Artificial Intelligence 1

Prof. Dr. Frank Hopfgartner
Dr. Matthias Horbach

Institute for Web Science and Technologies (WeST)
University of Koblenz

Overview

Knowledge representation and reasoning

- Definition: *Knowledge representation and reasoning (KR) is the field of artificial intelligence (AI) dedicated to representing information about the world in a form that a computer system can utilize to solve complex tasks such as diagnosing a medical condition.* *[Wikipedia]*
- Goal: Formalisation of beliefs (knowledge) and automatic reasoning
- But we already looked at propositional and first-order logic, isn't that enough?

Recall: logics

Every logic (=formal system) has the following components:

1. Syntax: What are the possible statements?
   1.1 Signature: What symbols are allowed? ($S = \{\text{Anna}, \text{human}, \text{student}\}$)
   1.2 Grammar: how can symbols be combined in order to obtain complex statements? (student $\Rightarrow$ human)
2. Semantics: Which are the "true" statements? What is the relationship between true statements?
   2.1 Interpretations: Which symbol represents which concrete object? (Anna = "Anna Schmidt")
   2.2 Models: Which statement is true in a given constellation of objects?
   2.3 Reasoning: How can we infer new information?
3. Calculus: How can "reasoning" be implemented?

Limitations of classical logic 1/4

- Classical logic is monotonic:

$$\text{If } \alpha \vdash \beta \qquad \text{then} \qquad \alpha \wedge \gamma \vdash \beta$$

  for every formula γ.
- In other words: inferences are never retracted when new information is received
- Classical logic is binary:

$$\alpha \wedge (\alpha \Rightarrow \beta) \vdash \beta$$

  is always true without exception.
- Central concept in knowledge representation are *rules*
- Rules always have exceptions (and there are also exceptions for that)

Limitations of classical logic 2/4

Example

Let us model our knowledge about some animals:

- birds are animals
- penguins are birds
- birds usually fly
- penguins do not fly

Naive formalisation:

$$\forall X : (bird(X) \Rightarrow animal(X))$$
$$\forall X : (penguin(X) \Rightarrow bird(X))$$
$$\forall X : (bird(X) \Rightarrow flies(X))$$
$$\forall X : (penguin(X) \Rightarrow \neg flies(X))$$
$$penguin(tweety)$$

What is the problem?

Limitations of classical logic 3/4

Example

Knowledge base in propositional logic (simpler)

$$\begin{aligned}\phi =&(\mathit{bird} \Rightarrow \mathit{animal}) \land (\mathit{penguin} \Rightarrow \mathit{bird}) \land (\mathit{bird} \Rightarrow \mathit{flies}) \\ &\land (\mathit{penguin} \Rightarrow \neg \mathit{flies}) \land \mathit{penguin}\end{aligned}$$

Observation:

- Knowledge base is inconsistent: $\phi \vdash \mathit{flies}$ and $\phi \vdash \neg \mathit{flies}$
- The "rule" $\mathit{bird} \Rightarrow \mathit{flies}$ is not a classical implication, it is not universally valid
- What now?
- Model exceptions explicitly: $\mathit{bird} \Rightarrow \mathit{flies} \longrightarrow$ $\mathit{bird} \land \neg \mathit{penguin} \Rightarrow \mathit{flies}$
- What about emus, dodos, ostriches, ...?

Limitations of classical logic 4/4

- Explicit enumeration of exceptions is not feasible (hard to maintain, . . .)
- How do we humans do it?
- When encountering a new bird for the first time, we make the *default assumption* that it can fly
- Only when obtaining explicit information that the bird does not fly, we will revise our inference

Overview

Overview

- Default logics are logics that allow for *non-monotonic* reasoning

  Even if $\alpha \vdash \beta$ it may be that $\alpha \wedge \gamma \nvdash \beta$

- In the following, we consider the prototype of a default logic: Reiter's default logic.

  R. Reiter. A logic for default reasoning. Artificial Intelligence, 13:81–132, 1980.

Some additional notation for first-order logic 1/2

Let $\Sigma = (U, P, F)$ be a first-order signature, V a set of variables, and $\mathcal{L}(\Sigma, V)$ the corresponding first-order language.

