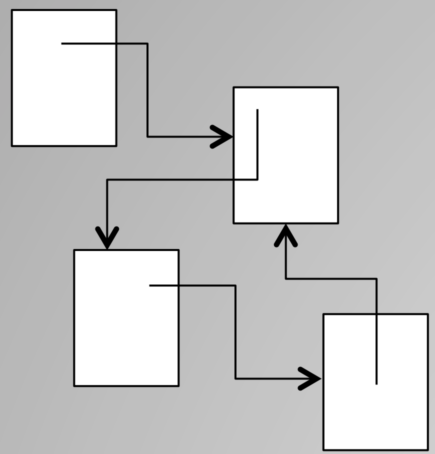
Viewing the Web as a Distributed Knowledge Base

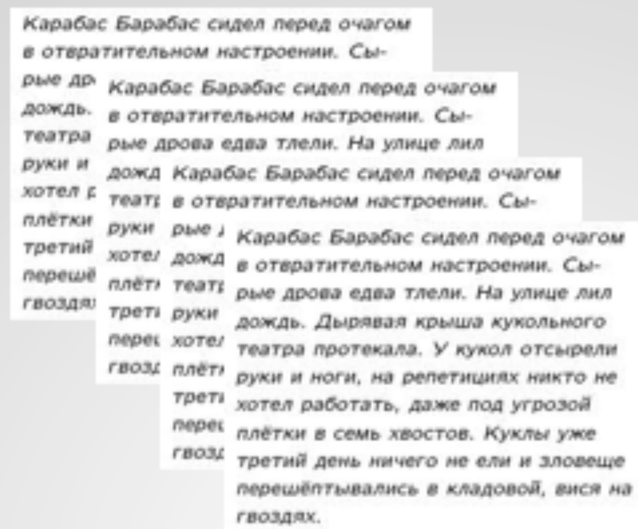
Serge Abiteboul
INRIA Saclay, Collège de France and ENS Cachan

- The Web as a distributed knowledge base
- WebdamLog: a rule-based language for the Web
- The WebdamLog system
- Inconsistencies and uncertainty
- Conclusion

The Web



hypertext



Карабас Барабас сидел перед очагом
в отвратительном настроении. Сы-
рые дрова едва тлели. На улице лил
дождь. Дырявая крыша кукольного
театра протекала. У кукол отсырели
руки и ноги, на репетициях никто не
хотел работать, даже под угрозой
плётки в семь хвостов. Куклы уже
третий день ничего не ели и зловеще
перешёптывались в кладовой, вися на
гвоздях.

universal library of text

and multimedia

personal/private data

social data

A typical Web user's data

- What kinds of data? *all kinds*

  - *data*: photos, music, movies, reports, email
  - *metadata*: photo taken by Alice in Paris on ...
  - *ontologies*: Alice's ontology and mapping with other ontologies
  - *localization*: Alice's pictures are on Picasa, back-ups are at INRIA
  - *security*: Facebook credentials (Alice, 123456)
  - *annotations*: Alice likes Elvis' website
  - *beliefs*: Alice believes Elvis is alive
  - *external knowledge*: Bob keeps copies of Alice's pictures
  - *time*, *provenance*, ...

Social data

A typical Web user's data

- What kinds of data? *all kinds*
- Where is the data? *everywhere*
  - laptop, desktop, smartphone, tablet, car computer
  - mail, address book, agenda
  - Facebook, LinkedIn, Picasa, YouTube, Tweeter
  - svn, Google docs
  - also access to data / information of family, friends, companies associations

A typical Web user's data

- What kinds of data? *all kinds*
- Where is the data? *everywhere*
- What kind of organization? *heterogeneous*
  - terminology: different ontologies
  - systems: personal machines, social networks
  - distribution: different localization
  - security: different protocols
  - quality: incomplete / inconsistent information

Example of processing

Alice and *Bob* are getting engaged. Their friends want to offer them an album of photos where they are together

To make such a photo album

- Find friends of *Alice* & *Bob* (say with *Facebook*)
- for each friend, find where she keeps her photos (say, *Picassa*)
  - find the means to access her photos possibly via friends
  - find the photos that feature *Bob* and *Alice* together, e.g., using tags or face recognition software
- possibly ask someone to verify the results

Some reasoning is needed to execute these tasks (automatically)!

A typical Web user

- Overwhelmed by the mass of information
- Cannot find the information needed
- Is not aware of important events
- Cannot manage/control how others access and use his/her own data

How can systems help?

- We need to move from a Web of text to a Web of knowledge
  - In the spirit of semantic Web
- To better support user needs,
  - Systems need to analyze what is happening and construct knowledge
  - Systems should exchange knowledge
  - Systems should reason and infer knowledge

YOU need help!

Thesis

All this forms a distributed knowledge base

with processing based on automated reasoning

Issues

- Distributed reasoning
- Exchanging facts and rules

WebdamLog

- Contradictions
- Missing and noisy data

Ignore for now

- The Web as a distributed knowledge base
- WebdamLog: a rule-based language for the Web
- The WebdamLog system
- Inconsistencies and uncertainty
- Conclusion

WebdamLog: a datalog-style language

Why datalog? A prehistoric language by Web time...

