Non-monotonic Reasoning

王丛

文因互联 - Memect

2016 年 6 月 2 日

目录

Computational Logic is trying to solve:

- how to formulate the model
- how to compute the model

Computational Logic has 3 components:

- a well-defined *syntax*
- a well-defined *semantics*
- a well-defined *proof-theory*

Propositional Logic - Syntax

Example: $\neg A \wedge B \vee C \rightarrow X \wedge \neg Y$

An atomic formula is a propositional variable. Formulas are defined by the following inductive process.

- All atomic formulas are formulas.
- For every formula F, $\neg F$ is a formula.
- For all formulas F and G, also $(F \vee G)$ and $(F \wedge G)$ are formulas.

Propositional Logic - Semantics

Define a function $\mathcal{A} : \mathcal{A}(F) \rightarrow \{0, 1\}$, where F is a formula, and $\{0, 1\}$ represents True and False.

- $\mathcal{A}(F \wedge G) = 1$ if $\mathcal{A}(F) = 1$ and $\mathcal{A}(G) = 1$, otherwise 0
- $\mathcal{A}(F \vee G) = 1$ if $\mathcal{A}(F) = 1$ or $\mathcal{A}(G) = 1$, otherwise 0
- $\mathcal{A}(\neg F) = 1$ if $\mathcal{A}(F) = 0$, otherwise 0

Propositional Logic - Proof

- Truth table
- Tableau Algorithm (Top-Down Tree)
- Resolution Algorithm (Bottom-Up Tree)

Tableau - Unsatisfiable

Given $\{(a \vee \neg b) \wedge b, \neg a\}$. Every branch is closed (conflicted).

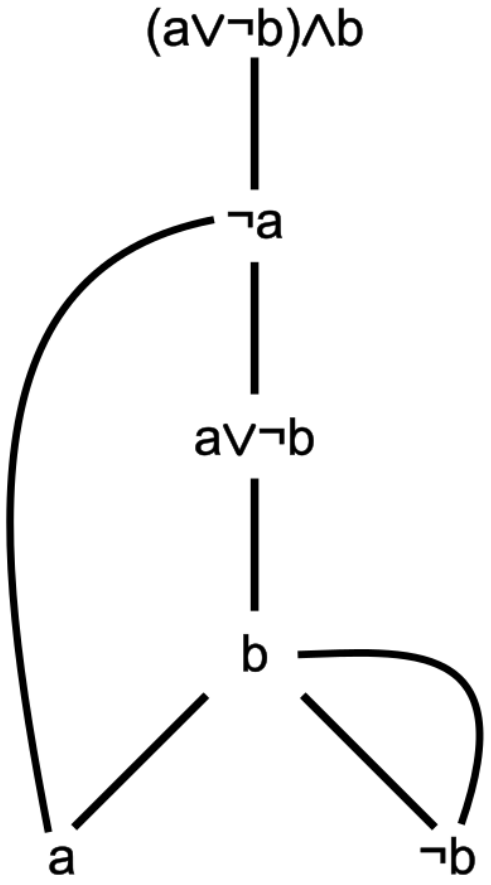
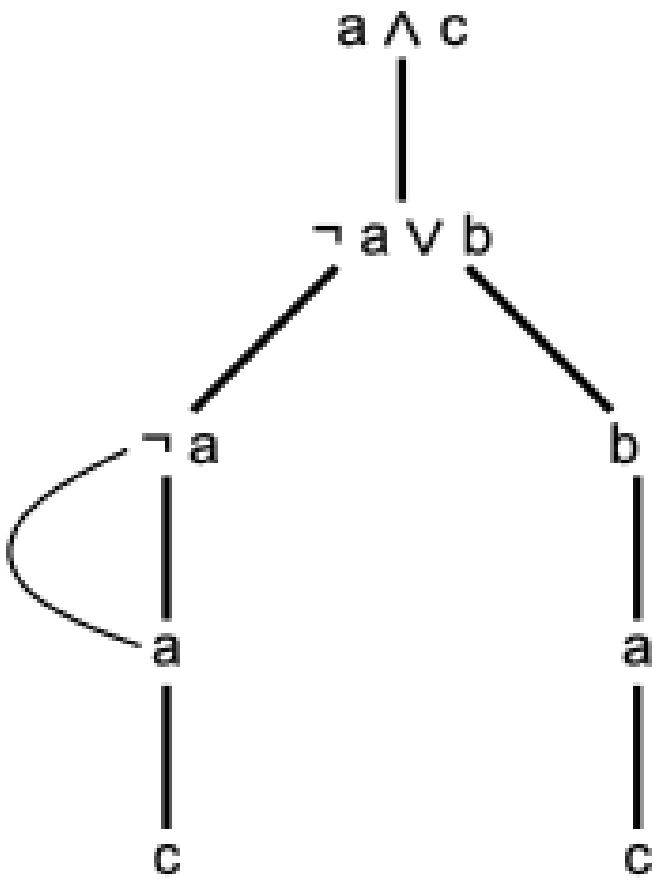