Definition

A variable $X \in V$ is *free* in formula ϕ if there is an occurrence of X in ϕ which is not quantified. A variable $X \in V$ is *bound* in ϕ if it occurs in a sub-formula of the type $\forall X : \phi'$ or $\exists X : \phi'$.
A formula $\phi \in \mathcal{L}(\Sigma, V)$ is *closed* if every appearing variable is bound.

Example

The following two formulas are closed:

$$\forall X : (a(X) \wedge \forall Y : b(X, Y)) \qquad\qquad r(s, t) \vee \exists Y : a(Y)$$

The following formulas are not closed

$$(\forall X : a(X)) \wedge b(X) \qquad\qquad r(s, Y) \vee \exists X : a(X)$$

Some additional notation for first-order logic 2/2

A set of first-order formulas is usually seen as equivalent with the conjunction of its elements

$$\{\phi_1, \phi_2, \phi_3\} \vdash \psi \qquad \Longleftrightarrow \qquad \phi_1 \wedge \phi_2 \wedge \phi_3 \vdash \psi$$

Definition

The *deductive closure* of a set of first-order formulas $\Psi \subseteq \mathcal{L}(\Sigma, V)$ is defined via $Cn(\Psi) = \{\phi \mid \Psi \vdash \phi\}$.

Observe that $\Psi \subseteq Cn(\Psi)$ is always the case.

Definition

A set of first-order formulas $\Psi \subseteq \mathcal{L}(\Sigma, V)$ is *deductively closed* if $Cn(\Psi) = \Psi$.

Default rules

Definition

Let $\phi, \psi_1, \ldots, \psi_n, \chi \in \mathcal{L}(\Sigma, V)$ be closed first-order formulas (or propositional formulas).

Then

$$\delta = \frac{\phi : \psi_1, \ldots, \psi_n}{\chi}$$

is called a *default rule* (or just *default*).

Meaning: If ϕ is known and $\psi_1, \ldots, \psi_n$ can be consistently assumed, then conclude χ.

$\phi = \mathit{pre}(\delta)$	(default) precondition
$\chi = \mathit{cons}(\delta)$	(default) conclusion
$\{\psi_1, \ldots, \psi_n\} = \mathit{just}(\delta)$	(default) justifications

Default rules - example

- Rules with exceptions:

$$\frac{bird : flies}{flies}$$

- Rules that are typically true:

$$\frac{going\_to\_work : take\_bus}{take\_bus}$$

- Rules that hold unless the opposite can be shown:

$$\frac{accused : innocent}{innocent}$$

Default schema

A default

$$\delta = \frac{\phi : \psi_1, \dots, \psi_n}{\chi}$$

with *open* formulas $\phi, \psi_1, \dots, \psi_n, \chi$ is interpreted as a schema, i. e., as its set of grounded defaults (over the given universe).

Default schema - example

The default rule

$$\frac{friend(X,Y) \wedge friend(Y,Z) : friend(X,Z)}{friend(X,Z)}$$

is a shorthand for (given $U = \{tom, bob, sally, tina\}$)

$$\frac{friend(tom, bob) \wedge friend(bob, sally) : friend(tom, sally)}{friend(tom, sally)}$$
$$\frac{friend(tom, bob) \wedge friend(bob, tina) : friend(tom, tina)}{friend(tom, tina)}$$

$\ldots$

Default theory (syntax of default logic)

Definition

A default theory T is a tuple $T = (W, \Delta)$ with

- $W \subseteq \mathcal{L}(\Sigma, V)$ (facts)
- Δ set of default rules

Main idea of default reasoning (semantics)

- apply defaults on W to extend the knowledge with plausible additional information
- apply defaults as long as no further information can be added

The resulting set of classical formulas is called *extension* and represents a plausible "belief state" of the default theory.

Main challenges

$$\delta = \frac{\phi : \psi_1, \ldots, \psi_n}{\chi}$$

Meaning: If ϕ is known and $\psi_1, \ldots, \psi_n$ can be consistently assumed, then conclude χ.

When is such a default δ applicable, i. e.

- When is ϕ known?
- When can $\psi_1, \ldots, \psi_n$ be consistently assumed?