- \+ nice and compact syntax
- \+ well-studied with many extensions
- \+ recursion essential in a distributed setting: cycles in the network

Extensional facts

friend(“peter”,”paul”) friend(“paul”, “mary”) friend(“mary”,”sue”)

Datalog program

fof(x,y) :- friend(x,y)
fof(x,y) :- friend(x,z), fof(z,y)

Intentional facts

fof(“peter”,”paul”) fof(“peter”,”mary”) fof(“peter”, “sue”)
fof(“paul”, “mary”) fof(“paul”, “sur”)
fof(“mary”,”sue”)

WebdamLog

Extends datalog

- negation, updates, distribution, delegation, time

For a world that is

- distributed: autonomous and asynchronous peers
- dynamic: knowledge evolves; peers come and go

Influenced by

- Active XML (INRIA) - for distribution & intentional data
- Dedalus (UC Berkeley) - for time & implementation

Warning

Not as simple

Not as beautiful

More procedural

But this is needed for
real Web applications!

WebdamLog is not datalog

Schema

(π, E, I, σ)

π possibly infinite set of peer IDs

E set of extensional relations of the form m@p

I set of intentional relations of the form m@p

σ sorting function

for each m@p, σ(m@p) is an integer (its sort)

Facts

Facts are of the form $m@p(a_1, ..., a_n)$, where

m is a *relation* name & p is a *peer* name

$a_1, ..., a_n$ are *data* values (n is the arity of $m@p$)

the set of data values includes the relations and peer names

Examples

friend@my-iphone(“peter”, “paul”) extensional

fof@my-iphone(“adam”, “paul”) intentional

Examples of facts

data & metadata: pictures@alice-iphone(1771.jpg, “Paris”, 11/11/2011)

ontology: isA@yago.com("Elvis”, theKing)

annotations: tags@delicious.com(“wikipedia.org”, encyclopedia)

localization: where@alice(pictures, picasa/alice)

access rights: right@picasa(pictures, friends, read)

security: secret@picasa/alice; public@picasa/alice

Rules

A ***term*** is a variable or a constant

Rules are of the form

$R@$P($U) :- (not) $R1@$P1($U1), ..., (not) $Rn@$Pn($Un)

where

$R, $Ri are *relation* terms

$P, $Pi are *peer* terms

$U, $Ui are *tuples* of terms

Safety condition

$R and $P must appear positively bound in the body

each variable in a negative literal must appear positively bound in the body

Examples coming up, stay tuned

Semantics

A state (I, Γ, Γ^*) : each peer p has

extensional facts $I(p)$, defining the local state of p

local rules $\Gamma(p)$, defining the program of p

rules $\Gamma^*(p,q)$ that have been delegated to p by some peer q

State transition

Choose some peer p randomly – asynchronously

Compute the transition of p

- the database updates at p
- the messages sent to other peers
- the delegations of rules to other peers

Keep going forever

$(I_0, \Gamma_0, \varnothing) \rightarrow (I_1, \Gamma_1, \Gamma_1^*) \rightarrow \ldots \rightarrow (I_n, \Gamma_n, \Gamma_n^*) \rightarrow \ldots$

Fair sequence: each peer is selected infinitely often

The semantics of rules

Classification based on locality and nature of head predicates (intentional or extensional)

- Local rule at my-laptop: all predicates in the body of the rules are from my-laptop

Local with local intentional head	classic datalog
Local with local extensional head	database update
Local with non-local extensional head	messaging between peers
Local with non-local intentional head	view delegation
Non-local	general delegation

Local rules with local intentional head

Example: Rule at peer my-laptop

friend is extensional, fof is intentional

fof@my-iphone($x, $y) :- friend@my-iphone($x,$y)

fof@my-iphone($x,$y) :- friend@my-iphone($x,$z), fof@my-iphone($z,$y)

fof is the transitive closure of friend

Datalog = WebdamLog with only local rules and local intentional head

Local rules with local extensional head

A new fact is inserted into the local database

```
believe@my-iphone("Alice", $loc) :-
        tell@my-iphone($p,"Alice", $loc),
        friend@my-iphone($p)
```

Local rules with non-local extensional head

A new fact is sent to an external peer via a message

$message@$peer($name, “Happy birthday!”) :-

today@my-iphone($date),

birthday@my-iphone($name, $message, $peer, $date)

Extensional facts:

today@my-iphone(March 6)

birthday@my-iphone("Manon”, “sendmail”, “gmail.com”, March 6)

sendmail@gmail.com("Manon”, “Happy birthday”)

Local rules with non-local intentional head

View delegation!

boyMeetsGirl@gossip-site($girl, $boy) :-

girls@my-iphone($girl, $loc),

boys@my-iphone($boy, $loc)

Semantics of boyMeetGirl@gossip-site is a join of relations girls and boys from my-iphone

Formally, my-iphone delegates a rule boyMeetGirl@gossip-site(g,b) for each g, b, l, girls@my-iphone(g,l), boys@my-iphone(b,l)

Non-local rules: general delegation

(at my-iphone): boyMeetsGirl@gossip-site($girl, $boy) :-

girls@my-iphone($girl, $loc),

boys@alice-iphone($boy, $loc)

Suppose that girls@my-iphone("Alice", "Julia's birthday") holds.

