

Logic, Advanced Course (TDDB08)<http://www.ida.liu.se/~TDDB08/>

Lectures given by: Andrzej Szalas

“It is reasonable to hope that the relationship between computing and logic will be as fruitful in the next century as that between analysis and physics in the last.”
(J. McCarthy, 1960)

Contents

1. What is logic? classical propositional logic, classical first- and second-order logic.
2. Logic and databases, complexity of querying databases.
3. Fixpoint calculus as a database query language.
4. Propositional modal logics.
5. Quantified modal logics.
6. Propositional three-valued logics.
7. Propositional many-valued logics.
8. Intuitionism and logic.
9. Approximate relations and approximate reasoning.
10. Elements of commonsense reasoning.

What is logic?

The first approximation: *logic* is the science of correct reasoning, i.e., reasoning based on correct (sound) arguments. A *correct (sound)* argument is one in which anyone who accepts its premises should also accept its conclusions. Thus logics formalize how *consequences* of particular assumptions are defined, derived and (if possible) mechanized.

To see whether an argument is correct, one looks at the connection between the premisses and the conclusion.

One does not judge whether there are good reasons for accepting the premisses, but whether a person who accepted the premisses, for whatever reasons, good or bad, ought also accept the conclusion.

Examples

1. Correct arguments:
 - if x is a parent of y , and y is a parent of z , then x is a grandparent of z
 - if A and B are true, then A is true.
2. Incorrect arguments:
 - if A implies B then B implies A
 - if A is true or B is true, then A is true.
3. Are the following arguments correct?
 - if A implies B then not B implies not A
 - if A is true, then A or B is true.

What is logic? – continued

Logical formalisms are applied in many areas of science as a basis for clarifying and formalizing reasoning. Intuitively, a logic is defined by the (family of) language(s) it uses and by its underlying reasoning machinery.

The intensive use of formal reasoning techniques resulted in defining hundreds, if not thousands, of logics that fit (or do not) to particular application areas.

We then first need to clarify what do we mean by a logic. In order to make any reasoning fruitful, we have

1. to decide what is the subject of reasoning or, in other words, what are we going to talk about and what language are we going to use
2. to associate a precise meaning to basic notions of the language, in order to avoid ambiguities and misunderstandings
3. to state clearly what kind of opinions (sentences) can be formulated in the language we deal with and, moreover, which of those opinions are consequences of other opinions, and which are not.

Now we can investigate the subject of reasoning via the consequences of expressed opinions. Such an abstraction defines a specific logic.

What is logic? – continued

Traditionally, there are two methodologies to introduce a logic:

- *syntactically*, where consequences are defined via notions of a proof and proof system
- *semantically*, where consequences are defined via notions of a model, satisfiability and truth.

Both methodologies first require to chose a language that suits best a particular application. For example,

1. talking about politics we use terms “political party”, “prime minister”, “parliament”, “statement”, etc. etc.
2. talking about computer science phenomena we use terms “software”, “program execution”, “statement”, etc. etc.

Of course we use different vocabulary talking about different areas.

Logical language is defined by means of basic concepts, formulas and logical connectives or operators. Connectives and operators have a fixed meaning. Vocabularies reflecting particular application domains are flexible.

What is a logical language?

Language of a logic is a formal language, reflecting natural language phenomena and allowing one to formulate sentences (expressions) (called *formulas*) about a particular domain of interest.

Example

Language of arithmetics of natural numbers consists of:

- constants, e.g., 0, 1, 2, 3, ...
- variables, e.g., l, m, n, k, \dots
- function symbols, e.g., addition (+) or multiplication (*)
- relation symbols, e.g., = or \leq .

Examples of formulas:

$$\begin{aligned} n &\leq 2 * n + 1 \\ n * (k + 1) &= n * k + n \\ n * n * n + 2 * n * n + 3 &= 20 \end{aligned}$$

However, $n^3 + 2 * n^2 + 3 = 20$ is a formula only if n^3 and n^2 are operations allowed in the language.

Why “function/relation symbols” instead of “functions/relations”?

In natural language names are not objects they denote!

Examples

1. Name “John” is not a person named “John”.
2. An object can have many names,
 - e.g., “John” and “father of Jack” can denote the same person.
3. An object may have no name — for example?
4. Many different objects may have the same name,
 - e.g., “John” denotes many persons.
5. Some names do not denote any real-world objects,
 - e.g., “Pegasus”.

In logic *symbols* correspond to names.
A function/relation symbol is not a function/relation, but its name.
However, comparing to natural language, usually **a symbol denotes a unique object.**

Elements of logical language

Building blocks of logical languages are usually among the following:

- *individual constants* (*constants*, for short), representing particular objects,
 - examples: 0, 1, *John*
- *variables*, representing a range of objects, sets of objects, sets of sets of objects, etc.
 - examples: x, y, m, n
- *function symbols*, representing functions,
 - examples: +, *, *father()*
- *relation symbols*, representing relations,
 - examples: =, \leq , \preceq , *Older*, *Smaller*
- *logical constants*: TRUE, FALSE, sometimes also other,
 - examples: UNKNOWN, INCONSISTENT
- *connectives* and *operators*, allowing one to form more complex formulas from simpler formulas,
 - examples of connectives: “and”, “or”, “implies”,
 - examples of operators: “for all”, “exists”, “is necessary”, “always”
- *auxiliary symbols*, making notation easier to understand
 - examples: “(”, “)”, “[”, “]”.

Terms (expressions)

Expressions formed using individual constants, variables and function symbols, are called *terms*. Intuitively, a term represents a value of the underlying domain.
A *ground term* is a term without variables.

Examples

1. In arithmetics the following expressions are terms:

$$\begin{aligned} 1 + 2 + 3 + 4 * 2 &\quad \text{– a ground term} \\ (n + 2 * k) * 5 + n * n &\quad \text{– a term (but not ground)} \end{aligned}$$

but the following are not:

$$\begin{aligned} 1 + 2 + 3 + \dots + 10 &\quad \text{– “...” is not a symbol in arithmetics} \\ (n + 2 * k) * 5 \leq 20 &\quad \text{– “≤” is a relation symbol.} \end{aligned}$$

2. If *father* and *mother* are function symbols and *John* is a constant then the following expressions are (ground) terms:

$$\begin{aligned} &father(John) \\ &mother(father(John)) \\ &father(father(father(John))) \\ &mother(father(mother(John))). \end{aligned}$$

Formulas (sentences)

Expressions formed using logical constants and relation symbols, are called *formulas*. Intuitively, a formula represents an expression resulting in a logical value. A *ground formula* is formula built from relation symbols applied to ground terms only.

Examples

1. In arithmetics the following expressions are formulas:

$1 + 2 + 3 + 4 * 2 < 122$ – a ground formula
 $(n + 2 * k) * 5 + n * n = n + 2$ – a formula (but not ground),

but the following are not:

$1 + 2 + 3 + 4 * 2$
 $(n + 2 * k) * 5 + n * n$

although are well-formed terms.

2. If *father* and *mother* are function symbols, *older* is a relation symbol and *John* is a constant then the following expressions are (ground) formulas:

$older(father(John), John)$
 $older(mother(father(John), father(John)))$
 $older(father(John), mother(John))$.

What is “propositional”, “zero-order”, “first-order”, “second-order” etc.?

- In *propositional logics* we deal with a sub-language, where the building blocks are *propositional variables* (representing logical constants and also called the *zero-order variables*), logical constants, connectives and operators.
- *Zero-order logics* are just propositional logics.
- *First-order logics* contain, in addition, variables ranging over domain elements (called also *individual variables* and *first-order variables*) and allow quantifiers binding such variables.
- *Second-order logics* contain, in addition, variables ranging over sets of domain elements (called also *second-order variables*) and allow quantifiers binding such variables. Alternatively, second-order variables can range over relations or functions (why?).
- *Third-order logics* contain, in addition, variables ranging over sets of sets of domain elements (called also *third-order variables*) and allow quantifiers binding such variables.
- ...
- *n-th order logics* contain, in addition, variables ranging over sets of elements of order $(n - 1)$ (called also *n-th order variables*) and allow quantifiers binding such variables.
- etc.

What is logic? – continued

Having only a language syntax defined, one knows how to formulate sentences (i.e., how to “speak”) correctly, but does not know what do the sentences mean and what are their consequences.

Examples

“John Watts is a good driver”

– is a correct English sentence, but is it true or false? Or maybe unknown?

To check it we might try to:

- find out to which person “John Watts” refers to (there might be many persons with that name)
- verify what is a meaning of a “good driver”? A professional driver? A person having a driving license?

Then we might be able to semantically verify our task.

Another possibility is to try to find out whether or not our sentence is a consequence of our knowledge, for example, we might know that “John Watts is a Formula I driver”. If, moreover, we accept that

“any Formula I driver is a good driver”,

we might derive that indeed “John Watts is a good driver”.

A *meaning* (also called an *interpretation*) is to be attached to any well-formed formula of the logic. The meaning is given either *semantically* (*semantical approach*) or *syntactically* (*syntactical approach*).

Examples

1. Example of semantical reasoning:

consider the sentence “Humans are rational animals”.
 – Is it true? Maybe false?

No matter whether we agree or not, in our everyday arguments we would usually have to know the meaning of “humans”, “to be rational” and “animals”.

According to the meaning we have in mind we find this sentence valid or not.

2. Example of syntactical reasoning:

consider the following rule:

“ A is/are B ” and “ B does/do D ” therefore “ A does/do D ”

This rule allows us to infer (prove) facts from other facts, if we agree that it is “sound”.

Consider the previous sentence together with “Rational animals think”. To apply our rule, we interpret A, B, C :

- A denotes “humans”
- B denotes “rational animals”
- D denotes “think”

Thus we infer: “Humans think”.

Semantical approach

In the *semantical approach* we attach meaning (“real entities”) to symbols:

- objects to constants
- range of objects to variables
- functions to function symbols
- relations to relation symbols.

The meaning of connectives, operators and auxiliary symbols is fixed by a given logic.

Example

Consider sentence “John is similar to the Jack’s father”.

In a logical form we would write this sentence more or less as follows:

John is similar to *father of Jack*

or, much more often, as: $sim(John, fath(Jack))$, where *sim* and *fath* are suitable abbreviations of *similar to* and *father of*.

In order to verify this sentence we have to know the meaning of:

- constants *John, Jack*
- function denoted by symbol *fath*
- relation denoted by symbol *sim*.

“Humans are rational animals” – continued

Analysis of this sentence is not immediate. We have to know the meaning of “humans”, “rational” and “animals”. We can agree that:

- *humans* – denotes the set of humans
- *animals* – denotes the set of animals
- *rational* – denotes the set of rational creatures.

Do we have a notion of sets in our logic? Maybe. In some logics we do, in some we do not.

Assume we have it. Then our sentence can be expressed as follows:

$$humans = rational \cap animals$$

since we want to say that humans are creatures that are both rational and animals.

Now we interpret “humans”, “rational” and “animals” as concrete sets and see whether the equality holds.

More often we consider relations rather than sets. In this case we would have:

- $human(x)$ – meaning that x is a human
- $rational(x)$ – meaning that x is rational
- $animal(x)$ – meaning that x is an animal.

Our sentence can be formulated as:

$$\text{for all } x, human(x) \text{ iff } rational(x) \text{ and } animal(x).$$

Now we have to interpret “human”, “rational” and “animal” as concrete relations, fix a range for variable x and make reasoning.