Approach: use a (at first unknown) extension E to check these conditions:

- ϕ is known iff $\phi \in E$;
- $\psi_1, \ldots, \psi_n$ can be consistently assumed iff $\neg\psi_i \notin E$, $1 \leq i \leq n$.

Semantics of default logic: extensions 1/3

An extension $E \subseteq \mathcal{L}(\Sigma, V)$ is characterised by the following properties:

- E contains all facts: $W \subseteq E$
- E is deductively closed: $Cn(E) = E$
- E is *closed under default application*, i. e. if $\delta = \frac{\phi:\psi_1,\ldots,\psi_n}{\chi} \in \Delta$ is applicable in E then $\chi \in E$ where:

  δ is applicable in E iff $\phi \in E$ and $\neg\psi_1 \notin E, \ldots, \neg\psi_n \notin E$

Semantics of default logic: extensions 2/3

More general:

- Let F be a deductively closed set of formulas
- Let K be a set of formulas (the context)

A default $\delta = \frac{\phi:\psi_1,\ldots,\psi_n}{\chi}$ is *applicable* in F wrt. K iff

$$\phi \in F \quad \text{and} \quad \neg\psi_1 \notin K, \ldots, \neg\psi_n \notin K$$

Let $T = (W, \Delta)$ be a default theory and S a set of formulas.

Define $\Lambda_T(S)$ to be the smallest set of formulas with

- $\Lambda_T(S)$ is deductively closed
- $W \subseteq \Lambda_T(S)$
- $\Lambda_T(S)$ is closed under default application wrt. the context S, i. e. for all $\delta = \frac{\phi:\psi_1,\ldots,\psi_n}{\chi} \in \Delta$, if $\phi \in \Lambda_T(S)$ and $\neg\psi_1 \notin S$,..., $\neg\psi_n \notin S$, then $\chi \in \Lambda_T(S)$.

Definition

E is an *extension* of $T = (W, \Delta)$ iff $\Lambda_T(E) = E$.

Remark: conceptually, extensions of default theories correspond to models of a classical logic formula.

Example

$$T = (\{aquatic\_creature\}, \{\frac{aquatic\_creature : fish}{fish}\})$$

$E = Cn(\{aquatic\_creature, fish\})$ is an extension of T,

$E' = Cn(\{aquatic\_creature, \neg fish\})$ is *not* an extension of T, despite the fact that

- $\{aquatic\_creature\} \subseteq E'$
- E' is deductively closed
- E' is closed under default application

but $\Lambda_T(E') = Cn(\{aquatic\_creature\}) \neq E'$

More examples

Example

$T = (\{p\}, \{\frac{p:q}{q}, \frac{q:r}{r}\})$

- $E = Cn(\{p, q, r\})$

Example

$T = (\{p\}, \{\frac{p:q}{q}, \frac{p:\neg q}{\neg q}\})$

- $E_1 = Cn(\{p, q\})$
- $E_2 = Cn(\{p, \neg q\})$

Example

$T = (\{p\}, \{\frac{p:r}{s}\})$

- $E = Cn(\{p, s\})$

A characterisation of extensions

Theorem (Reiter, 1980)

Let E be a set of closed formulas and let $T = (W, \Delta)$ be a default theory. Define a sequence of sets of formulas E_i, $i \geq 0$ via

- $E_0 = W$
- $E_{i+1} = Cn(E_i) \cup \{\chi \mid \frac{\phi:\psi_1,\ldots,\psi_n}{\chi} \in \Delta, \phi \in E_i,$
  $\neg\psi_1 \notin E, \ldots, \neg\psi_n \notin E\}$

Then E is an extension of T iff

$$E = \bigcup_{i=0}^{\infty} E_i$$

Computing extensions

Task: given $T = (W, \Delta)$, enumerate all extensions E of T

- We discuss an algorithm to explicitly enumerate all extensions using *default processes* and *process trees*
- Idea: apply defaults successively
  - as long as possible
  - use backtracking if inconsistency occurs

Let $T = (W, \Delta)$ be a fixed default theory.