Then my-iphone installs the following rule at alice-iphone

(at alice-iphone): boyMeetsGirl@gossip-site("Alice", $boy) :-

boys@alice-iphone($boy, "Julia's birthday")

When girls@my-iphone("Alice", "Julia's birthday") no longer holds, my-iphone uninstalls the rule

Non-local rules: general delegation

(at my-iphone): boyMeetsGirl@gossip-site($girl, $boy) :-

girls@my-iphone($girl, $loc),

boys@alice-iphone($boy, $loc)

An alternative, more database-ish, way of looking at this:

at my-iphone : seed@alice-iphone($girl, $loc):- *view*

girls@my-iphone($girl, $loc) *delegation*

at alice-iphone : boyMeetsGirl@gossip-site($girl, $boy) :-

seed@alice-iphone($girl, $loc),

boys@alice-iphone($boy, $loc) *delegation*

Complexity of delegation: illustration

fof(x,y) :- friend(x,y)

(at p) fof@p(x,y) :- peers@p($q), friend@$q(x,y)

If peers@p(q_1) holds, this rule installs

(at q_1) fof@p(x,y) :- friend@q_1(x,y)

If peers@p contains 100 000 tuples

peers@p(q_1), ..., peers@p($q_{100\,000}$)

This rule will install 100 000 rules!

for i=1 to 100 000 (at q_i) fof@p(x,y) :- friend@q_i(x,y)

Data complexity transformed into program complexity

Summary of results [PODS 2011]

- Formal definition of the semantics of WebdamLog
- Results on expressivity
  - the model with delegation is more general, unless all peers and programs are known in advance
- Convergence is very hard to achieve
  - positive WebdamLog
  - strongly stratified programs with negation

- The Web as a distributed knowledge base
- WebdamLog: a rule-based language for the Web
- The WebdamLog system
- Inconsistencies and uncertainty
- Conclusion

WebdamLog peers

[demo ICDE 2011, WebDB 2011]