Tableau - Satisfiable

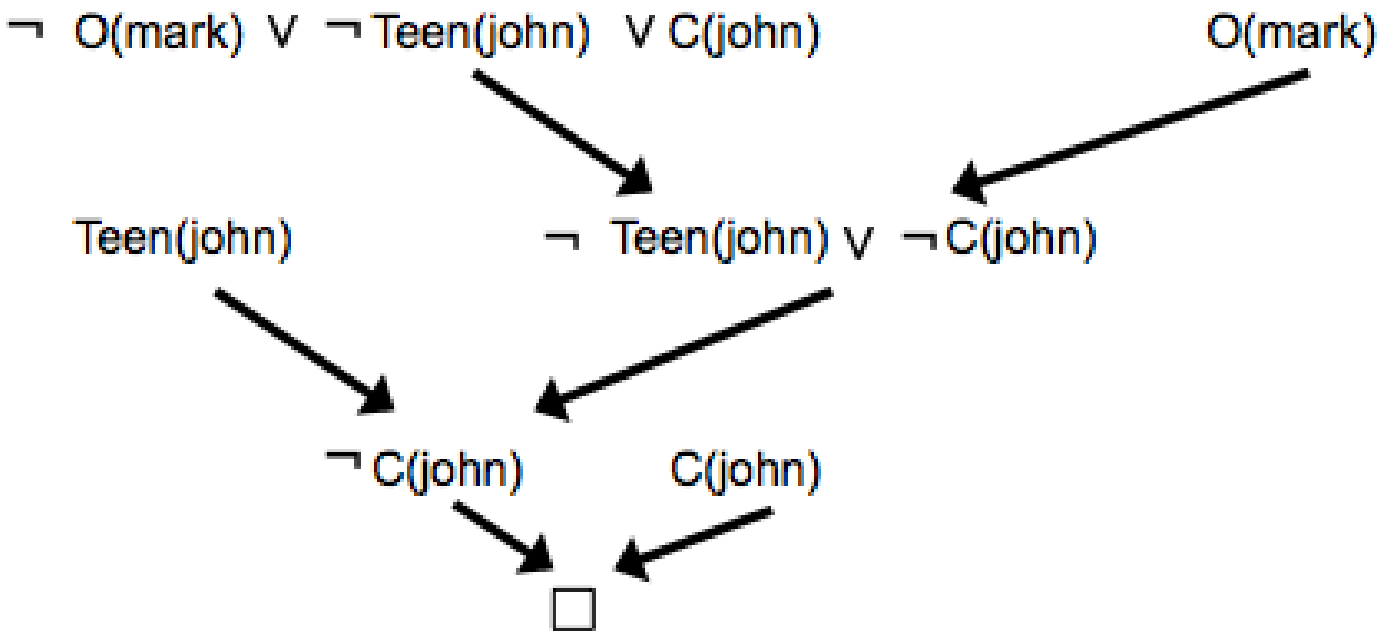
Given $\{(a \wedge c) \wedge (\neg a \vee b)\}$. Exist a branch is not closed.



Resolution

- Normalize all statement into Disjunction Normal Formal (DNF)
- Inferring between two DNF clause, one has a positive elment, the other has the negation of it.
- Inferring until no new clause can be inferred.

Resolution - Cont.



First-Order Logic

Example :

- Every cat is cute or extremely cute.

  $\forall x : Cat(x) \rightarrow Cute(x) \vee ECute(x)$

- Every cat has someone who loves it.

  $\forall x : Cat(x) \rightarrow \exists y : People(y) \vee Loves(y, x)$

First-Order Logic - Syntax

- $P(x_1, \ldots, x_k)$ is a formula.
- For each formula F, $\neg F$ is a formula.
- For all formulas F and G, $F \wedge G$ and $F \vee G$ are formulas.
- If x is a variable and F is a formula, $\exists x F$ and $\forall x F$ are formulas.

First-Order Logic - Semantics

Define U as a space, $\mathcal{A}$ as a function where mapping variables x in predicate P to some elements on the space, and mapping predicate P to a k-ary relation on the space.

