Datalog^4: Living with Inconsistency and Taming Nonmonotonicity

Jan Małuszyński and Andrzej Szałas

March 2010

- Introduction and motivations.
- Living with inconsistency.
  - Four-valued reasoning with $\mathfrak{t}$, $\mathfrak{f}$, $\mathfrak{u}$ and $\mathfrak{i}$.
  - Monotonic, intuitive and tractable rule language with unrestricted negation.
- Taming Nonmonotonicity.
  - Layered architecture.
  - Local Closed-World Assumption.
  - Lightweight nonmonotonic reasoning.
- Conclusions.

Why CWA?

- Efficient representation of negative information.
- Natural and intuitive in many application areas.

Why not CWA?

- Non-monotonicity not controlled by users.
- Not suitable for important areas including robotics, Semantic Web, multiagent systems.

Example

An autonomous vehicle approaches an intersection where there is no stop sign, yield sign or traffic signal. It should yield to vehicles coming from the right:

$$\mathit{halt}(X) \;{:}\!-\; \mathit{right}(X, Y). \quad \text{(Halt at intersection } X \text{ when there is a car } Y \text{ to the right.)}$$

If $\mathit{right}(X, Y) = \mathfrak{u}$ then, under $\textsc{Cwa}$, $\mathit{halt}(X) = \mathfrak{f}$.

Two truth values?

Example

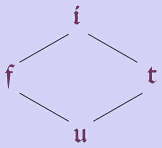
A web agent asks a Semantic Web service whether X is a reliable seller. What should be the answer when:

- the service has no information concerning the reliability of X ?
- the service has inconsistent information about X ?

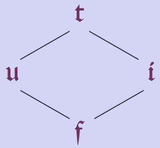
Remark

Such situations are typical for many information sources.
The semantics can be encoded using two truth values.
However, $\mathfrak{u}$ and $\mathfrak{i}$ remain more or less implicit there.

Orderings of truth values



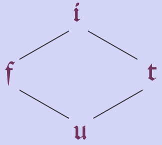
knowledge ordering ($\leq_k$)



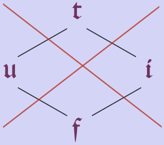
Belnap's truth ordering

Four-valued logic

Orderings of truth values

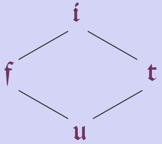


knowledge ordering ($\leq_k$)



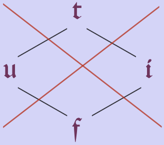
Belnap's truth ordering

Orderings of truth values



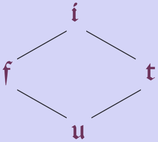
knowledge ordering ($\leq_k$)

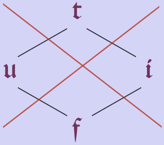
truth ordering ($\leq_t$)



Belnap's truth ordering

Orderings of truth values





knowledge ordering ($\leq_k$) truth ordering ($\leq_t$) Belnap's truth ordering

Conjunction and disjunction

The semantics of:

$$\left.\begin{array}{l} A \wedge B = \min\{A, B\} \\ A \vee B = \max\{A, B\} \end{array}\right\} \text{(w.r.t. truth ordering).}$$

E.g., $\mathfrak{t} \wedge \mathfrak{i} = \mathfrak{i}$, $\mathfrak{u} \vee \mathfrak{f} = \mathfrak{u}$, etc.