Syntactical approach

In the *syntactical approach* we attach meaning to symbols of the language by fixing *axioms* and *proof rules* (rules, in short).

- *Axioms* are facts “obviously true” in a given reality.
- *Rules* allow us to infer new facts on the basis of known facts (axioms, facts derived from axioms, etc.).

Axioms together with proof rules are called *proof systems*.

Example

Consider the following rule (called *modus ponens*):

if A holds
and
whenever A holds, B holds, too (i.e., A implies B)
then infer that B holds.

Assume that we have the following two axioms:

I read a good book.
Whenever I read a good book, I learn something new.

Taking

A to be “I read a good book”,
 B to be “I learn something new”,

we have “ A ” and “ A implies B ”, thus applying modus ponens we infer “ B ”, i.e., “I learn something new”.

Discussion

In the semantical reasoning we also apply rules, maybe more implicitly, so what is the difference?

In the syntactical approach we do not care about the meaning of sentences. We just syntactically transform formulas without referring to any specific meaning of items occurring in formulas.

The meaning is then given by facts that are true or false.

Example

Consider the following rule:

if a person x is a parent of a person y
then x is older than y .

Assume we have the following axioms:

John is a person.
Eve is a person.
Marc is a person.
John is a parent of Marc.
Eve is a parent of Marc.

Applying the considered rule we can infer:

John is older than Marc.
Eve is older than Marc.

This reasoning gives us a (partial) meaning of “to be older than”.

What is logic? – Conclusion

By a *logic* we shall understand any triple $\langle \text{TV}, \mathcal{L}, \Vdash \rangle$, where

- TV is the set of *truth values*, e.g., $\text{TV} = \{\text{TRUE}, \text{FALSE}\}$
- \mathcal{L} is a set of formulas
- \Vdash is a *consequence relation* assigning meaning to all formulas of the logic, $\Vdash : \text{Pow}(\mathcal{L}) \times \mathcal{L} \rightarrow \text{TV}$, i.e., for any set of formulas S and any formula $A \in \mathcal{L}$, the value $S \Vdash A$ is a truth value (e.g., TRUE or FALSE) indicating how A is a consequence of S .

Remarks

1. We do not explicitly consider terms in the above definition. However, the meaning of terms can be provided by formulas of the form *term* = *value*. If such a formula is TRUE then the value of term *term* is *value*.
2. As discussed earlier the consequence relation can be defined syntactically or semantically. However, we do not exclude other possibilities here, although these are rather rare.
3. Truth values can also be UNKNOWN, INCONSISTENT, etc., in some logics even all real values from the interval $[0, 1]$ (e.g., in fuzzy logics)
4. In logic the definition of consequences is not required to be constructive, although automated as well as practical everyday reasoning, is to be based on some constructive machinery.

Example

Let us fix a language for arithmetics of real numbers and define consequence, saying, among others, that formulas of the form “for every real number x property $A(x)$ holds are TRUE iff

for any real number substituting x in $A(x)$, property $A(x)$ holds.

This is highly non-constructive!

In everyday practice one uses more constructive techniques based on manipulating terms and formulas.

For example, in order to prove that for every real number x we have that $x < x + 1$ one does not pick all real numbers one by one and check whether a given number is less than the number increased by 1, like e.g., checking:

$$\begin{aligned} 2.5 &< 2.5 + 1 \\ \sqrt{5} &< \sqrt{5} + 1 \\ 1238 &< 1238 + 1 \\ &\dots \end{aligned}$$

One rather observes that

1. $0 < 1$
2. adding x to both sides of any inequality preserves the inequality and obtains that $x + 0 < x + 1$, i.e. $x < x + 1$.

Some meta-properties

A *meta-property* is a property of logic rather than of the reality the logic describes.

There are two important meta-properties relating syntactical and semantical approaches, namely *soundness* (also called *correctness* and *completeness* of a proof system wrt a given semantics).

Assume a logic is given via its semantics \mathcal{S} and via a proof system \mathcal{P} . Then we say that:

- proof system \mathcal{P} is *sound* (*correct*) wrt the semantics \mathcal{S} iff every property that can be inferred using \mathcal{P} is true under semantics \mathcal{S} ,
- proof system \mathcal{P} is *complete* wrt the semantics \mathcal{S} iff every property that is true under semantics \mathcal{S} can be inferred using \mathcal{P} .

In practical reasoning:

- **soundness is required**
- **completeness is desirable.**

What you should have learnt from Lecture I?

- what is correct argument and correct reasoning?
- what is logic?
- what is a logical language?
- what are the typical “building blocks” for expressions?
- what are function and relation symbols?
- what are terms? what are ground terms?
- what are formulas?
- what is the semantical approach?
- what is the syntactical approach?
- what is a proof system?
- soundness and completeness of proof systems.

What is a database?

A *database* is any finite collection of related data.

Examples

1. a figure illustrating dependencies between shapes on a plane
2. a phone book
3. a Swedish-to-English dictionary
4. a library
5. the Mendelejev periodic table
6. an electronic mailbox
7. a personnel database of a department
8. an accounting database
9. a credit card database
10. a flight reservation database
11. ...

A closer look at databases

Database	Representation in logic
an atomic data item = number, letter, etc.	an element of a domain (universe)
a record = a tuple of atomic data items	<i>a fact</i>
a table = a collection of "compatible" records	a relation
a relational database = a collection of tables + integrity constraints	<i>an extensional database</i> = a collection of relations + formulas
a deductive database = relational database + deduction mechanism	<i>a deductive database</i> extensional database + <i>intensional database</i> (rules)
a database query a linguistic tool for selecting pieces of information	a formula

An *integrity constraint* of a database is a constraint that has to be satisfied by any instance of the database.

Example: a mobile phone book

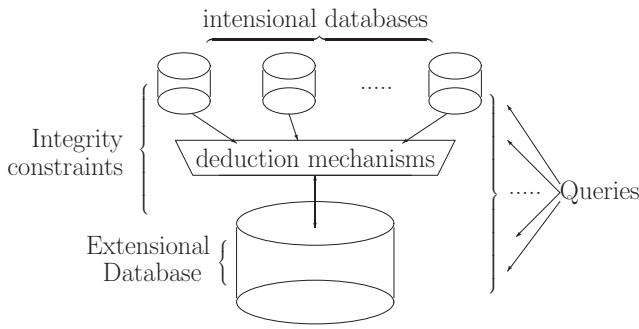
Database	Representation in logic
atomic data: = first&last name, phone number	domains: NAMES, NUMBERS
a record: {Eve Lin, 0701 334567}	a fact: $PN('Eve Lin', 0701 334567)$
a table: {Eve Lin, 0701 334567} {John Smith, 0701 334555} ...	a relation: $PN('Eve Lin', 0701 334567)$ $PN('JohnSmith', 0701 334555)$...
a relational database: the above table + constraints e.g., "different persons have different phone numbers"	the extensional database: the above relation + $\forall x, y \in \text{NAMES} \forall n, m \in \text{NUMBERS}$ $[PN(x, n) \wedge PN(y, m) \wedge x \neq y \rightarrow n \neq m]$
a database query: "find phone numbers of Eve Lin"	formula: $PN('Eve Lin', X)$

What is a database? – revisited

Given a logic, a *database* is a theory of the logic, expressing facts, integrity constraints and rules, and allowing one to ask queries in the language of the logic.

Logic	Database type
The classical first-order logic	Deductive databases (with expressive power of SQL)
Fixpoint calculus	Deductive databases (expressing PTIME computable queries)
Second-order logic	Deductive databases (expressing PSPACE computable queries)
Modal logic	Modal databases
Temporal logics	Temporal databases
Spatial calculus	Spatial databases
Fuzzy logics	Fuzzy databases
Rough calculus	Rough databases

The architecture of deductive databases



Extensional databases, integrity constraints and queries are present in traditional database management systems. Deductive databases offer means to store deductive rules (intensional databases) and provide a deduction machinery. The basic language for deductive databases is *Datalog*.

Common assumptions

- **complexity**: database queries are computable (in a reasonable time)
- if c_1, \dots, c_n are all constant symbols appearing in the database, then it is assumed that
 - *UNA (Unique Names Axiom)*:
for all $1 \leq i \neq j \leq n$ we have that $c_i \neq c_j$
 - *DCA (Domain Closure Axiom)*:
 $\forall x [x = c_1 \vee \dots \vee x = c_n]$
- *CWA (Closed World Assumption)*:
if $R(\bar{a}_1), \dots, R(\bar{a}_k)$ are all facts about R stored in the database then
 $\forall \bar{x} [R(\bar{x}) \rightarrow (\bar{x} = \bar{a}_1 \vee \dots \vee \bar{x} = \bar{a}_k)]$
holds.

Facts in Datalog

Facts in Datalog are represented in the form of relations

$$name(arg_1, \dots, arg_k),$$

where *name* is a name of a relation and arg_1, \dots, arg_k are constants.

Examples

1. $address(John, 'Kungsgatan 12')$
2. $likes(John, Marc)$.

Atomic queries about facts

Atomic queries are of the form $name(arg_1, \dots, arg_k)$, where arg_1, \dots, arg_k are constants or variables, where \bar{c} is also a variable (without a name, "do-not-care" variable).

Examples

1. $likes(John, Marc)$ – does John like Marc?
2. $likes(X, Marc)$ – who likes Marc?
(compute X 's satisfying $likes(X, Marc)$).

Rules in Datalog

Rules in Datalog are expressed in the form of (a syntactic variant of) Horn clauses:

$$R(\bar{Z}) : -R_1(\bar{Z}_1), \dots, R_k(\bar{Z}_k)$$

where $\bar{Z}, \bar{Z}_1, \dots, \bar{Z}_k$ are vectors of variable or constant symbols such that any variable appearing on the lefthand side of $:-$ (called the *head* of the rule) appears also on the righthand side of the rule (called the *body* of the rule).

The intended meaning of the rule is that

$$[R_1(\bar{Z}_1) \wedge \dots \wedge R_k(\bar{Z}_k)] \rightarrow R(\bar{Z}),$$

where all variables that appear both in the rule's head and body are universally quantified, while those appearing only in the rule's body, are existentially quantified.

Example

Rule:

$$R(X, c) : -Q(X, Z), S(Z, X)$$

denotes implication:

$$\forall X \{ \exists Z [Q(X, Z) \wedge S(Z, X)] \rightarrow R(X, c) \}.$$

Examples of Datalog facts and rules

Facts:

```
is_a(zebra,mammal).
is_a(whale,mammal).
is_a(herring,fish).
is_a(shark,fish).
```

```
lives(zebra,on_land).
lives(whale,in_water).
lives(frog,on_land).
lives(frog,in_water).
lives(shark,in_water).
```

Rules:

```
can_swim(X):- is_a(X,fish).
can_swim(X):- lives(X,in_water).
```

Based on the above facts and rules, can one derive that:

1. a whale can swim?
2. a frog can swim?
3. a salmon can swim?

Datalog queries

A *Datalog query* is a conjunction of literals (conjunction is denoted by comma).

Assumptions UNA, DCA and CWA allow one to formally define the meaning of queries as minimal relations which make queries logical consequences of facts and rules stored in the database.

Intuitively,

- if a query does not have free variables, then the answer is either TRUE or FALSE dependently on the database contents
- if a query contains variables, then the result consists of all assignments of values to variables, making the query TRUE.

Examples

1. `is_a(X,mammal)`
returns all X 's that are listed in the database as mammals
2. `is_a(X,mammal), can_swim(X)`
returns all mammals listed in the database that can swim.