Support communication with other peers

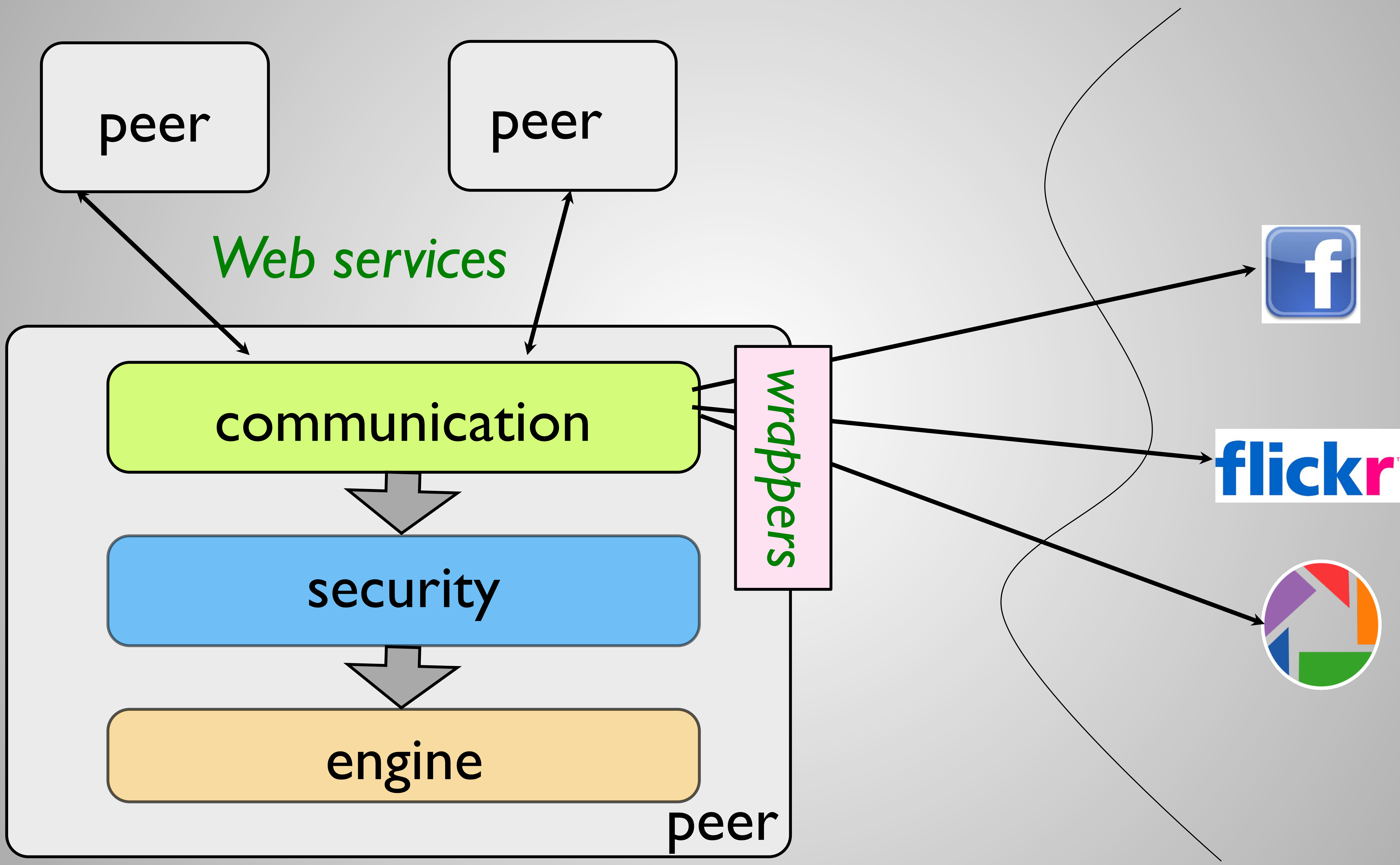
Support common security protocols

Support wrappers to external systems such as Facebook

Manage knowledge

- store knowledge (facts and rules)
- exchange knowledge with other peers
- perform reasoning

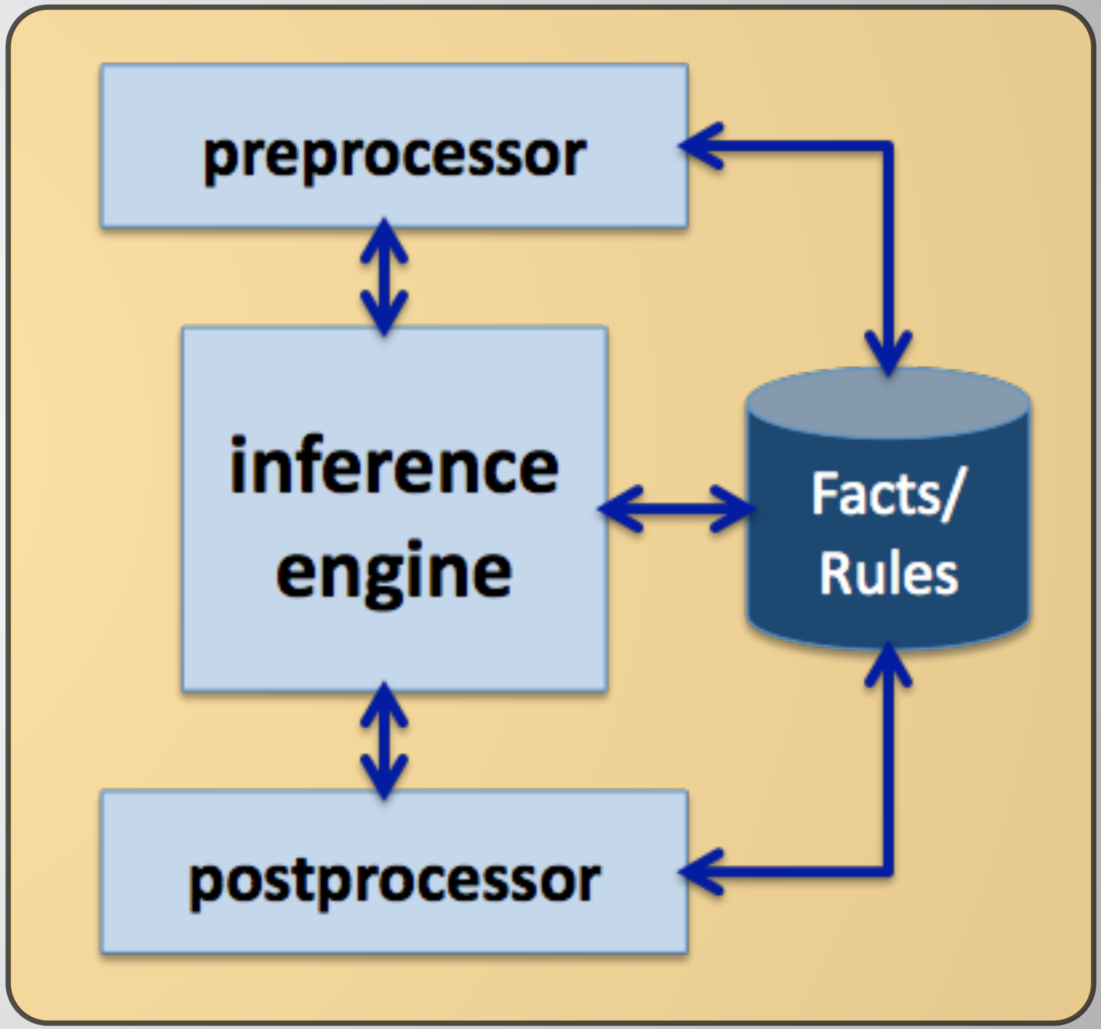
WebdamLog peers



WebdamLog engine [ongoing work]