Semantics of negation and implication

$\rightarrow$	$\mathfrak{f}$	$\mathfrak{u}$	$\mathfrak{i}$	$\mathfrak{t}$
$\mathfrak{f}$	$\mathfrak{t}$	$\mathfrak{t}$	$\mathfrak{t}$	$\mathfrak{t}$
$\mathfrak{u}$	$\mathfrak{t}$	$\mathfrak{t}$	$\mathfrak{t}$	$\mathfrak{t}$
$\mathfrak{i}$	$\mathfrak{f}$	$\mathfrak{f}$	$\mathfrak{t}$	$\mathfrak{f}$
$\mathfrak{t}$	$\mathfrak{f}$	$\mathfrak{f}$	$\mathfrak{t}$	$\mathfrak{t}$

	$\neg$
$\mathfrak{f}$	$\mathfrak{t}$
$\mathfrak{u}$	$\mathfrak{u}$
$\mathfrak{i}$	$\mathfrak{i}$
$\mathfrak{t}$	$\mathfrak{f}$

Remark

Implication $B \rightarrow C$ is $\mathfrak{f}$ only when the conclusion C has to be corrected to satisfy the corresponding rule $C :\!- B$.

Discussion

We proposed $\rightarrow$ in our previous work with A. Vitória.
It reflects the following principles:

- new facts are not deduced from premises evaluating to $\mathfrak{f}$ or $\mathfrak{u}$
- a fact can be assigned $\mathfrak{t}$ only on the basis of premises evaluating to $\mathfrak{t}$
- true premises are allowed to imply inconsistency of a fact, since another rule can support the negation of this fact.

Deduction from unknown and false

Discussion continued

- Deduction from unknown leads to nonmonotonicity.
  It will later be allowed in a well controlled manner.
- Deduction from false is questionable. For example:

  $$late :– overslept.$$

  If deductions from false premises are allowed, then the falsity of *overslept* makes *late* false which is an incorrect conclusion both intuitively and in logic.

  (DATALOG provides the same result due to CWA.)
  In our semantics *late* remains unknown still satisfying the rule.

Definition

By an *interpretation* we mean any set of literals. *Truth value* of a literal ℓ in interpretation $\mathcal{I}$:

$$\mathcal{I}(\ell) \stackrel{\text{def}}{=} \begin{cases} \mathfrak{t} & \text{if } \ell \in \mathcal{I} \text{ and } (\neg\ell) \notin \mathcal{I} \\ \mathfrak{i} & \text{if } \ell \in \mathcal{I} \text{ and } (\neg\ell) \in \mathcal{I} \\ \mathfrak{u} & \text{if } \ell \notin \mathcal{I} \text{ and } (\neg\ell) \notin \mathcal{I} \\ \mathfrak{f} & \text{if } \ell \notin \mathcal{I} \text{ and } (\neg\ell) \in \mathcal{I}. \end{cases}$$

Extending the definition for all formulas

The *truth value* of a formula in interpretation $\mathcal{I}$ is defined as usual, using truth tables provided for $\neg, \wedge, \vee, \rightarrow$

Syntax of rules

In the sequel we consider ground rules only and assume that for each head ℓ there is only one rule of the form:

$$\begin{aligned} \ell \;{:}{-}\quad & (b_{11}, \ldots, b_{1i_1}) && \vee \\ & (b_{21}, \ldots, b_{2i_2}) && \vee \\ & \ldots && \vee \\ & (b_{m1}, \ldots, b_{mi_m}). \end{aligned} \tag{1}$$

Disjunction in (1) gathers all ground bodies with ℓ as the head.

Notation

Let ϱ be a rule of the form (1). Then:

- $head(\varrho) \stackrel{\text{def}}{=} \ell$
- $body(\varrho) \stackrel{\text{def}}{=}$
  $(b_{11}, \ldots, b_{1i_1}) \vee (b_{21}, \ldots, b_{2i_2}) \vee \ldots \vee (b_{m1}, \ldots, b_{mi_m})$

Four-valued semantics of rules

A set of literals $\mathcal{I}$ is a *model of a set of rules S* iff for each rule $\varrho \in S$ we have that $\mathcal{I}\big(body(\varrho) \rightarrow head(\varrho)\big) = \mathfrak{t}$, assuming that the empty body takes the value $\mathfrak{t}$ in any interpretation.