Example: authors database

Consider a database containing information about book authors.

It is reasonable to store authors in one relation, say A , and book titles in another relation, say T . The intended meaning of relations is the following:

- $A(N, Id)$ means that the author whose name is N has its identification number Id
- $T(Id, Ttl)$ means that the author whose identification number is Id is a (co-) author of the book entitled Ttl .

Now, for example, one can define a new relation $Co(N_1, N_2)$, meaning that authors N_1 and N_2 co-authored a book:

```
Co(N_1, N_2) :-
  A(N_1, Id_1),
  T(Id_1, B),
  A(N_2, Id_2),
  T(Id_2, B),
  N_1 /= N_2.
```

Note that the above rule reflects the intended meaning provided that authors have unique Id numbers, i.e., that the following integrity constraint holds:

$$\forall n, m, i, j \\ [(A(n, i) \wedge A(n, j)) \rightarrow i = j] \wedge \\ [(A(n, i) \wedge A(m, i)) \rightarrow n = m]$$

Querying the author database

In the authors database:

- $Co('Widom', 'Ullman')$ returns TRUE iff the database contains a book co-authored by **Widom** and **Ullman**
- $Co(N, 'Ullman')$ returns a unary relation consisting of all co-authors of **Ullman**.

In fact, query language can easily be made more general by allowing all first-order formulas to be database queries.

Example

Query

$$\forall X, Y [T(X, Y) \rightarrow \exists Z.(T(X, Z) \wedge Z \neq Y)]$$

returns TRUE iff any person in the database is an author of at least two books.

Exercises

Design a database and formulate sample queries:

1. for storing information about your favorite movies
2. for storing information about bus/train connections.

Complexity of queries

Two ways to measure complexity of queries:

- **data complexity**, i.e., the complexity of evaluating a fixed query for variable database inputs,
- **expression complexity**, i.e., the complexity of evaluating, on a fixed database instance, the various queries specifiable in a given query language.

Data complexity is typically more relevant so we use the term **complexity** to refer to data complexity.

For a query Q we have the following **query recognition problem**:
given a database D and a tuple \bar{u} , determine whether \bar{u} belongs to the answer $Q(D)$.

For each Turing time of space complexity class C , one can define the corresponding **complexity class of queries**, denoted by QC .

The class of queries QC consists of all queries whose recognition problem is in C .

Examples: QP_{TIME}, QP_{SPACE}, QN_{PTIME},...

Data complexity of first-order queries

Data complexity of first-order queries is QP_{TIME} and QLOG_{SPACE}.

Sketch of proof:

Let DOM be the database domain.

Let $Q_1 z_1 \dots Q_r z_r A(z_r, \dots, z_1, x_1, \dots, x_k)$ be a first-order query with quantifier-free A , $Q_1, \dots, Q_r \in \{\forall, \exists\}$ and $z_r, \dots, z_1, x_1, \dots, x_k$ being all variables in A .

Naive algorithm (suffices to show that data complexity of first-order queries is QP_{TIME} — how to show that it is QLOG_{SPACE}?)

1. compute the set R of $(r+k)$ -tuples satisfying $A(z_1, \dots, z_r, x_1, \dots, x_k)$
2. for $i = r, \dots, 1$ do
 - if $Q_i = \forall$ then $R := \{\langle b_1, \dots, b_{k+i-1} \rangle \mid \text{for all } x \in DOM, \langle x, b_1, \dots, b_{k+i-1} \rangle \in R\}$
 - otherwise (if $Q_i = \exists$) $R := \{\langle b_1, \dots, b_{k+i-1} \rangle \mid \text{there is } x \in DOM \text{ such that } \langle x, b_1, \dots, b_{k+i-1} \rangle \in R\}$
3. return R as the result.

Data complexity of Datalog queries

Data complexity of Datalog queries is QP_{TIME} and QP_{SPACE}.

We shall show that using results on fixpoint calculus (Lecture III)

Expression complexity of first-order and Datalog queries

Expression complexity of first-order and Datalog queries is QEXP_{TIME}

What you should have learnt from Lecture II?

- what is a logical (deductive) database?
- what are extensional and intensional databases?
- what are integrity constraints?
- what are UNA, DCA and CWA?
- what is Datalog?
- what are Datalog facts and rules?
- how to query a deductive database?
- what is data complexity and expression complexity?
- what is the expression complexity and data complexity of first-order and Datalog queries?

What are Fixpoints?

Given a mapping $f : \text{Pow}(\text{DOM}) \rightarrow \text{Pow}(\text{DOM})$, where $\text{Pow}(\text{DOM})$ denotes the *powerset* of DOM , i.e., the set of all subsets of DOM

- by a *fixpoint of f* we understand $A \subseteq \text{DOM}$ such that $A = f(A)$, if exists.
- by the *least* and the *greatest fixpoint* of f , if exist, we mean respectively fixpoints $A, B \subseteq \text{DOM}$ such that for all fixpoints C of f , we have that $A \subseteq C$ and $C \subseteq B$. The least fixpoint of f , if exists, is denoted by $\text{LFP } X.f(X)$ and the greatest fixpoint of f , if exists, is denoted by $\text{GFP } X.f(X)$.

Examples

1. Let $f(A) \stackrel{\text{def}}{=} -A$. Then f has no fixpoint — why?
2. Let

$$f(A) \stackrel{\text{def}}{=} \begin{cases} \emptyset & \text{for } A = \emptyset \\ -A & \text{otherwise.} \end{cases}$$

Then the least and the greatest fixpoint of f is \emptyset .

3. Let $f : \text{Pow}(\omega) \rightarrow \text{Pow}(\omega)$ be defined by

$$f(A) \stackrel{\text{def}}{=} \begin{cases} A & \text{for } A \subsetneq \omega \\ \emptyset & \text{for } A = \omega. \end{cases}$$

Then the least fixpoint of f is \emptyset and the greatest fixpoint of f does not exist (why?).

Monotone mappings

Given a set DOM and a mapping $f : \text{Pow}(\text{DOM}) \rightarrow \text{Pow}(\text{DOM})$, we say that f is *monotone* provided that for any $A, B \subseteq \text{DOM}$, $A \subseteq B$ implies $f(A) \subseteq f(B)$.

Examples

1. $f(A) \stackrel{\text{def}}{=} A$ is monotone
2. $f(A) \stackrel{\text{def}}{=} B \cap A$ is monotone
3. let $A \subseteq \omega$. Then $f(A) \stackrel{\text{def}}{=} A \cup \{\min(\omega - A)\}$ is monotone
4. none of mappings considered in examples given in slide 37 is monotone — why?
5. recall that every classical first-order formula represents a set of tuples satisfying the formula (elements of tuples correspond to free variables). We then have:

- $f(X(x, y)) \stackrel{\text{def}}{=} \{x = y \vee \exists z [R(x, z) \wedge X(z, y)]\}$
— is monotone
- $f(X(x, y)) \stackrel{\text{def}}{=} \{x = y \vee \exists z [R(x, z) \wedge \neg X(z, y)]\}$
— is not monotone

Knaster-Tarski Theorem

Theorem [Knaster-Tarski]

Let $f : \text{Pow}(\text{DOM}) \rightarrow \text{Pow}(\text{DOM})$ be a monotone mapping. Then there exist the least and the greatest fixpoint of f .

The case of finite domains

Let DOM be a finite domain and let f be monotone.

We first prove that $\bigcup_{i \in \omega} f^i(\emptyset)$ is a fixpoint.

Under our assumptions:

- $\emptyset \subseteq f(\emptyset)$ — since \emptyset is the least element of $\text{Pow}(\text{DOM})$
- $f(\emptyset) \subseteq f(f(\emptyset))$ — by monotonicity of f
- ...
- $f^k(\emptyset) \subseteq f^{k+1}(\emptyset)$ for any $k \in \omega$.

DOM is finite. For any $k \in \omega$, either $f^k(\emptyset) = f^{k+1}(\emptyset)$ or $f^{k+1}(\emptyset)$ contains at least one element which is not in $f^k(\emptyset)$. By finiteness of DOM , this can happen at most a finite number of times. Therefore $\bigcup_{i \in \omega} f^i(\emptyset)$ is a fixpoint of f .

Let us prove that $\bigcup_{i \in \omega} f^i(\emptyset)$ is the least fixpoint.

Let B be another fixpoint of f , i.e., $f(B) = B$. Now:

- $\emptyset \subseteq B$
- $f(\emptyset) \subseteq f(B) = B$ — by monotonicity of f
- ...
- $f^k(\emptyset) \subseteq f^k(B) = B$ for any $k \in \omega$.

In conclusion, $\bigcup\{f^k(\emptyset) \mid k \in \omega\} \subseteq B$. □

Proof of Knaster-Tarski Theorem – the general case

Let us prove the theorem for the case of the least fixpoint of f , i.e., $\text{LFP } X.f(X)$. Define the sequence A^α , where α is an ordinal:

$$A^\alpha = \begin{cases} \emptyset & \text{when } \alpha = 0 \\ f(A^\beta) & \text{when } \alpha = \beta + 1 \text{ is a successor ordinal} \\ \bigcup_{\xi < \alpha} A^\xi & \text{when } \alpha \text{ is a limit ordinal.} \end{cases}$$

Observe that $A^\alpha \subseteq A^{\alpha+1}$. Therefore this sequence reaches a fixpoint. Define $\text{LFP } X.f(X)$ to be A^α for the least ordinal α such that $A^\alpha = A^{\alpha+1}$ (such a α is called the *closure ordinal* for f).

Of course, in this case, $f(A^\alpha) = A^{\alpha+1} = A^\alpha$, i.e., A^α is indeed a fixpoint of f .

Suppose that B is a fixpoint of f . By transfinite induction we show that for any ordinal α we have that $A^\alpha \subseteq B$:

1. $A^0 \stackrel{\text{def}}{=} \emptyset \subseteq B$
2. for any α , if $A^\alpha \subseteq B$, then by monotonicity of f , $f(A^\alpha) \subseteq f(B) = B$, i.e., $A^{\alpha+1} \subseteq B$
3. suppose that α is a limit ordinal, $\alpha = \bigcup_{\xi < \alpha} \xi$ and suppose that for all $\xi < \alpha$, $A^\xi \subseteq B$. Then $\alpha = \bigcup_{\xi < \alpha} A^\xi \subseteq B$. □

Fixpoints of functions defined by means of first-order formulas

Set-theoretical concept	Representation in logic
a set A	characteristic formula $A(x)$ $A(x) = \text{TRUE}$ iff $x \in A$
the empty set \emptyset	FALSE
the whole domain DOM	TRUE
inclusion \subseteq	implication \rightarrow
equality $=$	equivalence \equiv
complement $-$	negation \neg
intersection \cap	conjunction \wedge
union \cup	disjunction \vee

Examples

- Let $f(P) \stackrel{\text{def}}{=} (P \cup Q)$. Then the corresponding definition in the classical first-order logic is $f(P(x)) \stackrel{\text{def}}{=} (P(x) \vee Q(x))$. Then the least and the greatest fixpoints of f are:
 - the least fixpoint: $f(\text{FALSE}) \equiv (\text{FALSE} \vee Q(x)) \equiv Q(x)$,
 $f^2(\text{FALSE}) \equiv f(f(\text{FALSE})) \equiv f(Q(x)) \equiv (Q(x) \vee Q(x)) \equiv Q(x)$ (i.e., $Q(x)$ is the least fixpoint)
 - the greatest fixpoint: $f(\text{TRUE}) \equiv (\text{TRUE} \vee Q(x)) \equiv \text{TRUE}$, (i.e., TRUE is the greatest fixpoint).
- Let $f(P(x)) \stackrel{\text{def}}{=} R(x, x) \vee \exists y [R(x, y) \wedge P(y)]$ be a mapping over first formulas ordered by implication — what are the least and the greatest fixpoints of f ?