- $\mathcal{A}(F(x_1, \ldots, x_k)) = 1$ if $(\mathcal{A}(x_1), \ldots, \mathcal{A}(x_k)) \in F^{\mathcal{A}}$, otherwise 0.
- $\mathcal{A}(F \wedge G) = 1$ if $\mathcal{A}(F) = 1$ and $\mathcal{A}(G) = 1$, otherwise 0.
- $\mathcal{A}(F \vee G) = 1$ if $\mathcal{A}(F) = 1$ or $\mathcal{A}(G) = 1$, otherwise 0.
- $\mathcal{A}(\neg F) = 1$ if $\mathcal{A}(F) = 0$, otherwise 0.
- $\mathcal{A}(\forall x F) = 1$ if for all $x \in U_{\mathcal{A}}$, $\mathcal{A}(F(x)) = 1$, otherwise 0
- $\mathcal{A}(]\exists x F) = 1$ if exists some $x \in U_{\mathcal{A}}$, $\mathcal{A}(F(x)) = 1$, otherwise 0

First-Order Logic - Semantics

$$F = \forall x \forall y (P(a) \wedge (P(x) \rightarrow Q(x, x)))$$

Example1: $U_{\mathcal{A}} = \mathcal{N}, P^{\mathcal{A}} = \mathcal{N}, Q^{\mathcal{A}} = \{(n, k) | n < k\}$

Example2: $U_{\mathcal{A}} = \{\text{☺}, \text{☹}\}$, $P^{\mathcal{A}} = U_{\mathcal{A}}$, $Q_{\mathcal{A}} = \{(\text{☹}, \text{☹})\}$

First-Order Logic - Proof

The problem “Given a First-formula F, is F valid?”is undecidable.
(proof by reduction of the Halting Problem.)

Reasoning Algorithms:

- Tableau
- Resolution

Monotonic Logic

The conclusion is not changed by the addition of premises.

- If $K \models F$, then $K \sqcup G \models F$.
- If $M \subseteq N$, then $\{F | M \models F\} \subseteq \{F | N \models F\}$.

Non-monotonic Logic

If $K \models F$, then not necessarily $K \sqcup G \models F$.

- Typically birds fly.
- Penguins do not fly.
- Tweety is a bird.
- Tweety flies.

Non-monotonic Logic

If $K \models F$, then not necessarily $K \sqcup G \models F$.

- Typically birds fly.
- Penguins do not fly.
- Tweety is a bird.
- Tweety flies.

Non-monotonic Logic

If $K \models F$, then not necessarily $K \sqcup G \models F$.

- Typically birds fly.
- Penguins do not fly.
- Tweety is a bird.
- Tweety flies.
- Tweety is a penguin.

Non-monotonic Logic

If $K \models F$, then not necessarily $K \sqcup G \models F$.

- Typically birds fly.
- Penguins do not fly.
- Tweety is a bird.
- Tweety flies.
- Tweety is a penguin.

Non-monotonic Logic

If $K \models F$, then not necessarily $K \sqcup G \models F$.

- Typically birds fly.
- Penguins do not fly.
- Tweety is a bird.
- ~~Tweety flies.~~
- Tweety is a penguin.

The previous conclusion must be retracted, such that Tweety does not fly will hold.

Non-monotonic History

Non-Monotonic logics have been proposed at the beginning of the 80's, here are historically the most important proposals.

- Non-monotonic logic, by McDermott and Doyle, '80
- Default Logic, by Reiter, '80
- Circumscription, by McCarthy, '80
- Autoepistemic logic, Moore, '84

Most of current research is based on these 3 main frames.

Frame Problem

Problem: How to represent that objects are not affected by state change?

Example: Moving an object does not change its color.

- $color(x, c, s) \rightarrow color(x, c, result(move, x))$
- $color(x, c, s) \rightarrow color(x, c, result(open\_door, x)$
- $color(x, c, s) \rightarrow color(x, c, result(a\_cat\_pass\_by, x)$

We need a great number of frame axioms

Frame Problem

Problem: How to represent that objects are not affected by state change?

Example: Moving an object does not change its color.

- $color(x, c, s) \rightarrow color(x, c, result(move, x))$
- $color(x, c, s) \rightarrow color(x, c, result(open\_door, x)$
- $color(x, c, s) \rightarrow color(x, c, result(a\_cat\_pass\_by, x)$

We need a great number of frame axioms

Frame Problem

We nee a *General axiom* of such form:

$$holds(p, s) \wedge \neg exception(p, a, s) \rightarrow holds(p, result(a, s))$$

But what action is not an exception?