Based on Bud

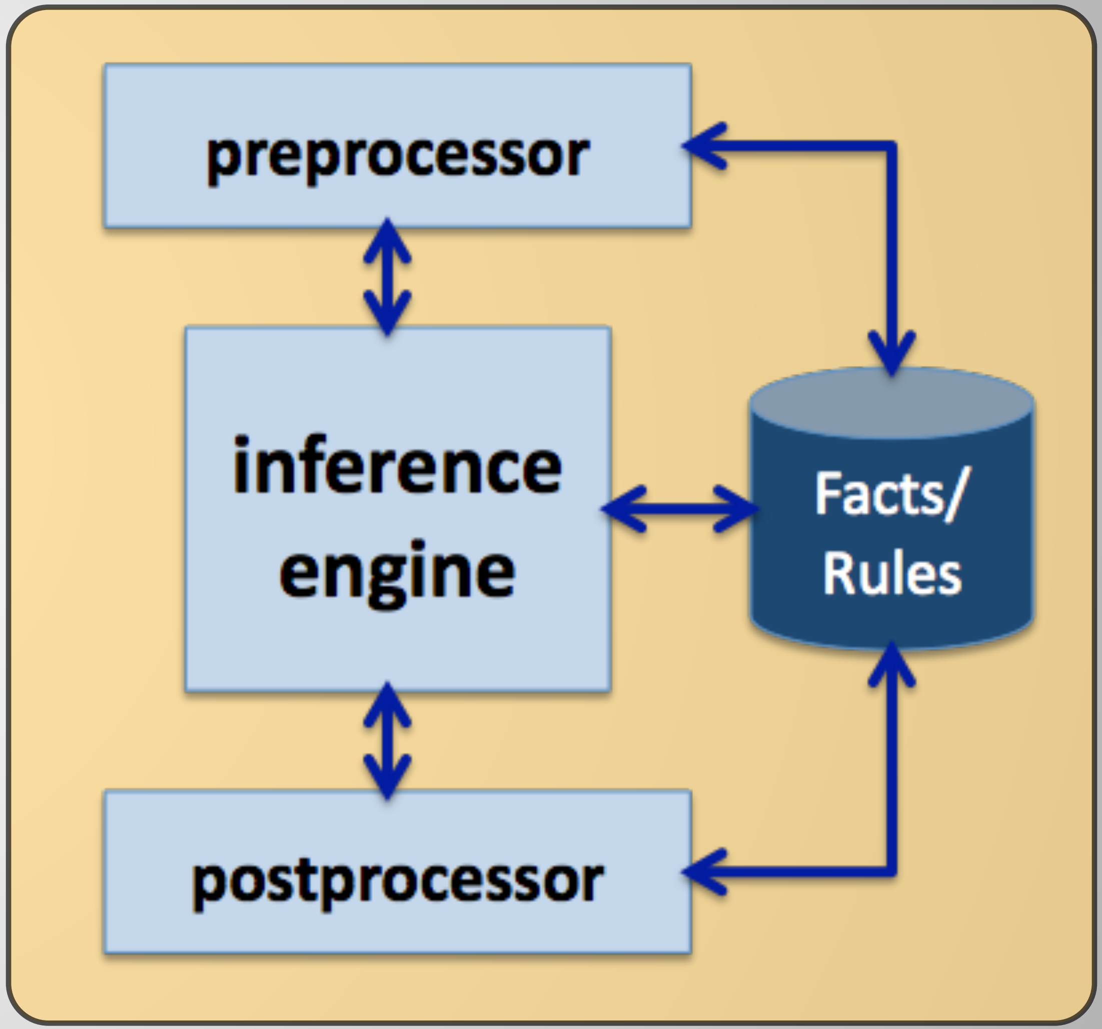
- developed at UC Berkeley, implemented in Ruby, open-source
- supports Bloom - an extension of datalog
- implements communication between peers
- serious experiments



WebdamLog inference: beyond Bud

- Translation of WebdamLog to Bloom (Bud's language)
- Features of WebdamLog not supported in Bud

1. Variable relation and peer names
2. Delegation: non-local rules, non-local relations in the body
3. Adding and removing rules at runtime: needed because of delegation



Example of runtime inference

(rule$_1$ at p) boyMeetsGirl@p($girl, $boy) :-

girls@p($girl, $loc),

boys@p($boy, $loc)

direct knowledge

(rule$_2$ at q) gossip@$peer($girl, $boy) :-

boyMeetsGirl@q($girl, $boy),

allPeers($peer)