The duality of the least and greatest fixpoints

Least and greatest fixpoints are dual to each other. For every function $f : \text{POW}(\text{DOM}) \rightarrow \text{POW}(\text{DOM})$ we define the *dual fixpoint operator* $f^d(X) = -f(-X)$. If f is monotone then f^d is also monotone and

$$\text{LFP } X.f(X) = -\text{GFP } X.f^d(X)$$

$$\text{GFP } X.f(X) = -\text{LFP } X.f^d(X).$$

Examples

- Let $f(X(x)) \stackrel{\text{def}}{=} R(x, x) \vee \exists y [R(x, y) \wedge X(y)]$. Then

$$f^d(X(x)) \stackrel{\text{def}}{=} \neg [R(x, x) \vee \exists y [R(x, y) \wedge \neg X(y)]]$$

$$\equiv \neg R(x, x) \wedge \forall y [\neg R(x, y) \vee X(y)]$$
- Let $f(X) \stackrel{\text{def}}{=} R \vee X$. Then $f^d(X) \stackrel{\text{def}}{=} \neg f(\neg X) = \neg R \wedge X$. Now:
 - $\text{LFP } X.f^d(X) = \text{FALSE}$
 - $\text{GFP } X.f^d(X) = \neg R$
 - $\text{LFP } X.f(X) = R = \neg \text{GFP } X.f^d(X)$
 - $\text{GFP } X.f(X) = \text{TRUE} = \neg \text{LFP } X.f^d(X)$.

Positive and monotone formulas

- A *literal* is either an atomic formula (then it is also called the *positive literal*) or its negation (then it is also called the *negative literal*).
- A classical first-order formula is in the *negation normal form*, if the negation sign \neg appears in literals only (if at all) and it contains no other connectives than \wedge, \vee, \neg .
- A formula A is *positive wrt a given atomic formula S* iff S appears in A only in the form of a positive literal.
- A formula A is *negative wrt a given atomic formula S* iff S appears in A only in the form of a negative literal.
- A formula $A(S)$ is *monotone wrt an atomic formula S* iff for any formulas B, C , if $B \rightarrow C$ is true then $A(B) \rightarrow A(C)$ is true, too.

Fact

Any formula of the classical propositional or first-order logic positive wrt a given atomic formula S is also monotone wrt S .

Proof

Induction on the structure of formulas (to be shown during the lecture). □

Examples

- p and $R(x, y, z)$ are positive literals
- $\neg p$ and $\neg R(x, y, z)$ are negative literals
- $P(x) \vee R(x, y, z)$ and $\neg R(x, y, z)$ are not literals
- $P(x) \vee \neg R(x, y, z) \wedge \neg S(x, y)$ is in the negation normal form
- $P(x) \vee \neg [R(x, y, z) \wedge S(x, y)]$ is not in the negation normal form
- $P(x) \vee \neg R(x, y, z) \wedge \neg S(x, y)$ is positive wrt P and negative wrt R and S
- $P(x) \vee [R(x, y, z) \wedge \neg P(y)]$ is neither positive wrt P nor negative wrt P
- $P(x) \wedge \neg P(x)$ is monotone wrt P , but not positive wrt P .

Transforming formulas into the negation normal form

Replace subformulas according to the table below, until the negation normal form is obtained.

Rule	Subformula	Replaced by
1	$A \equiv B$	$(\neg A \vee B) \wedge (A \vee \neg B)$
2	$A \rightarrow B$	$\neg A \vee B$
3	$\neg \neg A$	A
4	$\neg (A \vee B)$	$\neg A \wedge \neg B$
5	$\neg (A \wedge B)$	$\neg A \vee \neg B$
6	$\neg \forall x A$	$\exists x \neg A$
7	$\neg \exists x A$	$\forall x \neg A$

Fixpoint calculus (fixpoint logic)

By the *monotone fixpoint logic* we mean an extension of the classical first-order logic, obtained by allowing also fixpoint formulas of the form

$$\begin{aligned} \text{LFP } X(\bar{x}).A(X) \\ \text{GFP } X(\bar{x}).A(X), \end{aligned}$$

where $A(X)$ is a monotone first-order or fixpoint formula.

Testing first-order formulas for monotonicity is undecidable. Therefore much more frequent fixpoint logic applied in practice is the positive fixpoint logic, as defined below.

By the *positive fixpoint logic* (or *fixpoint logic*, for short) we mean an extension of the classical first-order logic, obtained by allowing also fixpoint formulas of the form

$$\begin{aligned} \text{LFP } X(\bar{x}).A(X) \\ \text{GFP } X(\bar{x}).A(X), \end{aligned}$$

where $A(X)$ is a positive first-order or fixpoint formula.

Semantics of fixpoint calculus

The semantics of $\text{LFP } X.f(X)$ and $\text{GFP } X.f(X)$ is the least and the greatest fixpoint of $f(X)$, i.e. the least and the greatest relation X such that $X \equiv f(X)$. Since f is assumed monotone or positive w.r.t. X , such fixpoints exist. More precise definition follows.

Given a relational structure $\langle \text{DOM}, \{f_i^{\text{DOM}} : i \in I\}, \{R_j^{\text{DOM}} : j \in J\} \rangle$, and any valuation $v : V_1 \rightarrow \text{DOM}$ can be extended to valuation assigning boolean values to fixpoint formulas as follows, assuming that first-order connectives and quantifiers are defined as in the case of predicate calculus:

$$\begin{aligned} v(\text{LFP } X(\bar{x}).A(X)) &= \text{the least (w.r.t. } \subseteq) \\ &\quad \text{relation } S \text{ such that} \\ &\quad S(x) \equiv v(A(S)) \\ v(\text{GFP } X(\bar{x}).A(X)) &= \text{the greatest (w.r.t. } \subseteq) \\ &\quad \text{relation } S \text{ such that} \\ &\quad S(x) \equiv v(A(S)). \end{aligned}$$

Examples

1. The transitive closure of a binary relation R can be defined by the following fixpoint formula:
 $\text{TC}(R)(x, y) \equiv \text{LFP } X(x, y).[R(x, y) \vee \exists z.(R(x, z) \wedge X(z, y))].$
2. Assume we are given a unary relation $Wise$ and a binary relation $Colleague$ defined on the set $Persons$ and suppose we want to calculate the relation $Wisest$ as the greatest relation satisfying the following constraint, meaning that $Wisest$ are those who are wise and have only wisest colleagues:

$$\forall x.Wisest(x) \rightarrow (Wise(x) \wedge \forall y.(Colleague(x, y) \rightarrow Wisest(y))).$$

The $Wisest$ relation is defined by the fixpoint formula $\text{GFP } X(x).[Wise(x) \wedge \forall y.(Colleague(x, y) \rightarrow X(y))]$, since we are interested in calculating the greatest such relation.

3. Consider a game with states a, b, \dots . The game is between two players. The possible moves of the games are held in a binary relation M . A tuple $\langle a, b \rangle$ in M indicates that when in a state a , one can chose to move to state b .

A player loses if he or she is in a state from which there are no moves. The goal is to compute the set of winning states (i.e., the set of states such that there exists a winning strategy for a player in this state). These are defined by a unary predicate W .

Then winning states are then defined by:

$$W(x) \equiv \text{LFP } X(x).\{\exists y.[M(x, y) \wedge \forall z.(M(y, z) \rightarrow X(z))]\}.$$

4. Assume we are given relations

$$\text{bus}(x, y) \text{ and } \text{train}(x, y)$$

meaning that there is a direct connection by bus, or respectively by train between places x and y .

Using a fixpoint formula, define relation

$$\text{connected}(x, y),$$

meaning that places x and y are directly or indirectly connected (possibly mixing bus and train connections).

5. Assume we are given a relation $R(x, y)$ meaning that a newspaper article x is related to article y .

Define relation $\text{related}(x, y)$ meaning that article x is directly or indirectly related to article y

(for example, if a given article, say

“Famous Forests in Europe” (1),

is related to article

“Ecology in EU” (2),

and “Ecology in EU” is related to

“Natural Resources” (3),

then (1) is directly related to (2), (2) is directly related to (3) and

(1) is indirectly related to (3). If (3) is related to some article (4)

then (1) is indirectly related to (4), etc.).

Complexity of fixpoint calculus

1. Checking whether a fixpoint formula is satisfiable or whether it is a tautology are not partially computable problems.
2. Given a fixpoint formula, checking its satisfiability or validity over a given finite domain relational structure is in PTIME.

Proof of 2 (sketch)

In order to see that satisfiability or validity over a given finite domain relational structure is in PTIME, assume that a given fixpoint formula defines a k -argument relation

$$R(x_1, \dots, x_k) \equiv \text{LFP } X(x_1, \dots, x_k).f(X, x_1, \dots, x_k)$$

and that the domain contains n elements. It follows that there are at most n^k tuples in R .

As indicated in slide 39, $R(x_1, \dots, x_k) \equiv \bigvee_{i \in \omega} f^i(\text{FALSE}, x_1, \dots, x_k)$,

where the sequence $\langle f^i(\text{FALSE}, x_1, \dots, x_k) \mid i \in \omega \rangle$ is non-decreasing.

It now suffices to note that the number of iterations to compute fixpoint is not greater than n^k (each iteration either stabilizes at the least fixpoint, or adds at least one tuple). Each iteration (by inductive argument) requires at most a polynomial time to calculate the result. \square

Data complexity of Datalog queries

Application: data complexity of Datalog queries is QPTIME and QPSpace.

Assume that the following Datalog rules are all rules with R in their heads, where we assume that \bar{Z} consists of variables only (this restriction is not substantial — why?):

$$R(\bar{Z}): -A_1(R, \bar{Z}_1)$$

...

$$R(\bar{Z}): -A_p(R, \bar{Z}_p)$$

The rules can equivalently be expressed by a single formula:

$$\exists \bar{Y} [A_1(R, \bar{Z}_1) \vee \dots \vee A_p(R, \bar{Z}_p)] \rightarrow R(\bar{Z}),$$

where \bar{Y} consists of all variables appearing in $\bar{Z}_1, \dots, \bar{Z}_p$ and not appearing in \bar{Z} .

Now R can be defined by the following fixpoint:

$$R(\bar{Z}) \equiv \text{LFP } X(\bar{Z}). \exists \bar{Y} [A_1(X, \bar{Z}_1) \vee \dots \vee A_p(X, \bar{Z}_p)].$$

The above fixpoint provides us with a PTIME computational machinery for computing the results of Datalog queries.

Let us emphasize that the full semantics of Datalog is best presented by the use of simultaneous fixpoints, but this subject is beyond the scope of this lecture (but see the literature).

What you should have learnt from Lecture III?

- What are fixpoints?
- Knaster-Tarski fixpoint theorem.
- Characterization of fixpoints via increasing chains.
- Duality between least and greatest fixpoint.
- What are monotone and positive formulas?
- How to transform classical formulas into the negation normal form?
- What is monotone and positive fixpoint logic (fixpoint calculus)?
- What is the complexity of fixpoint calculus?
- What is the complexity of fixpoint calculus over finite models?
- Applications of fixpoint calculus in databases, in particular in computing Datalog queries.