We need a non-monotonic reasoning mechanism.

Frame Problem

We nee a *General axiom* of such form:

$holds(p, s) \wedge \neg exception(p, a, s) \rightarrow holds(p, result(a, s))$

But what action is not an exception?

We need a non-monotonic reasoning mechanism.

CWA and OWA

- OWA (Open World Assumption):

  Define: What is True only if it's known to be True

  Used in Semantic Web, assume no one has complete knowledge.

  Monotonic Logic.

CWA and OWA

- OWA (Open World Assumption):

  Define: What is True only if it's known to be True

  Used in Semantic Web, assume no one has complete knowledge.

  Monotonic Logic.

- CWA (Closed World Assumption):

  Define: What is not known to be True, is False.

  Used in Database System, whhich is assumed to be complete.

  Also called Negation as failure.

CWA and OWA

- OWA (Open World Assumption):

  Define: What is True only if it's known to be True

  Used in Semantic Web, assume no one has complete knowledge.

  Monotonic Logic.

- CWA (Closed World Assumption):

  Define: What is not known to be True, is False.

  Used in Database System, whhich is assumed to be complete.

  Also called Negation as failure.

  Non-monotonic Logic.

CWA and OWA

- OWA (Open World Assumption):

  Define: What is True only if it's known to be True

  Used in Semantic Web, assume no one has complete knowledge.

  Monotonic Logic.

- CWA (Closed World Assumption):

  Define: What is not known to be True, is False.

  Used in Database System, whhich is assumed to be complete.

  Also called Negation as failure.

  Non-monotonic Logic.

Default Logic

Default logic extends classical logic by non-standard inference rules. These rules allows one to express default properties.

$$\frac{bird(x) : fly(x)}{fly(x)}$$

if x is a bird and we can consistently assume that x flies then we can infer that x flies.

Default Logic

More generally we have rules of the form:

$$\frac{P(x) : J(x)}{C(x)}$$

that can be interpreted as: if $P(x)$ holds and $J(x)$ can be consistently assumed then we can conclude $C(x)$.

where, $P(x)$ is preprequisite; $J(x)$ is justifitcation; $C(x)$ is consequent.

Minimal Model

Consider $a \wedge (b \vee c)$, We have 3 models

- Model 1: $\{a, b, c\}$ not Minimal
- Model 2: $\{a, b\}$ Minimal
- Model 3: $\{a, c\}$ Minimal

Circumscription

- define an abnormality predicate *ab* in First-Order Logic.
- then minimize abnormality by minimizing ab's model.
- Circumscription is generalizaation of Closed world assumption.
- In fact, *ab* is a second-order logic predicate.

Circumscription

- $bird(tweety)$
- $(penguin(x) \rightarrow bird(x)$
- $penguin(x) \rightarrow \neg fly(x)$
- $bird(x) \wedge \neg expetional(x) \rightarrow fly(x)$

we could conclude that tweety flies: by considering only models in which there are as few as possible exceptional individuals

Autoepistemic Logic

Models the beliefs of an rational and introspective agent.

Example: "if tweety is a bird and we don't believe it can't fly, then it flies"

- $\mathcal{B}a$ will mean that we believe in a.
- $\neg\mathcal{B}a$ will mean that we don't believe in a.
- $\mathcal{B}\neg a$ will mean that we believe in $\neg a$.
- $\neg\mathcal{B}\neg a$ will mean that we don't believe in $\neg a$.
- $\neg\mathcal{B}\neg fly(tweety)$ We don't believe tweety can't fly.

Autoepistemic Logic

Models the beliefs of an rational and introspective agent.

Example: "if tweety is a bird and we don't believe it can't fly, then it flies"

- $\mathcal{B}a$ will mean that we believe in a.
- $\neg\mathcal{B}a$ will mean that we don't believe in a.
- $\mathcal{B}\neg a$ will mean that we believe in $\neg a$.
- $\neg\mathcal{B}\neg a$ will mean that we don't believe in $\neg a$.
- $\neg\mathcal{B}\neg fly(tweety)$ We don't believe tweety can't fly.

Thanks

因为是内部讲义，所以部分内容未严格定义

参考 1 http://www.lsis.org/olivetti/TEACHING/MASTER/INTRONMR1.pdf

参考 2 http://www.computational-logic.org/content/events/iccl-ss-2005/lectures/bonatti/iccl-slides.pdf