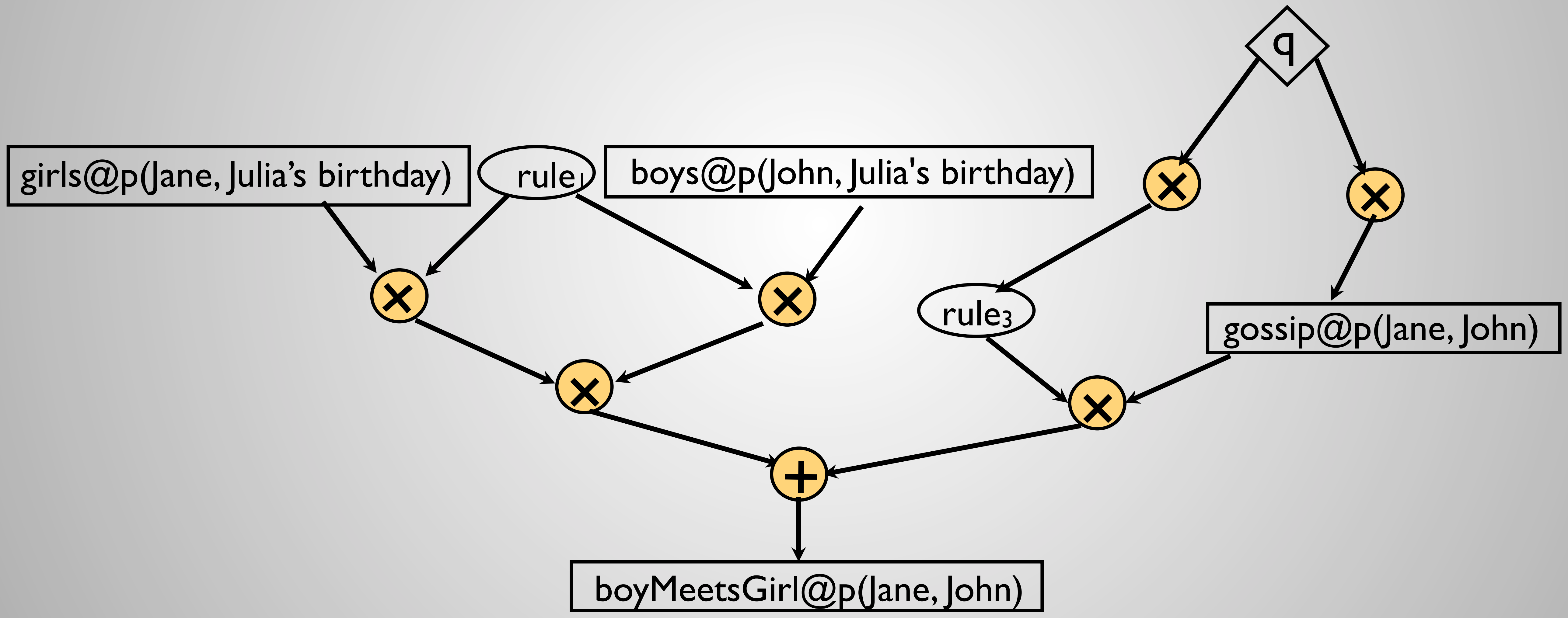
hearsay

(rule$_3$ at q) boyMeetsGirl@p($girl, $boy) :-

gossip@p($girl, $boy)