Default sequences

$\Pi = (\delta_0, \ldots, \delta_m)$ sequence of defaults from Δ (finite, no repetitions)

$\Pi[k] = (\delta_0, \ldots, \delta_{k-1})$ sub-sequence of the first k elements

Definition

Let Π be a default sequence, define

$$In(\Pi) = Cn(W \cup \{cons(\delta) \mid \delta \in \Pi\})$$
$$Out(\Pi) = \{\neg\psi \mid \psi \in just(\delta), \delta \in \Pi\}$$

- $In(\Pi)$ collects formulas that are concluded by applying defaults from Π; represents the current belief state after processing Π
- $Out(\Pi)$ collects formulas that should be not be proven true later

Default sequences - example

$T = (W, \Delta)$ with

$$W = \{a\} \qquad \Delta = \{\delta_1 = \frac{a : \neg b}{\neg b}, \delta_2 = \frac{a : c}{c}\}$$

Let $\Pi_1 = (\delta_1)$, $\Pi_2 = (\delta_2, \delta_1)$.

$$In(\Pi_1) = Cn(\{a, \neg b\}) \qquad Out(\Pi_1) = \{b\}$$

$$In(\Pi_2) = Cn(\{a, c, \neg b\}) \qquad Out(\Pi_2) = \{\neg c, b\}$$

Note: $In(\Pi)$, $Out(\Pi)$ are independent of the actual order of defaults in Π

Definition

$\Pi = (\delta_0, \ldots, \delta_m)$ is a *process* iff every δ_k is applicable in $In(\Pi[k])$.

In particular, δ_0 must be applicable in $In(()) = Cn(W)$.

A process Π is called

- *successful* iff $In(\Pi) \cap Out(\Pi) = \emptyset$;
- *not successful* (or *has failed*) iff $In(\Pi) \cap Out(\Pi) \neq \emptyset$;
- *closed* iff every $\delta \in \Delta$ that is applicable in $In(\Pi)$ appears in Π.

Theorem

E is an extension of T iff there is a closed and successful process Π with $E = In(\Pi)$.

Proof.

$\Leftarrow$:

Let Π be a closed and successful process of T with $E = In(\Pi)$. We have to show $\Lambda_T(E) = E$:

- $\Lambda_T(E) \subseteq E$:
  E is deductively closed and $W \subseteq E$; as Π is closed, E is closed wrt. default application. $\Lambda_T(E)$ is defined to be the *smallest* such set, so $\Lambda_T(E) \subseteq E$.
- $E \subseteq \Lambda_T(E)$:
  By induction $In(\Pi[k]) \subseteq \Lambda_T(E)$ for all k. It follows $E \subseteq \Lambda_T(E)$.

$\Rightarrow$:

Let $E = \Lambda_T(E)$ be an extension of T.
Let $\Delta = \{\delta_0, \ldots, \delta_n\}$ finite, arbitrary.
Construct process Π of T with $In(\Pi[k]) \subseteq E$ and $Out(\Pi[k]) \cap E = \emptyset$ as follows:

- $\Pi[0] = ()$;
- Let $\Pi[k]$ be constructed. If every default $\delta \in \Delta$ that is applicable in $In(\Pi[k])$ wrt. E is already in $\Pi[k]$, define $\Pi = \Pi[k]$ and halt. Otherwise select any default δ that is applicable in $In(\Pi[k])$ wrt. E and define $\Pi[k+1] = (\Pi[k], \delta)$.

In the end, $E = In(\Pi)$. $\square$

Processes - examples

$T = (W, \Delta)$ with

$$W = \{a\}$$
$$\Delta = \{\delta_1 = \frac{a : \neg b}{\neg b}, \delta_2 = \frac{\top : c}{b}\}$$

- the process $\Pi_1 = (\delta_1)$
  - is successful: $In(\Pi_1) \cap Out(\Pi_1) = Cn(\{a, \neg b\}) \cap \{b\} = \emptyset$
  - is not closed as δ_2 is applicable in $In(\Pi_1)$
- the process $\Pi_2 = (\delta_1, \delta_2)$
  - is not successful:
    $In(\Pi_2) \cap Out(\Pi_2) = Cn(\{a, \neg b, b\}) \cap \{b, \neg c\} = \{b\} \neq \emptyset$
  - is closed
- the process $\Pi_3 = (\delta_2)$
  - is successful: $In(\Pi_3) \cap Out(\Pi_3) = Cn(\{a, b\}) \cap \{\neg c\} = \emptyset$
  - is closed
  - $E = In(\Pi_3) = Cn(\{a, b\})$ is extension of T