Extensional versus intensional relations

By an *extensional relation* we mean any relation which allows us to replace its arguments by equal elements. *Intensional* relations are those that are not extensional.

To test whether a relation, say $R(x_1, x_2, \dots, x_k)$, is extensional or intensional consider its arguments x_1, x_2, \dots, x_k , one by one.

The test for the i -th argument, x_i , where $1 \leq i \leq k$, runs as follows:

– if $x_i = e$ implies that

$$R(x_1, \dots, x_i, \dots, x_k) \text{ and } R(x_1, \dots, e, \dots, x_k)$$

are equivalent, then the test is passed.

If for all arguments the test is passed then the relation is extensional. Otherwise (if there is at least one argument where the test is not passed) the relation is intensional.

Examples of extensional relations

1. All relations in traditional mathematics are extensional, e.g., \leq , \geq , “to be parallel”, to be perpendicular, etc.
2. many relations in natural language are extensional, e.g., “likes”, “has”, “betterThan”, etc.. For example, *likes* is extensional (why?)

Examples of intensional relations

1. Consider sentence “Electra knows that Orestes is her brother. The sentence contains relation “P knows Q” with meaning “P knows that Q is her brother”. It appears to be an intensional relation, as the following paradox of Orestes and Electra shows (the paradox is attributed to to Eubulides), where *MM* is a man in a mask, so that Electra is unable to recognize whether *MM* is or not is *Orestes*:

$$\text{knows}(\text{Electra}, \text{Orestes}) \wedge \text{Orestes} = \text{MM} \\ \vdash \text{knows}(\text{Electra}, \text{MM})$$

so we conclude that Electra knows the masked man, contrary to our assumption that she cannot recognize him. The above reasoning is invalid. In consequence “knows” is not extensional.

2. The following relations are intensional: “P believes in Q”, “it is necessary that P holds”, “P is obligatory”, “P is allowed” — why?
3. Give another examples of intensional relations.

Propositional modal logics

Intensional relations give rise to so-called *modal operators* (*modalities*, in short). Logics allowing such operators are called *modal logics* and sometimes *intensional logics*.

In the simplest case one deals with a single one-argument modality \Box and its dual \Diamond , i.e.,

$$\Diamond A \stackrel{\text{def}}{=} \neg \Box \neg A$$

Example

Suppose that the intended meaning of $\Box A$ is “A is sure”. Then its dual, $\Diamond A$, is defined to be:

$$\begin{aligned} \Diamond A &\stackrel{\text{def}}{=} \neg \Box \neg A \\ &\equiv \neg \text{“}\neg A \text{ is sure”} \\ &\equiv \text{“}\neg A \text{ is not sure”} \\ &\equiv \text{“}\neg A \text{ is doubtful”} \\ &\equiv \text{“}A \text{ is doubtful”} \end{aligned}$$

Readings of modalities

Modalities \Box and \Diamond have many possible readings, dependent on a particular application. The following table summarizes the most frequent readings of modalities.

a possible reading of $\Box A$	a possible reading of $\Diamond A$
A is necessary	A is possible
A is obligatory	A is allowed
always A	sometimes A
in the next time moment A	in the next time moment A
A is known	A is believed
all program states satisfy A	some program state satisfies A
A is provable	A is not provable

Types of modal logics

The types of modal logics reflect the possible readings of modalities. The following table summarizes types of logics corresponding to the possible readings of modalities.

Reading of modalities	Type of modal logic
necessary, possible	<i>alethic logics</i>
obligatory, allowed	<i>deontic logics</i>
always, sometimes, in the next time	<i>temporal logics</i>
known, believed	<i>epistemic logics</i>
all (some) states of a program satisfy (un)provable	<i>logics of programs</i> <i>provability logics</i>

How to chose/design a logic for a particular application

Recall (see Lecture I), that there are two approaches to defining logics, the semantical and syntactical approach. In consequence there are two approaches to choosing/defining modal logics suitable for a given application:

1. syntactical approach, which depends on providing a set of axioms describing the desired properties of modal operators
2. semantical approach, which starts with suitable models and then develops syntactic characterization of modalities.

The applications might, e.g., be:

- temporal reasoning
- understanding (fragments of) natural language
- commonsense reasoning
- specification and verification of software
- understanding law and legal reasoning
- etc.

Example: syntactical approach

Let us first focus on the first approach. Suppose we are interested in formalizing knowledge operator. One can postulate, or instance, the following properties:

1. $\Box A \rightarrow \Diamond A$
— if A is known then A is believed
2. $\Box A \rightarrow A$
— if A is known then it actually holds
3. $A \rightarrow \Diamond A$
— if A holds then it is believed.

Of course there are many important questions at this point, including the following:

- are the properties consistent?
- do the properties express all the desired phenomena?
- is a property a consequence of another property?

Usually answers to such questions are given by semantic investigations.

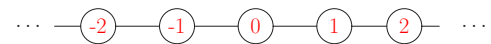
Example: semantical approach

Consider temporal logic, where $\Box A$ means “always A ” and $\Diamond A$ means “sometimes A ”.

In order to define meaning of \Box and \Diamond we first have to decide what is the time structure, e.g.,

- consisting of points or intervals?
- continuous or discrete?
- linear or branching?
- is there the earliest moment (starting point) or there is an unbounded past?
- is there the latest moment (end point) or there is an unbounded future?

Assume that time is discrete and linear, with time points labelled by integers, as in th figure below.



Now “always A ” and “sometimes A ” can be more or less be defined as follows:

- “always A ” $\stackrel{\text{def}}{=} \forall i [A \text{ is satisfied in time point } i]$
 “sometimes A ” $\stackrel{\text{def}}{=} \exists i [A \text{ is satisfied in time point } i]$

Lemmon’s classification of modal logics

Lemmon introduced a widely accepted classification of modal logics, defined below.

The basis for Lemmon’s classification is the logic **K**. **K** is defined to extend the classical propositional logic by adding rules:

1. rule (K):
 $\Box(A \rightarrow B) \vdash \Box A \rightarrow \Box B$
2. rule (Gen):
 $A \vdash \Box A$

Other normal modal logics are defined by additional axioms expressing the desired properties of modalities.

Bellow we present some of the most important axioms.

- | | |
|------------|--|
| D | $\stackrel{\text{def}}{=} \Box A \rightarrow \Diamond A$ |
| T | $\stackrel{\text{def}}{=} \Box A \rightarrow A$ |
| 4 | $\stackrel{\text{def}}{=} \Box A \rightarrow \Box \Box A$ |
| E | $\stackrel{\text{def}}{=} \Diamond A \rightarrow \Box \Diamond A$ |
| B | $\stackrel{\text{def}}{=} A \rightarrow \Box \Diamond A$ |
| Tr | $\stackrel{\text{def}}{=} \Box A \equiv A$ |
| M | $\stackrel{\text{def}}{=} \Box \Diamond A \rightarrow \Diamond \Box A$ |
| G | $\stackrel{\text{def}}{=} \Diamond \Box A \rightarrow \Box \Diamond A$ |
| H | $\stackrel{\text{def}}{=} (\Diamond A \wedge \Diamond B) \rightarrow [\Diamond(A \wedge B) \vee \Diamond(A \wedge \Diamond B) \vee \Diamond(B \wedge \Diamond A)]$ |
| Grz | $\stackrel{\text{def}}{=} \Box(\Box(A \rightarrow \Box A) \rightarrow A) \rightarrow A$ |
| Dum | $\stackrel{\text{def}}{=} \Box(\Box(A \rightarrow \Box A) \rightarrow A) \rightarrow (\Diamond \Box A \rightarrow A)$ |
| W | $\stackrel{\text{def}}{=} \Box(\Box A \rightarrow A) \rightarrow \Box A.$ |

In the formulas given above:

- **D** comes from *deontic*
- **T** is a traditional name of the axiom (after Feys)
- **4** is characteristic for logic **S4** of Lewis
- **E** comes from *Euclidean* (this axiom is often denoted by **5**)
- **B** comes after *Brouwer*
- **Tr** abbreviates *trivial*
- **M** comes after *McKinsey*
- **G** comes after *Geach*
- **H** comes after *Hintikka*
- **Grz** comes after *Grzegorzcyk*
- **Dum** comes after *Dummett*
- **W** comes from *reverse well founded* (it is also known as the Löb axiom).

In Lemmon's classification $\mathbf{KX}_0\dots\mathbf{X}_m$ denotes the logic extending **K** in which formulas $\mathbf{X}_0, \dots, \mathbf{X}_m$ are accepted as axioms.

The following logics are frequently used and applied.

- KT** = **T** = logic of Gödel/Feys/Von Wright
- KT4** = **S4**
- KT4B** = **KT4E** = **S5**
- K4E** = **K45**
- KD** = deontic **T**
- KD4** = deontic **S4**
- KD4B** = deontic **S5**
- KTB** = Brouwer logic
- KT4M** = **S4.1**
- KT4G** = **S4.2**
- KT4H** = **S4.3**
- KT4Dum** = **D** = Prior logic
- KT4Grz** = **KGrz** = Grzegorzcyk logic
- K4W** = **KW** = Löb logic
- KTr** = **KT4BM** = trivial logic.

Kripke semantics for modal logics

There are many semantics for modal logics. Here we shall follow the Kripke-like style of defining semantics for modal logics (in fact, similar semantics was a couple of years earlier defined by Kanger. Then, in the same year, the similar semantics was given by Hintikka and also Guillaume).

The basis for the Kripke semantics is the universe of *possible worlds*. Formulas are evaluated in the so-called *actual worlds*. The other worlds, alternatives to the actual world correspond to possible situations.

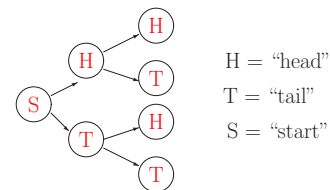
Modalities \Box and \Diamond have the following intuitive meaning:

- formula $\Box A$ holds in a given world w , if A holds in all worlds alternative for w
- formula $\Diamond A$ holds in a given world w , if A holds in some world alternative for w .

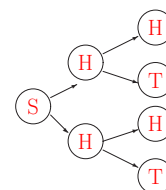


Example

Consider the situation where one drops a coin twice. The following Kripke structure describes all possible situations.



1. does $\Diamond T$ hold?
2. does $\Box H$ hold?
3. does $\Diamond\Diamond H$ hold?



What are answers to the same questions for the above structure?

Formal definition of Kripke semantics

- By a *Kripke frame* we mean any relational system of the form $\langle \mathcal{W}, R \rangle$, where \mathcal{W} is any set and R is a binary relation defined on \mathcal{W} , called an *accessibility relation*. Elements of \mathcal{W} are called *worlds*.
- By a *Kripke structure* we mean triple $\langle \mathcal{K}, w, v \rangle$, where
 - $\mathcal{K} = \langle \mathcal{W}, R \rangle$ is a Kripke frame
 - $w \in \mathcal{W}$ is the *actual world*
 - v is a mapping, $v : \mathcal{F} \times \mathcal{W} \rightarrow \{\text{TRUE}, \text{FALSE}\}$, assigning truth values to propositional variables in worlds
- the semantics of the classical connectives is as usual, where the valuation of propositional variables is given by the actual world of a given Kripke structure
- $\mathcal{K}, w, v \models_{\mathcal{M}} \Box \alpha$ iff for any w' such that $R(w, w')$ holds, we have that $\mathcal{K}, w', v \models_{\mathcal{M}} \alpha$, where R is the relation of the Kripke frame \mathcal{K} (i.e., $\forall w'[R(w, w') \rightarrow \alpha(w')]$)
- $\mathcal{K}, w, v \models_{\mathcal{M}} \Diamond \alpha$ iff there is w' such that $R(w, w')$ and $\mathcal{K}, w', v \models_{\mathcal{M}} \alpha$, where R is the relation of the Kripke frame \mathcal{K} (i.e., $\exists w'[R(w, w') \wedge \alpha(w')]$).