The monotonic layer: declarative semantics

Example

Let S be the following set of rules:

$$
\begin{aligned}
&wait \;:\!-\; overloaded \vee rest\_time\,.\\
&rest\_time \;:\!-\; wait\,.\\
&\neg overloaded \;:\!-\; rest\_time\,.\\
&overloaded\,.
\end{aligned}
$$

A minimal model of S is

$$\{overloaded, \neg overloaded, wait, rest\_time\}.$$

There are no facts supporting the truth of $wait$ and $rest\_time$ in this model. The intuitively correct model for S is

$$\{overloaded, \neg overloaded, wait, \neg wait, rest\_time, \neg rest\_time\}.$$

The monotonic layer: declarative semantics

Well-supported model (formal definition in the paper)

Intuitively, a *well-supported model* is a model where each literal has value $\mathfrak{t}$ or $\mathfrak{i}$ iff this is forced by a finite derivation starting from facts.

Theorem

For any set of rules S there is the unique well-supported model.

Theorem

Computing the well-supported model is in $\mathrm{PT{\small IME}}$ w.r.t. the size of the database domain.

The monotonic layer: computing the well-supported model

Algorithm

Input: a set of rules S

Output: the unique well-supported model $\mathcal{I}^S$ for S

1. (*finding basic inconsistencies*):
   - compute the least Herbrand model $\mathcal{I}_0^S$ of $Pos(S)$, where by $Pos(S)$ we understand the DATALOG program obtained from S by replacing each negative literal $\neg\ell$ of S by its (unique and fresh) duplicate ℓ'
   - let $\mathcal{I}_1^S \stackrel{\text{def}}{=} \{\ell, \neg\ell \mid \ell, \ell' \in \mathcal{I}_0^S\}$
2. (*finding potentially true literals*):
   - let $S' = \{\varrho \mid \varrho \in S \text{ and } \mathcal{I}_1^S(head(\varrho)) \neq \mathbf{i}\}$
   - set $\mathcal{I}_2^S$ to be the the least Herbrand model for $Pos(S')$

The monotonic layer: computing the well-supported model

Algorithm – continued

3. (*reasoning with inconsistency*):
   - define the following transformation Φ^S on interpretations:

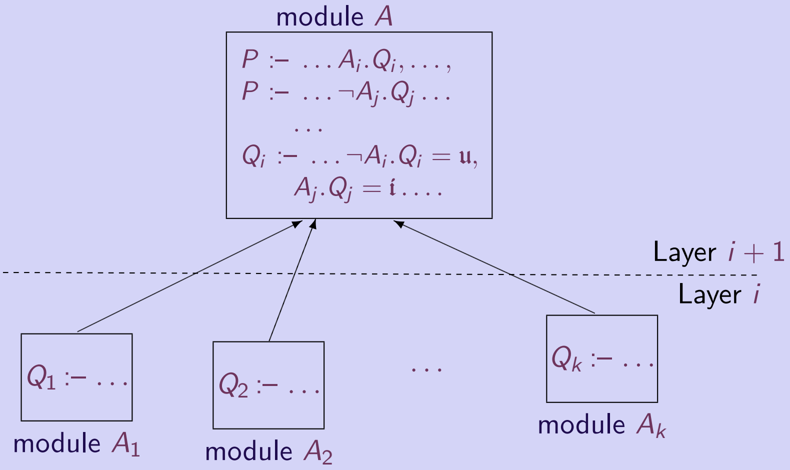
$$\Phi^S(\mathcal{I}) \stackrel{\text{def}}{=} \mathcal{I} \cup \{\ell, \neg\ell \mid \text{there is a rule } [\ell \text{ :- } b_1 \vee \ldots \vee b_m] \in S \text{ such that } \exists k \in \{1,\ldots,m\}[\mathcal{I}(b_k) = \mathfrak{i}\,] \text{ and } \neg\exists n \in \{1,\ldots,m\}[\,(\mathcal{I}_2^S - \mathcal{I})(b_n) = \mathfrak{t}\,]\}.$$

   - The transformation Φ^S is monotonic (!)
     Denote by $\mathcal{I}_3^S$ the fixpoint of Φ^S obtained by iterating Φ^S on $\mathcal{I}_1^S$, i.e.,

$$\mathcal{I}_3^S = \bigcup_{i\in\omega} (\Phi^S)^i(\mathcal{I}_1^S)$$

   - set $\mathcal{I}^S = \mathcal{I}_2^S \cup \mathcal{I}_3^S$.

