, C.: Extending the power of datalog recursion. VLDB J. 22, 2013

Mumick et al. 1990 Mumick, I. S.; Pirahesh, H.; Ramakrishnan R.: The Magic of Duplicates and Aggregates. VLDB 1990

Ross & Sagiv 1997 Ross, K. A.; Sagiv, Y.: Monotonic Aggregation in Deductive Database. J. Comput. Syst. Sci. 54, 1997

Seo et al. 2015 Seo, J.; Guo, S.; Lam, M. S.: SociaLite: An efficient graph query language based on datalog. IEEE TKDE 27, 2015

Shkapsky et al. 2016 Shkapsky, A.; Yang, M.; Interlandi, M.; Chiu, H.; Condie, T.; Zaniolo, C.: Big data analytics with datalog queries on Spark. SIGMOD 2016

Van Gelder 1993 Van Gelder, A.: Foundations of Aggregation in Deductive Databases. DOOD 1993

Wang et al. 2015 Wang, J.; Balazinska, M.; Halperin, D.: Asynchronous and fault-tolerant recursive datalog evaluation in shared-nothing engines. PVLDB 8, 2015