Correspondences in Kripke semantics

Given a modal axiom the question is what Kripke frames validate the axiom. If the property is expressed in the classical first-order logic then we have a *correspondence* between modal axiom and the first-order property.

Examples of correspondences

Formula	Property of R
D	$\forall x. \exists y. R(x, y)$ (R is serial)
T	$\forall x. R(x, x)$ (R is reflexive)
4	$\forall x, y, z. (R(x, y) \wedge R(y, z)) \rightarrow R(x, z)$ (R is transitive)
E	$\forall x, y, z. (R(x, y) \wedge R(x, z)) \rightarrow R(y, z)$ (R is Euclidean)
B	$\forall x, y. (R(x, y) \rightarrow R(y, x))$ (R is symmetric)
Tr	$\forall x, y. (R(x, y) \equiv x = y)$ (R is trivial)
G	$\forall x, y, z. ((R(x, y) \wedge R(x, z)) \rightarrow \exists w (R(y, w) \wedge R(z, w)))$ (R is directed)
$\Diamond \alpha \rightarrow \Box \alpha$	$\forall x, y, z. (R(x, y) \wedge R(x, z)) \rightarrow y = z$ (R is a partial function)
$\Diamond \alpha \equiv \Box \alpha$	$\forall x. \exists! y. R(x, y)$ (R is a function)
$\Box \Box \alpha \rightarrow \Box \alpha$	$\forall x, y. R(x, y) \rightarrow \exists z (R(x, z) \wedge R(z, y))$ (R is dense)

Exercises

1. Design a logic for formalizing the reasoning of perception, where $\Box A$ is intended to mean that “an event satisfying formula A is being observed”.
2. Propose a deontic logic (for “obligatory” – “allowed” modalities)
3. Characterize a commonsense implication.

Multimodal logics

In many applications one needs to introduce many modalities. In such a case we have *multimodal logics*.

Examples

What modalities appear in the following sentences:

1. “When it rains one ought to take umbrella”.
2. “It is necessary that anyone should be allowed to have right to vote”.
3. “If you know what I believe to, then you might be wiser than myself”.

What you should have learnt from Lecture IV?

- what are extensional and intensional relations?
- what is modal operator (modality)?
- what are possible readings of modalities?
- what types of modal logics are frequently considered?
- what is Lemmon’s classification of modal logics?
- what are correspondences between modal and classical logic?
- what are multi-modal logics?

Quantified modal logics

Quantified modal logics are logics with the language of the classical first-order logic extended by adding modalities \Box, \Diamond , possibly with indices (in the case of multi-modal logics).

Examples

1. “Everybody is obliged to pay taxes” translates into

$$\forall x [\Box \text{paysTaxes}(x)],$$

where \Box is interpreted as modality “is obliged”

2. “We believe that no object can move with speed greater than the speed of light” translates into

$$\Diamond [\neg \exists x (\text{speed}(x) > \text{speed}(\text{light}))],$$

where \Diamond is interpreted as modality “we believe”

3. “For everybody it is known that (s)he is always obliged to obey law” translates into

$$\forall x [\Box_k \Box_a \Box_o \text{obeysLaw}(x)],$$

where \Box_k, \Box_a, \Box_o are interpreted as modalities “is known”, “always” and “is obliged”, respectively.

Free logics

In the classical logic any constant symbol has to be assigned a domain value. In natural language this does not have to be the case (see the discussion in page 6).

Examples

- “Every Pegasus has wings” is usually considered to be TRUE, while “Every Pegasus is pink” is considered FALSE, even if Pegasus does not denote any object from our universe.
- In the classical first-order logic formula $\exists x a = x$ is a tautology, where a is a constant symbol.
Whenever we have a constant symbol in our language, we can conclude that the symbol denotes some existing object. Classical logicians then accept that anything named exists. For example, $\exists x \text{Pegasus} = x, \exists x \text{SnowWhite} = x$, etc., are tautologies of the classical logic.
- In modal logics we might deal with object that exist in one world and do not exist in another world, e.g., raindrops can exist in the current world and can disappear in some future world.

By the *minimal free logic*, denoted by MFL, we understand logic with language that of the classical first-order logic with unary relation symbol E with meaning (“exists”), obtained by accepting the axioms of the classical propositional connectives plus identity axioms and rules:

$$\forall x A(x) \vdash E(t) \rightarrow A(t) \quad (1)$$

$$A(x) \rightarrow [E(t) \rightarrow A(t)] \vdash A(x) \rightarrow \forall y A(y), \quad (2)$$

where t is a term that does not appear in $A(x) \rightarrow \forall y A(y)$.

Example

Formula $\forall x [\text{Horse}(x) \vee \neg \text{Horse}(x)]$ is a tautology.

Using rule (1) we can infer, e.g.,

$$E(\text{Pegasus}) \rightarrow [\text{Horse}(\text{Pegasus}) \vee \neg \text{Horse}(\text{Pegasus})].$$

Inferring

$$\text{Horse}(\text{Pegasus}) \vee \neg \text{Horse}(\text{Pegasus}),$$

as could be done in the classical logic (assuming that a constant symbol *Pegasus* is in the language), would be doubtful.

Models for quantified modal logics

- Objectual models for quantified modal logics* are of the form $\langle \mathcal{K}, D, Q, a \rangle$, where:
 - \mathcal{K} is a Kripke frame
 - D is a non-empty set of possible objects
 - Q is the domain of quantification; in the case when quantifier rules are based on free logic we require that predicate E of free logic is interpreted as Q
 - a interprets terms and formulas w.r.t. \mathcal{K} and D .
- Conceptual models for quantified modal logics* are of the form $\langle \mathcal{K}, D, QC, a \rangle$, where:
 - \mathcal{K} is a Kripke frame
 - D is a non-empty set of possible objects
 - QC is the set of functions from W into D , representing the domains of quantification
 - a interprets terms and formulas w.r.t. \mathcal{K} and D .

The objectual interpretation: rigid terms

By *rigid terms semantics* we understand the semantics, where the interpretation of terms is the same in all worlds of Kripke frames. In consequence, we accept rules:

$$t = t' \vdash \Box t = t' \tag{3}$$

$$t \neq t' \vdash \Box t \neq t' \tag{4}$$

Remarks

1. Rigid terms semantics fixes the meaning of terms. However, modal operators can still be intensional w.r.t. formulas, e.g., relation $king(p, c, t)$ meaning that person p is a king of country c in time t can still be world-dependent, even if terms do not change their values.
2. In many applications the rigid terms semantics is oversimplified. For example, let $\Box A$ mean that "I wonder whether A holds". Consider the reasoning:

$$John = John \vdash \Box John = John,$$

i.e., I wonder whether $John$ is $John$.

The objectual interpretation: logic Q1 (fixed domains)

In Q1 we assume that in models $\langle \mathcal{K}, D, Q, a \rangle$ (recall that $\mathcal{K} = \langle W, R \rangle$), the domain of quantification Q is equal to the set of possible objects D and a satisfies the condition:

$$a(t)(w) = a(t)(w'), \text{ for any } w, w' \in W \tag{5}$$

Important:

Let S be a propositional modal logic sound and complete w.r.t. a class C of Kripke frames. Then the system Q1-S is sound and complete w.r.t. class C , where Q1-S is obtained from S by adding the principles of the first-order logic, rules (3), (4) and:

$$t = t \tag{6}$$

$$t = t' \vdash P(t) \rightarrow P(t') \text{ where } P \text{ is an atom} \tag{7}$$

$$\forall x \Box A(x) \rightarrow \Box \forall x A(x) \text{ (Barcan formula)} \tag{8}$$

The meaning of the Barcan formula and its converse

The *converse of the Barcan formula* is

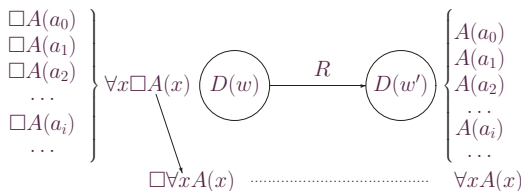
$$\Box \forall x A(x) \rightarrow \forall x \Box A(x). \tag{9}$$

Fact:

1. Barcan formula (8) expresses
if $R(w, w')$ then $D(w') \subseteq D(w)$
2. converse Barcan formula (9) expresses the *principle of nested domains*:

$$\text{if } R(w, w') \text{ then } D(w) \subseteq D(w'). \quad \square$$

Proof of 1: — see figure below.



The objectual interpretation: logic Q1R (world-relative domains)

Logic Q1R is defined over *world-relative model with rigid terms* which are of the form $\langle \mathcal{K}, D, Q, a \rangle$, where a satisfies condition (5).

Important:

Let S be a propositional modal logic sound and complete w.r.t. a class C of Kripke frames. Then the system Q1R-S is sound and complete w.r.t. class C , where Q1R-S is obtained from S by adding the principles of the minimal free logic, MFL, and rules (3), (4), (6) and (7).

The objectual interpretation: nested domains

In order to reflect the principle of nested domains one simply accepts the converse of the Barcan formula. It should be emphasized that the Barcan formula itself is provable in KB and stronger logic using the classical rules for quantifiers.

**The conceptual interpretation: logic Q2
(world-relative domains)**

In Q2 we assume that in models $\langle \mathcal{K}, D, Q, a \rangle$ (recall that $\mathcal{K} = \langle W, R \rangle$), Q is a function assigning a domain $D(w)$ to each world w , and a satisfies the condition:

$a(\forall x A)(w) = \text{TRUE}$, iff
for any $f : W \rightarrow D$, if $f(w) \in D(w)$,
then $a(f/x)(A)(w) = \text{TRUE}$,

where $a(f/x)$ represents the assignment function identical to a except that x is substituted by f .

**The conceptual interpretation: logic QC
(fixed domains)**

In QC we assume that in models $\langle \mathcal{K}, D, Q, a \rangle$ Q is the set of functions from W into D and a satisfies the condition:

$a(\forall x A)(w) = \text{TRUE}$, iff
for any $f \in Q$, $a(f/x)(A)(w) = \text{TRUE}$.

Case study: quantified temporal logic of programs

The goal is to be able to express:

- **Invariance (safety) properties**
 - $A(\bar{x}) \rightarrow \Box B(\bar{x})$ (all states reached by a program after the state satisfying $A(\bar{x})$ will satisfy $B(\bar{x})$)
 - $(atFirst \rightarrow A(\bar{x})) \rightarrow \Box(atEnd \rightarrow B(\bar{x}))$ (partial correctness w.r.t conditions $A(\bar{x})$ and $B(\bar{x})$, where propositional variable $atFirst$ is true only at the beginning of the specified program, while $atEnd$ is true only when the program reaches its terminal state)
 - $\Box(\neg B(\bar{x}) \vee \neg A(\bar{x}))$ (the program cannot enter critical regions $A(\bar{x})$ and $B(\bar{x})$ simultaneously (mutual exclusion)).
- **Eventuality properties**
 - $A(\bar{x}) \rightarrow \Diamond B(\bar{x})$ (there is a program state satisfying $B(\bar{x})$ reached by a program after the state satisfying $A(\bar{x})$)
 - $(atFirst \rightarrow A(\bar{x})) \rightarrow \Diamond(atEnd \wedge B(\bar{x}))$ (total correctness w.r.t. conditions $A(\bar{x})$ and $B(\bar{x})$)
 - $\Box \Diamond A(\bar{x}) \rightarrow \Diamond B(\bar{x})$ (repeating a request $A(\bar{x})$ will force a response $B(\bar{x})$)
 - $\Box A(\bar{x}) \rightarrow \Diamond B(\bar{x})$ (permanent holding a request $A(\bar{x})$ will force a response $B(\bar{x})$).