The architecture



External literals

- External literals are crucial for expressing nonmonotonic rules.
- An *external literal* is of one of the forms:
$$A.R,\ \neg A.R,\ A.R \text{ IN } T,\ \neg A.R \text{ IN } T,$$
where:
  - A is a module (the *reference module* of the external literal) and R is a relation in A ($\neg A.R \text{ IN } T$ is to be read as "$(\neg A.R) \text{ IN } T$")
  - $T \subseteq \{\mathfrak{t}, \mathfrak{f}, \mathfrak{i}, \mathfrak{u}\}$ (if $T = \emptyset$ then $\ell \text{ IN } T$ is $\mathfrak{f}$).
- An external literal may only appear in rule bodies of a module B, provided that
  - its relation appears in the head of a rule in its reference module
  - its reference module is in a strictly lower layer than B.
- We write $\ell = v$ rather than $\ell \text{ IN } \{v\}$.

Semantics of modules and external literals

- Formally, relation symbol R occurring in module A is an abbreviation for $A.R$.
- Each module operates on its "local" relations, accessing "external" relations only via dotted notation.
- External literals, when used in a given module, are fully defined in modules in lower layers.
- Relations assigned to external literals, when used, cannot change.

Typical sources of nonmonotonicity

Generally, attempts to fill gaps in missing knowledge, e.g.,

- efficient representation of (negative) information (like CWA, LCWA)
- drawing rational conclusions from non-conclusive information (e.g., circumscription, default logics)
- drawing rational conclusions from the lack of knowledge (e.g., autoepistemic reasoning)
- resolving inconsistencies (e.g., defeasible reasoning).

Local Closed World Assumption

Intuitively, one often wants to contextually close part of the world, not necessarily all relations in the database.

Example

The following rules in module, say A, locally close *location*:

$$
\begin{aligned}
location(X,Y,T) :- \;& \neg B.chngPos(X,S)\ \text{IN}\ \{\mathfrak{u},\mathfrak{t}\}, house(X),\\
& nextTime(T,S), C.location(X,Y,S).\\
\neg location(X,Y,T) :- \;& \neg B.chngPos(X,S)\ \text{IN}\ \{\mathfrak{u},\mathfrak{f}\}, movingCar(X),\\
& nextTime(T,S), C.location(X,Y,S).
\end{aligned}
$$

Some results

Theorem

DATALOG^4 with modules has PTIME data complexity.

Theorem

Stratified DATALOG programs are expressible in DATALOG^4.

Remark

Stratified DATALOG captures PTIME on ordered structures.

Default rules

Default rules have the form:

$$\mathit{prerequisite} : \mathit{justification} \vdash \mathit{consequent},$$

with the intuitive meaning

"deduce *consequent* whenever *prerequisite* is true
and *justification* is consistent with current knowledge".

Example: expressing default-like rules

Default rule:

$$\mathit{car}(X) \wedge \mathit{speed}(X, \mathit{high}) : \mathit{onRoad}(X) \vdash \mathit{onRoad}(X)$$

captures similar intuitions as

$$\mathit{onRoad}(X) :\!- \mathit{car}(X), \mathit{speed}(X, \mathit{high}), B.\mathit{onRoad}(X) \text{ IN } \{\mathfrak{t}, \mathfrak{u}\}.$$

Defaults for resolving inconsistencies

Module B:

$$stop \mathrel{:-} red\_light.$$
$$\neg stop \mathrel{:-} policeman\_directs\_to\_go\_through.$$

Module A:

$$\neg stop \mathrel{:-} B.stop = \mathfrak{i}.$$

The idea

1. A typical pattern of autoepistemic reasoning:

   "If you do not know A, conclude $\neg A$."