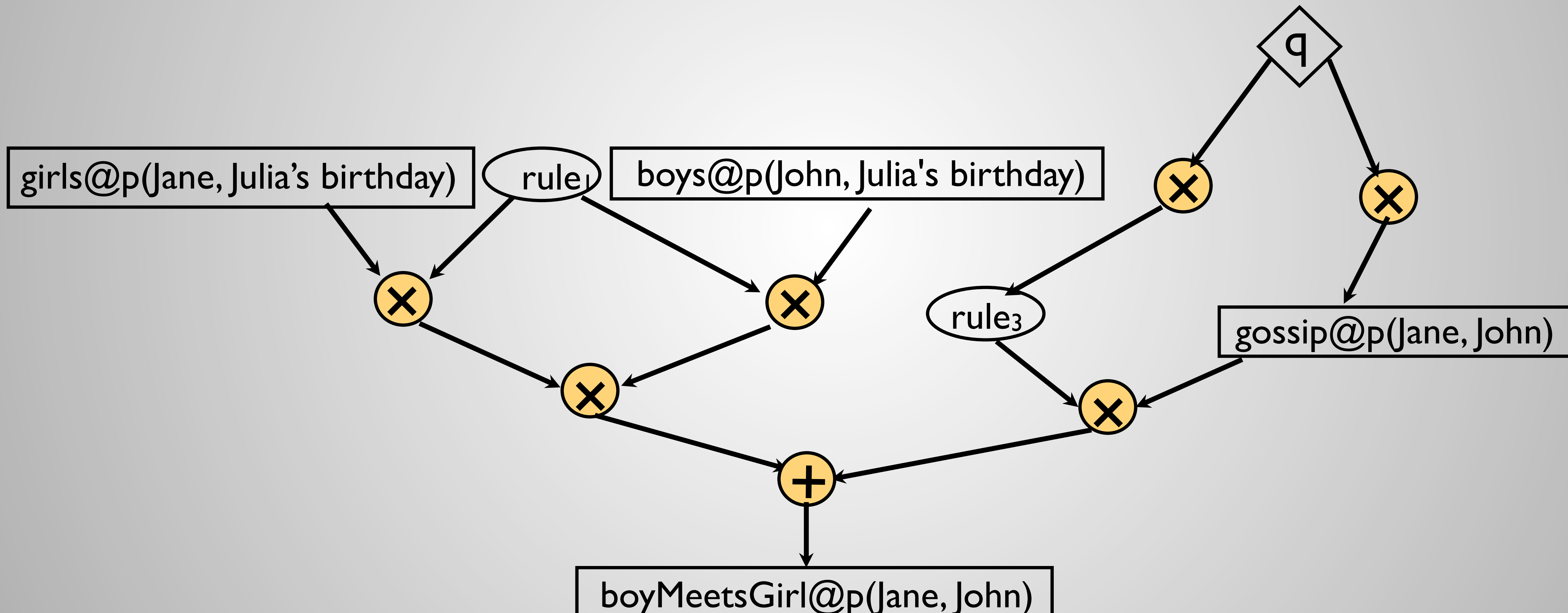
Adding facts at runtime

Maintain a provenance graph for update management



Removing facts at runtime

Avoid recomputation at each update using provenance



Provenance graphs

- Records the history of derivation
- Provenance semiring semantics [Green et al. 07]
  - alternative or joint use of data
  - facts, rules, peers are nodes
- Useful for performance optimization
- Other uses
  - explain results to users
  - specify and verify access rights

- The Web as a distributed knowledge base
- WebdamLog: a rule-based language for the Web
- The WebdamLog system
- Inconsistencies and uncertainty
- Conclusion

Motivation

- Contradictions (in intentional or extensional data) come from
  - errors, lies, rumors, updates
  - FD violations: some think Alice was born in Paris, others that she was born in London
  - opinions: some think Brahms is great; others don't
- Uncertainty comes from
  - lack of information
  - contradictions
- Probabilities may be used to measure uncertainty
  - 80% think Alice was born in Paris, 20% in London
  - sources: we observed that Peter is wrong 20% of the time

Roadmap

We consider

reasoning in an uncertain and inconsistent world

We do this

- first for the centralized setting
- then with distribution
- finally with probabilities

Datalog + FDs

WebdamLog

and sampling

Datalog example

- Where is Alice?
- A relation IsIn(*person*, *city*, *peer*)

  with the FD (*person*, *peer*) → *city*

  peer believes *person* to be in *city*
- Consider a datalog rule

IsIn($per, $city, $p') :- IsIn($per, city, $p), friend($p', $p)

IsIn(Alice, London, Bob) IsIn(Alice, Paris, Sue)

friend(my-iphone, Bob) friend(my-iphone, Sue)

Datalog with nondeterministic fact-at-a-time semantics

Immediate consequence operator: a single fact is derived only if it does not contradict known facts

A possible world is a maximal consequence. Example:

IsIn($per, $city, $p') :- IsIn($per, city, $p), friend($p', $p)

IsIn(Alice, London, Bob)

friend(my-iphone, Bob)

IsIn(Alice, Paris, Sue)

friend(my-iphone, Sue)

Infer: IsIn(Alice, Paris, my-iphone)

In practice set-at-a-time semantics is more efficient

Discussion

Inflationary non-deterministic semantic ("stubborn" choices)

Related to 2-stable models

Proof theory

- Possible facts NP-complete
- Sure facts coNP-complete

Many possible alternative semantics

Distributed setting: use WebdamLog

To simplify, we focus only on local and deductive rules

The semantics is inflationary and non-deterministic

A subtlety: Each peer has to recall the choices made to always make the same choice in the future (when talking to other peers): stubborn

The causes of uncertainty

- Uncertainty in base facts
- Uncertainty in the order of peer activations
- Uncertainty in choosing immediate consequences

Probabilities

Probabilistic interpretation to measure uncertainty

- For base facts, use independent probabilistic events
- Uniform distribution for the next peer to activate
- Uniform distribution in choosing the next immediate consequence
  - Can be done efficiently if there is a single FD & more complicated otherwise

Example: captures voting

Bob's rules

IsIn@$p($x,$y) :- Follower@bob($p), IsIn@bob($x,$y)

IsIn@bob($x,$y) :- baseIsIn@bob($x,$y)

Suppose each peer has similar rules

Claim: For acyclic networks, the probability of a peer inferring a fact is exactly its relative support at his friends

Note: this also give semantics for more complicated cases such as networks with cycles

Query answering

Resulting tuples of a query q have associated probabilities

Exact evaluation using c-tables

- Too costly in practice

Sampling technique

- Each peer makes probabilistic choices along the way
- Converges to the probability of q when the number of samples grows

- The Web as a distributed knowledge base
- WebdamLog: a rule-based language for the Web
- The WebdamLog system
- Inconsistencies and uncertainty
- Conclusion

Thesis

Let us turn the Web into a distributed knowledge base

with billions of users

supported by billions of systems

analyzing information

extracting knowledge

exchanging knowledge

inferring knowledge

Contribution

WebdamLog

- A language for distributed data management [PODS 2011]
- Datalog with distribution, updates, messaging
- Main novelty: delegation

System implementation

- Handles heterogeneity, localization and access control [WebDB 2011]
- WebdamlExchange peer In Java [demo ICDE 2011]
- WebdamLog engine based on Bud

On-going work

The implementation

- More optimization strategies such as Magic Set

Probabilistic WebdamLog

- Query processing
- Explaining results to users: top-k proofs

Collaboration between peers to answer queries

Lots of fun & many open questions

Issues

- Access control based on provenance
- Concurrency control
  - Difficulty: right revocation
- Optimization
  - Links with optimization in Active XML
- Verification of applications
  - Links with business artifacts