Recall that time structures are strongly application oriented (see slide 58). In the case of temporal logics of programs one can accept branching time (e.g., reflecting non-determinism), linear time, time consisting of intervals (— when could it be useful?).

Here we, for simplicity, we consider linear and discrete time corresponding to program execution (parallel programs are modelled here by the interleaving semantics) and we mainly illustrate properties discussed previously.



We shall focus on temporal logic with modalities \Box and \Diamond only, but the area is much more subtle (see the additional material supplied via the course web page).

We ask the following questions:

1. are terms rigid or flexible?
2. what choice as to underlying domains suits best? In which cases domains are to be flexible, in which fixed?

Some axioms for the considered temporal logic of programs

One of reasonable choices is the logic Q1-S over a suitably chosen base modal logic S, i.e., we accept the following axioms characteristic for Q1:

- (6), i.e., $t = t$
- (7), i.e., $t = t' \vdash P(t) \rightarrow P(t')$, where P is an atom
- (8), i.e., $\forall x \Box A(x) \rightarrow \Box \forall x A(x)$ (Barcan formula)

In addition it is reasonable to accept

- (9), i.e., $\Box \forall x A(x) \rightarrow \forall x \Box A(x)$ (converse of the Barcan formula)

S can contain, among others, axioms:

- $\Box A \rightarrow A$ (reflexivity)
- $\Box A \rightarrow \Box \Box A$ (transitivity)
- $(\Diamond A \wedge \Diamond B) \rightarrow [\Diamond(A \wedge \Diamond B) \vee \Diamond(B \wedge \Diamond A)]$ (linearity)

Exercises

1. Check that the last axiom above indeed forces linearity of time.
2. Propose some other properties of time and express them as temporal axioms.

What you should have learnt from Lecture V?

- what are quantified modal logics?
- what is free logic?
- what models are assumed for quantified modal logics?
- what are objectual and conceptual interpretations?
- what choices as to terms and domains can be made?
- what is the meaning of Barcan formula and converse Barcan formula?

3-valued logics

In many applications one cannot attach to formulas value TRUE nor FALSE. Some facts might be UNKNOWN, some other INCONSISTENT, etc. In consequence, there are many logics accepting truth values other than {TRUE, FALSE}.
The meaning of other truth values is strongly application-dependent.

Examples

1. Incomplete knowledge:
"temperature in Paris is now 12°C" – TRUE or FALSE?
— it is TRUE or FALSE, but do we know the value?
If not, which one is to be accepted in practical reasoning?
2. Inaccurate knowledge:
"my eyes are grey" – TRUE or FALSE?
In the case of uncertainty we cannot assume TRUE or FALSE.
3. Nonsense:
"If the Sun is pink then elephants are red" – TRUE or FALSE?
— The classical logic would say TRUE, but maybe we find such an implication to be NONSENSE?
4. Inconsistency:
— maybe two different information sources claim the opposite about whether a given object is red. Is it then red? Or maybe we find this information INCONSISTENT?

3-valued logic of Łukasiewicz L_3

- Introduced by Łukasiewicz in 1920.
- Logical values: TRUE, FALSE, NEUTRAL.
- Connectives: $\neg, \rightarrow, \leftrightarrow, \vee, \wedge$.

Examples

1. Voting is basically three-valued:
TRUE (I do support), FALSE (I do not support) or NEUTRAL.
2. Do I support a given political party?
3. Do I like certain types of cars, certain sounds, tastes or certain colors?

The semantics of many-valued logics is usually given by means of truth tables. However, the reasoning is usually supported by proof systems.

Truth tables for L_3

	FALSE	NEUTRAL	TRUE
\neg	TRUE	NEUTRAL	FALSE

\rightarrow	FALSE	NEUTRAL	TRUE
FALSE	TRUE	TRUE	TRUE
NEUTRAL	NEUTRAL	TRUE	TRUE
TRUE	FALSE	NEUTRAL	TRUE

\leftrightarrow	FALSE	NEUTRAL	TRUE
FALSE	TRUE	NEUTRAL	FALSE
NEUTRAL	NEUTRAL	TRUE	NEUTRAL
TRUE	FALSE	NEUTRAL	TRUE

\vee	FALSE	NEUTRAL	TRUE
FALSE	FALSE	NEUTRAL	TRUE
NEUTRAL	NEUTRAL	NEUTRAL	TRUE
TRUE	TRUE	TRUE	TRUE

\wedge	FALSE	NEUTRAL	TRUE
FALSE	FALSE	FALSE	FALSE
NEUTRAL	FALSE	NEUTRAL	NEUTRAL
TRUE	FALSE	NEUTRAL	TRUE

Reasoning in L_3

- Suppose that John neither likes nor dislikes orange juice and likes apple juice. What is the truth value of sentences:
 - “John likes orange juice or apple juice”?
 - “John likes orange juice and apple juice”?
- Is the following sentence true:

“John likes orange juice and apple juice, therefore he likes orange juice”?
- Check whether:
 - $(A \rightarrow B) \equiv \neg A \vee B$
 - $A \vee B \equiv (A \rightarrow B) \rightarrow B$; $A \wedge B \equiv \neg(\neg A \vee \neg B)$
 - $A \rightarrow (B \rightarrow A)$
- Can neutrality model indefiniteness?

Consider sentence:
 “Eve will vote for party XYZ provided that the party will not change its current policy”.
 This sentence can be formulated as implication

$$\text{noChange} \rightarrow \text{EveWillVote}. \tag{10}$$

Suppose that at the moment it is undetermined whether sentences *noChange* and *EveWillVote* are TRUE or FALSE. Using L_3 we would have to assume that the value of implication (10) is TRUE. However, it should stay undetermined, since *noChange* might further appear TRUE while, for some reasons, *EveWillVote* might appear FALSE.

Wajsberg’s axiomatization of L_3

Wajsberg’s axiomatization of L_3 :

- axioms
 - $A \rightarrow (B \rightarrow A)$
 - $(A \rightarrow B) \rightarrow [(B \rightarrow C) \rightarrow (A \rightarrow C)]$
 - $(\neg B \rightarrow \neg A) \rightarrow (A \rightarrow B)$
 - $[(A \rightarrow \neg A) \rightarrow A] \rightarrow A$
- rule:

$$A, A \rightarrow B \vdash B.$$

Theorem (Wajsberg)

Wajsberg’s axiomatization is sound and complete for L_3 . □

Exercise

Check the validity of Wajsberg’s axioms and soundness of modus ponens.

Kripke-like semantics for L_3

Kripke structure for L_3 is a pair $\langle \{0, 1, 2\}, \leq \rangle$, where \leq is the standard ordering on numbers 0, 1, 2. Assume that the consequence relation \Vdash is defined on atoms (i.e., the meaning of $x \Vdash A$ is given when A is a propositional variable) and that \Vdash always satisfies condition:

$$x \Vdash A \text{ and } x \leq y \text{ implies } y \Vdash A.$$

Semantics of L_3 is then defined by:

- $x \Vdash \neg A$ iff $2 - x \not\Vdash A$
- $x \Vdash A \rightarrow B$ iff $\forall y \leq 2 - x [y \not\Vdash A \text{ implies } y - x \Vdash B]$.

Fact

- $x \Vdash A \vee B$ iff $x \Vdash A$ or $x \Vdash B$
- $x \Vdash A \wedge B$ iff $x \Vdash A$ and $x \Vdash B$. □

Sheffer stroke

Sheffer in 1913 provided a connective, called the *Sheffer stroke* and denoted by $|$, which is sufficient to express any other connective in the case of two-valued propositional logic.

In 1936 Webb provided a connective which allows one to define any other three-valued connective (in fact, he covered the case of arbitrary finite valued logics).

Webb 3-valued operator

	FALSE	THIRD	TRUE
FALSE	THIRD	TRUE	FALSE
THIRD	TRUE	TRUE	FALSE
TRUE	FALSE	FALSE	FALSE

The Wesselkamper operator

In 1975 Wesselkamper proposed a three-argument operator S , which allows one to define all arbitrary (finite) valued connectives:

$$S(x, y, z) \stackrel{\text{def}}{=} \begin{cases} z & \text{if } x = y \\ x & \text{otherwise.} \end{cases}$$

Examples

For simplicity consider the case of the classical two-valued logic:

1. negation:

$$\neg P = S(\text{TRUE}, P, \text{FALSE})$$

2. implication:

$$P \rightarrow Q = S(\text{TRUE}, P, Q)$$

3. conjunction:

$$P \wedge Q = S(P, \text{TRUE}, Q)$$

4. disjunction:

$$P \vee Q = S(P, \text{FALSE}, Q)$$

5. Sheffer stroke:

$$P | Q = S(\text{TRUE}, S(P, \text{TRUE}, Q), \text{FALSE}).$$

3-valued logic of Kleene K_3 (strong Kleene logic)

- Introduced by Kleene in 1952.
- Logical values: TRUE, FALSE, INDEFINITE.
- Connectives: $\neg, \rightarrow, \leftrightarrow, \vee, \wedge$.

Examples

1. In reals, what is the truth value of formula $\sqrt{-2} = 1$?
2. Some programs loop forever. What are properties of their final states?
3. What is the value of formula $brother(\text{NULL}, John)$, when NULL represents a value that is missing, for example a missing value in a database table?
4. It might be undetermined at the moment whether a given sentence is TRUE or FALSE — see Example 4, p. 85.
5. When we ask a robot about a certain situation, it quite often has to spend some time determining certain properties and might temporarily answer INDEFINITE or UNKNOWN.

Truth tables for K_3

	FALSE	INDEFINITE	TRUE
\neg	TRUE	INDEFINITE	FALSE

\rightarrow	FALSE	INDEFINITE	TRUE
FALSE	TRUE	TRUE	TRUE
INDEFINITE	INDEFINITE	INDEFINITE	TRUE
TRUE	FALSE	INDEFINITE	TRUE

\leftrightarrow	FALSE	INDEFINITE	TRUE
FALSE	TRUE	INDEFINITE	FALSE
INDEFINITE	INDEFINITE	INDEFINITE	INDEFINITE
TRUE	FALSE	INDEFINITE	TRUE

\vee	FALSE	INDEFINITE	TRUE
FALSE	FALSE	INDEFINITE	TRUE
INDEFINITE	INDEFINITE	INDEFINITE	TRUE
TRUE	TRUE	TRUE	TRUE

\wedge	FALSE	INDEFINITE	TRUE
FALSE	FALSE	FALSE	FALSE
INDEFINITE	FALSE	INDEFINITE	INDEFINITE
TRUE	FALSE	INDEFINITE	TRUE

3-valued logic of Bočvar (Bochvar)

- Introduced by Bočvar in 1939.
- Logical values: TRUE, FALSE, NONSENSE.
- Connectives: $\neg, \rightarrow, \leftrightarrow, \vee, \wedge$.