Process trees

Process trees give an overview on all possible processes of a default theory $T = (W, \Delta)$:

- every node represents a process Π and is annotated with two labels: $In(\Pi)$ and $Out(\Pi)$;
- the root represents the empty process $\Pi = ()$ with $In(()) = Cn(W)$ and $Out(()) = \emptyset$;
- every application of a default induces a branch in the tree
- every leaf represents either
  - a failed process or
  - a closed and successful process

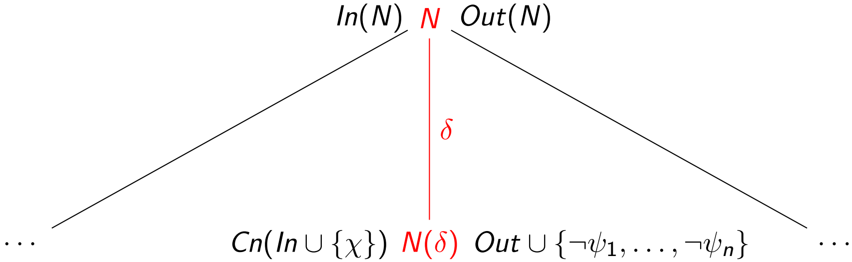
Construction of process trees

1. Construct a tree with a root N_0 and $In(N_0) = Cn(W)$ and $Out(N_0) = \emptyset$
2. As long as there is a leaf node N that is not marked with "failure" or "closed and successful", repeat
   - If $In(N) \cap Out(N) \neq \emptyset$: mark node with "failure"
   - $In(N) \cap Out(N) = \emptyset$
     - for every applicable default $\delta = \frac{\phi:\psi_1,\ldots,\psi_n}{\chi} \in \Delta$ that has not yet been considered in the process, add a child node $N(\delta)$ to N with

$$In(N(\delta)) = Cn(In(N) \cup \{\chi\})$$
$$Out(N(\delta)) = Out(N) \cup \{\neg\psi_1, \ldots, \neg\psi_n\}$$

     - Is there no further applicable default that has not yet been considered, mark N "closed and successful"; $In(N)$ is then an extension

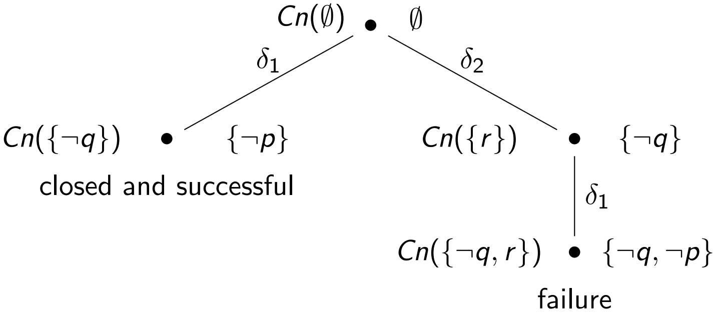
Process tree



$$\delta = \frac{\varphi : \psi_1, \dots, \psi_n}{\chi}$$

Process tree - example 1

$$T: \qquad W = \emptyset \qquad \Delta = \{\delta_1 = \frac{\top : p}{\neg q}, \delta_2 = \frac{\top : q}{r}\}$$



Process tree - example 2

$$T: \quad W_0 = \{penguin \Rightarrow bird, penguin \Rightarrow \neg flies, bird\}$$

$$\Delta = \{\delta_1 = \frac{bird : flies}{flies}\}$$

$Cn(W_0)$ • $\emptyset$

|

$Cn(W_0 \cup \{flies\})$ • $\{\neg flies\}$

closed and successful

For $W_1 = W_0 \cup \{penguin\}$
note:

$Cn(W_1)$ is the only extension of $T_1 = (W_1, \Delta)$ as $\neg flies \in Cn(W_1)$.