2. The rule stating: "If you do not know that you have a sister, conclude that you do not have a sister" can be expressed in module $A \neq B$ by a rule assuming that knowledge of the reasoner is specified in module B:

   $$\neg have\_sister \;{:}\!-\; B.have\_sister = \mathfrak{u}.$$

Abnormality theories

In general, replacing circumscription by rules is not doable.
However, abnormality theories are typically expressed by formulas of the following pattern:

$$(condition \wedge \neg abnormal) \rightarrow conclusion.$$

In such cases one can:

- locally close abnormality
- make varied predicates heads of rules
  (this sometimes requires finding their definitions. Even if often can be done automatically, this is not a lightweight task).

Example

Consider the theory:

$$\forall X[(ill(X) \wedge \neg ab(X)) \rightarrow consults\_doctor(X)]$$

and assume one minimizes ab varying $consults\_doctor$. Let B be a module with (among others) the following rule:

$$ab(X) :- ill(X), \neg consults\_doctor(X).$$

We define a module A, consisting of rules:

$$\neg ab(X) :- B.ab(X) \text{ IN } \{\mathfrak{f}, \mathfrak{u}\}.$$
$$consults\_doctor(X) :- \neg ab(X), ill(X).$$

Example

Consider the following defeasible rules reflecting buyer's requirements as to apartments:

$$r1 : size(X, large) \Rightarrow acceptable(X)$$
$$r2 : \neg pets\_allowed(X) \Rightarrow \neg acceptable(X)$$

with priorities $r2 > r1$.

Example continued

Assume module B contains rules:

$$
\begin{array}{lcl}
acceptable(X) & :- & size(X, large). \\
\neg acceptable(X) & :- & \neg pets\_allowed(X).
\end{array}
$$

The following rules in some other module resolves possible inconsistencies according to required priority (but note that we have also cases with $\mathfrak{u}$, not covered by defeasible rules).

$$
\begin{array}{lcl}
acceptable(X) & :- & B.acceptable(X) = \mathfrak{t}. \\
\neg acceptable(X) & :- & B.acceptable(X) = \mathfrak{i}.
\end{array}
$$

The most relevant papers

- Departure point: our previous work with A. Vitória (Transactions on Rough Sets 2007, RSCTC 2008, RSKT 2008, Fundamenta Informaticae 2009): focussed on knowledge fusion and approximate reasoning (e.g., disjunction w.r.t. knowledge ordering, nonmonotonicity of disjunction w.r.t. truth ordering).
- S. Amo, M.S. Pais (Int. Journal of Approximate Reasoning 2007): use the same truth ordering, but assume CWA and only allow negation in the rule bodies.
- J. Alcântara, C.V. Damásio and L.M. Pereira (J. Applied Logic 2005): the focus on semantical integration of explicit and default negation.
- M.C. Fitting (Theoretical Computer Science 2002): syntactically the same programs, but uses Belnap's logic.

Conclusions

- The proposed DATALOG^4 is powerful but still lightweight and intuitive. It provides means for monotonic reasoning supported by facts together with a mechanism for expressing nonmonotonic rules.
- The intended methodology:
  - the lowest layer provides solid knowledge, supported by facts, e.g., reflecting perception, expert knowledge, etc.
  - higher layers allow one to derive conclusions still supported by facts or using various forms of nonmonotonic reasoning, usually reflecting expert knowledge.
- Open questions:
  - provide an efficient top-down query evaluation (e.g., resolution or tableaux-based).
    We have one, but it is complex (EXPTIME in the worst case)
  - is the provided algorithm for computing well-supported model time-optimal?