Joint work with Emilien Antoine, Meghyn Bienvenu, Daniel Deutch, Alban Galland, Kristian Lyndbaek, Julia Stoyanovich, Jules Testard

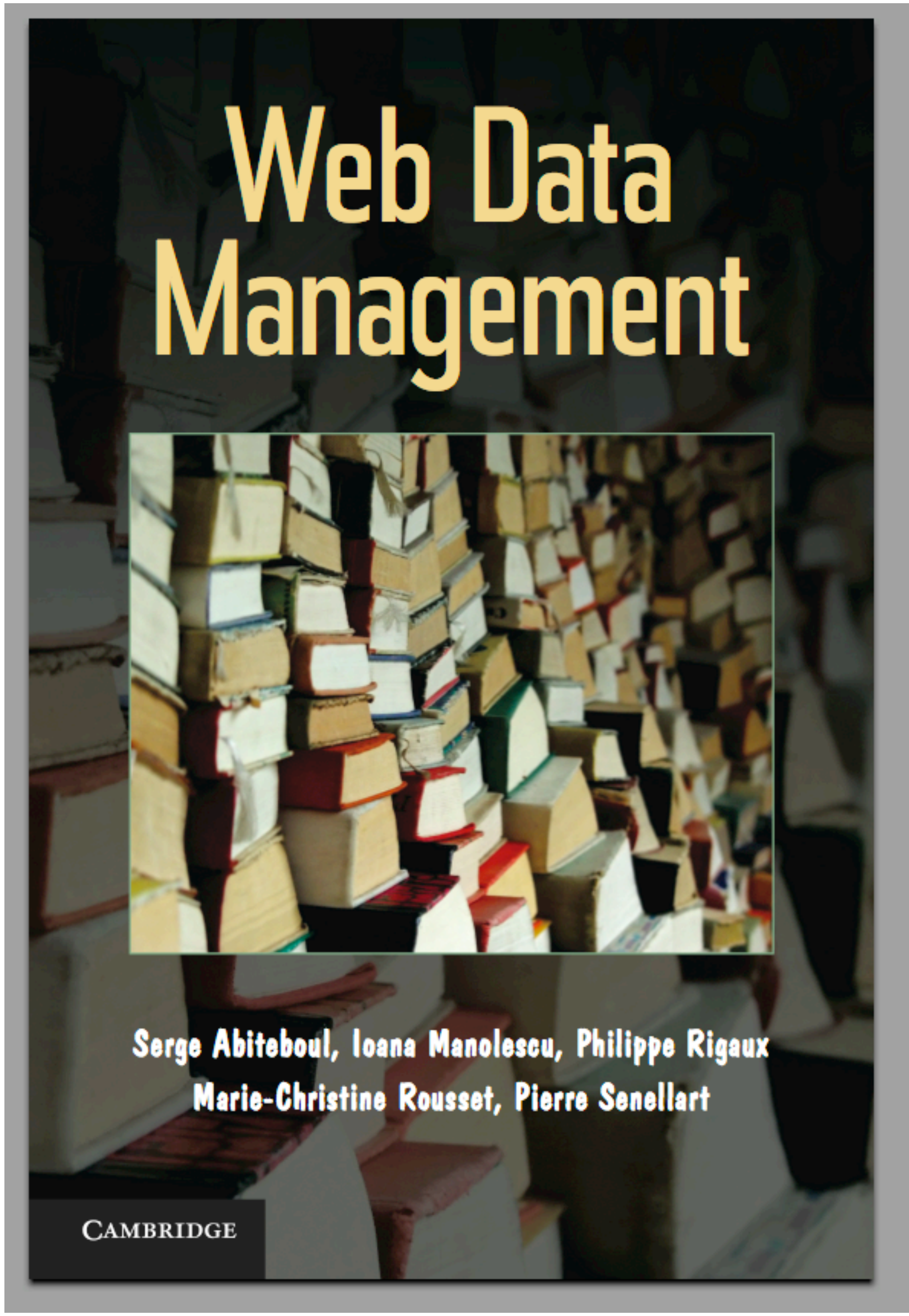
inria informatiques mathématiques

COLLÈGE DE FRANCE 1530

ENS CACHAN

After a short break

- Two authors of the Web Data Management Book (aka Jorge)
- Two friends



Marie-Christine Rousset
Reasoning in the Semantic Web

Professor of CS at the Univ. Grenoble.

PhD (1983) and a Thèse d'Etat (1988) in CS from Univ. Paris-Sud.

Best paper award from AAAI in 1996

Junior member of *Institut Universitaire de France* 1997-2001 and Senior member in 2011-now

Interest;: Knowledge Representation, Information Integration, Pattern Mining and the Semantic Web.

Pierre Senellart
Social networks

Associate Professor at Telecom ParisTech

PhD from Univ. of Paris-Sud

Information Director for Journal of ACM

Interest: Data management, Web data management, Probabilistic data

Interest: Natural language processing
Software Engineer at SYSTRAN SA