Properties of default logic 1/2

Theorem (Minimality of extensions)

Let E, E' be extensions of a default theory T with $E \subseteq E'$. Then $E = E'$.

Theorem (Uniqueness of extensions)

Let $T = (W, \Delta)$ be a default theory a let

$$W \cup \{\psi_1 \wedge \ldots \psi_n \wedge \chi \mid \frac{\phi : \psi_1, \ldots, \psi_n}{\chi} \in \Delta\}$$

be classically consistent. Then T has exactly one extension.

Properties of default logic 2/2

Theorem (Inconsistency 1)

A default theory $T = (W, \Delta)$ *has an inconsistent extension iff* W *is already inconsistent.*

Theorem (Inconsistency 2)

If T *has an inconsistent extension* E *then* E *is the only extension of* T.

$$T: \qquad W = \emptyset \qquad\qquad \Delta = \{\delta_0 = \frac{\top : a}{a}\}$$

T has exactly one extension $E = Cn(\{a\})$.

Add defaults to T:

- $\Delta_1 = \{\delta_0, \delta_1 = \frac{\top:b}{\neg b}\}$. $T_1 = (W, \Delta_1)$ has *no extensions*
- $\Delta_2 = \{\delta_0, \delta_2 = \frac{b:c}{c}\}$. $T_2 = (W, \Delta_2)$ has still exactly one extension E
- $\Delta_3 = \{\delta_0, \delta_3 = \frac{\top:\neg a}{\neg a}\}$. $T_3 = (W, \Delta_3)$ has two extensions E and $Cn(\{\neg a\})$.
- $\Delta_4 = \{\delta_0, \delta_4 = \frac{a:b}{b}\}$. $T_4 = (W, \Delta_4)$ has the extension $Cn(\{a, b\})$ which is a superset of E.

Extending a set of defaults (or the facts) can therefore

- remove extensions
- modify extensions
- create new extensions

Remember: this was the main motivation for default logic.

However, sometimes this behaviour may be to unpredictable.

Definition

Let $T = (W, \Delta)$ and $T' = (W, \Delta')$ default theories with the same set of facts and defaults $\Delta \subseteq \Delta'$. If every extension of T is contained in some extension of T', then T' is called a *semi-monotone* extension of T.

In general, we can also not expect a semi-monotonic behaviour in default logic.

Normal defaults

Definition

A default δ is called *normal* if $\mathit{just}(\delta) = \mathit{cons}(\delta)$, so δ is of the form

$$\delta = \frac{\phi : \psi}{\psi}$$

Example

$$\frac{\mathit{bird} : \mathit{flies}}{\mathit{flies}}$$

Using a normal default we can conclude a formula ψ if ψ is consistent with the current beliefs.

Processes of normal default theories

Let $T = (W, \Delta)$ be a normal default theory (=contains only normal defaults) with a consistent set of facts W.

Let $\Pi = (\delta_0, \ldots, \delta_n)$ be a process of T with $\delta_i = \frac{\phi_i : \psi_i}{\psi_i}$.

Remember:

$$
\begin{aligned}
In(\Pi) &= Cn(W \cup \{\psi_i\}_{i \geq 0}) \\
Out(\Pi) &= \{\neg\psi_i\}_{i \geq 0}
\end{aligned}
$$

Can Π be a failure?

Every default δ_i was applicable, so $\neg\psi_i \notin In(\Pi)$. It follows $In(\Pi) \cap Out(\Pi) = \emptyset$ and therefore:

Theorem

Every process of a normal default theory is successful.

Extensions of normal default theories

Theorem

A normal default theory always possesses at least one extension. Every finite process can be extended to a closed and successful process.

Theorem

Normal default theories are semi-monoton (adding another normal default extends previous extensions or preserves them completely).

Chapter 4.1: Default logic

Summary

Chapter 4.1: Summary

- Default rules of the form

$$\delta = \frac{\phi : \psi_1, \ldots, \psi_n}{\chi}$$

  represent plausible (but not necessarily generally valid) rules
- Extensions: deductively closed, include facts, closed under default application
- Fix point characterisation of extensions
- Computing extensions with process trees
- Normal default theories and semi-monotony