Examples

What is the truth value of sentences:

- “Small pink elephants do not like to fly”
- “If NY is the capitol of the US then Sweden is in the EU”
(in the classical logic this sentence is TRUE, but should it be the case?)

In the natural language NONSENSE often means FALSE. This meaning should not be mixed with the concept of NONSENSE, as understood above.

Truth tables for the logic of Bočvar

	FALSE	NONSENSE	TRUE
\neg	TRUE	NONSENSE	FALSE

\rightarrow	FALSE	NONSENSE	TRUE
FALSE	TRUE	NONSENSE	TRUE
NONSENSE	NONSENSE	NONSENSE	NONSENSE
TRUE	FALSE	NONSENSE	TRUE

\leftrightarrow	FALSE	NONSENSE	TRUE
FALSE	TRUE	NONSENSE	FALSE
NONSENSE	NONSENSE	NONSENSE	NONSENSE
TRUE	FALSE	NONSENSE	TRUE

\vee	FALSE	NONSENSE	TRUE
FALSE	FALSE	NONSENSE	TRUE
NONSENSE	NONSENSE	NONSENSE	TRUE
TRUE	TRUE	TRUE	TRUE

\wedge	FALSE	NONSENSE	TRUE
FALSE	FALSE	FALSE	FALSE
NONSENSE	FALSE	NONSENSE	NONSENSE
TRUE	FALSE	NONSENSE	TRUE

3-valued logic of Hallden

- Introduced by Hallden in 1949.
- Logical values: TRUE, FALSE, MEANINGLESS.
- Connectives: \neg, \vdash, \wedge ($\vdash \alpha$ stands for “ α is meaningful”).

- $\alpha \vee \beta \stackrel{\text{def}}{=} \neg(\neg\alpha \wedge \neg\beta)$
- $\alpha \rightarrow \beta \stackrel{\text{def}}{=} \neg\alpha \vee \beta$
- $\alpha \leftrightarrow \beta \stackrel{\text{def}}{=} (\alpha \rightarrow \beta) \wedge (\beta \rightarrow \alpha)$
- $-\alpha \stackrel{\text{def}}{=} \neg \vdash \alpha$ ($-\alpha$ stands for “ α is meaningless”).

Examples

1. Many sentences are meaningful only in a certain context, e.g., what is the meaning of “He did it.”? We know that somebody has done something, but who, when, what?
2. Classical tautologies often provide no “real” meaning, e.g., “If whenever you do not eat then you eat, then you eat.”
3. Sometimes a sentence is meaningless for those who do not know its meaning, e.g., what is the meaning of “Ta lódź jest żółta.”?

Truth tables for the logic of Hallden

	FALSE	MEANINGLESS	TRUE
\neg	TRUE	MEANINGLESS	FALSE
\vdash	TRUE	FALSE	TRUE

\wedge	FALSE	MEANINGLESS	TRUE
FALSE	FALSE	MEANINGLESS	FALSE
MEANINGLESS	MEANINGLESS	MEANINGLESS	MEANINGLESS
TRUE	FALSE	MEANINGLESS	TRUE

Exercises

1. Consider examples provided on slide 85, replacing L_3 by:
 - (a) 3-valued logics of Kleene
 - (b) 3-valued logic of Bočvar
 - (c) 3-valued logic of Hallden.
2. Develop a 3-valued logic with the third value IRRATIONAL. — Compare it to the logics presented so far.
3. Develop a 3-valued logic with the third value INCONSISTENT. — Compare it to the logics presented so far.
4. Develop a 3-valued logic with the third value POSSIBLE. — Compare it to the logics presented so far.

External negation

The negation considered so far is called the *internal negation*. The *external negation* is defined by table:

	FALSE	UNKNOWN	TRUE
\sim	TRUE	TRUE	FALSE

External negation corresponds to the closed world assumption CWA.

Example

Suppose that our knowledge database does not entail the fact “There is a car to the right” (abbreviated further by P). Thus P is FALSE or UNKNOWN, i.e., $\text{FALSE} \vee \text{UNKNOWN}$, which usually is UNKNOWN (at least in all logics we have considered). According to the CWA, when we query the database about $\sim P$ in the given circumstances, we receive the answer TRUE. Observe, that actually P is UNKNOWN, thus $\sim P = \sim \text{UNKNOWN} = \text{TRUE}$, as needed.

What you should have learnt from Lecture VI?

- What are 3-valued logics?
- 3-valued logic of Lukasiewicz.
- Wajsberg axiomatization of the Lukasiewicz logic.
- Kripke-like semantics for the Lukasiewicz logic.
- What is the Sheffer stroke?
- 3-valued logics of Kleene.
- 3-valued logic of Bočvar.
- 3-valued logic of Hallden.
- External versus internal negation.

Many-valued logics

3-valued logics can be generalized to deal with more truth values. Systems of *many-valued logics* are usually defined by fixing:

- the set of truth values
- the truth degree functions which interpret the propositional connectives
- the semantical interpretation of the quantifiers
- the *designated truth values*, which act as substitutes for the truth value TRUE (a formula is considered valid under some interpretation *I* iff it has a designated truth value under *I*).

Examples of well-known many valued logics

1. Logics of Lukasiewicz (1920).
2. Logics of Post (1921).
3. logics of Gödel (1932).
4. Logics of Kleene (1938).
5. Fuzzy logic of Zadeh (1965).
6. Logic of Belnap (1977).

4-valued logic of Belnap

Consider situations with many information sources. Then a fact *A* might be:

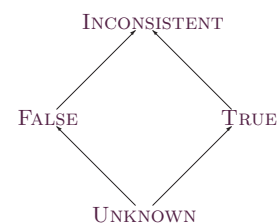
- TRUE, e.g., when some sources claim that *A* holds and no source claims the contrary
- FALSE, e.g., when some sources claim that *A* does not hold and no source claims the contrary
- UNKNOWN, e.g., when all sources claim that *A* is unknown
- INCONSISTENT, e.g., when some sources claim that *A* holds and some sources claim that *A* does not hold (and perhaps some sources claim that *A* is unknown).

Remark

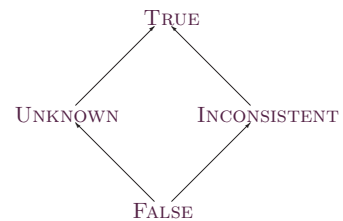
In the classical logic inconsistency is modelled by FALSE. Moreover, FALSE implies any other formula, so inconsistent theories are trivial (their consequences consist of all formulas of the classical logic).

Belnap's logic allows us to deal with inconsistent information which does not trivialize inconsistent theories which are, in fact, quite often in our reality.

Orderings of truth values



Information ordering
= Knowledge ordering
= Approximation ordering



Truth ordering

Truth tables for connectives (reflecting truth ordering)

	UNKNOWN	FALSE	TRUE	INCONSISTENT
\neg	UNKNOWN	TRUE	FALSE	INCONSISTENT

\rightarrow	FALSE	UNKNOWN	INCONS.	TRUE
FALSE	TRUE	TRUE	TRUE	TRUE
UNKNOWN	FALSE	TRUE	FALSE	TRUE
INCONS.	FALSE	FALSE	TRUE	TRUE
TRUE	FALSE	FALSE	FALSE	TRUE

\vee	FALSE	UNKNOWN	INCONS.	TRUE
FALSE	FALSE	UNKNOWN	INCONS.	TRUE
UNKNOWN	UNKNOWN	UNKNOWN	TRUE	TRUE
INCONS.	INCONS.	TRUE	INCONS.	TRUE
TRUE	TRUE	TRUE	TRUE	TRUE

\wedge	FALSE	UNKNOWN	INCONS.	TRUE
FALSE	FALSE	FALSE	FALSE	FALSE
UNKNOWN	FALSE	UNKNOWN	FALSE	UNKNOWN
INCONS.	FALSE	FALSE	INCONS.	INCONS.
TRUE	FALSE	UNKNOWN	INCONS.	TRUE

Exercises

1. Consider examples 1 and 2 from slide 85 and analyze them from the point of view of the Belnap's logic.
2. Consider sentences "John likes cinema and he does not like cinema. Therefore he will go to a restaurant". Analyze these sentences from the point of view of the truth tables for connectives in the Belnap's logic.
3. Consider truth tables for connectives in Belnap's logic. Do you find them intuitive?
4. Construct truth tables for connectives $\rightarrow, \otimes, \oplus$ reflecting knowledge ordering shown in slide 100 (here we deal with *consensus* \otimes and *accept all* \oplus rather than with conjunction and disjunction).
5. Imagine another ordering of truth values and provide truth tables reflecting this ordering.
6. Consider two worlds w_1 and w_2 equipped with the classical propositional logic. Let:
 - $w_1 \cap w_2$ be a world with the set of true formulas defined as the set of formulas true in both w_1 and w_2
 - $w_1 \cup w_2$ be a world with the set of true formulas defined as the set of formulas true in w_1 or w_2 .

Then $w_1 \cap w_2$ is called an *underdefined world* and $w_1 \cup w_2$ is called an *overdefined world*.

Analyze notions of underdefined and overdefined using the Belnap's logic.

Lattices and bilattices

A *lattice* is a structure $L = \langle U, \leq \rangle$, where \leq is a partial order such that all nonempty finite subsets of U have a least upper bound wrt \leq (also called *join*) and a greatest lower bound wrt \leq (also called *meet*).

A *bilattice* is a structure $B = \langle U, \leq_t, \leq_k, \neg \rangle$ such that U is a set containing at least two elements, $\langle U, \leq_t \rangle$ and $\langle U, \leq_k \rangle$ are lattices and \neg is a unary operation on U with the following properties, where $x, y \in U$:

- if $x \leq_t y$ then $\neg y \leq_t \neg x$
- if $x \leq_k y$ then $\neg x \leq_k \neg y$
- $\neg \neg x = x$.

Usually \wedge, \vee stand for the meet and join wrt \leq_t and \otimes, \oplus stand for the meet and join wrt \leq_k .

Operations \wedge, \vee are also called the *conjunction* and *disjunction*, and \otimes, \oplus are called *consensus* and *accept all*.

Constructing bilattices

Let $L_1 = \langle U, \leq_1 \rangle$ and $L_2 = \langle U, \leq_2 \rangle$ be lattices. A *bilattice product* is the bilattice

$$L_1 \odot L_2 \stackrel{\text{def}}{=} \langle U \times U, \leq_t, \leq_k, \neg \rangle,$$

where

- $\langle x_1, x_2 \rangle \leq_t \langle y_1, y_2 \rangle \stackrel{\text{def}}{=} x_1 \leq_1 y_1$ and $y_2 \leq_2 x_2$
- $\langle x_1, x_2 \rangle \leq_k \langle y_1, y_2 \rangle \stackrel{\text{def}}{=} x_1 \leq_1 y_1$ and $x_2 \leq_2 y_2$
- $\neg \langle x, y \rangle \stackrel{\text{def}}{=} \langle y, x \rangle$.

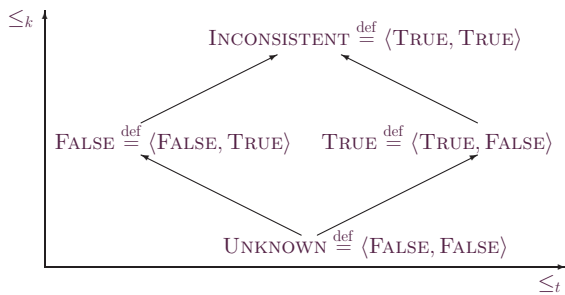
An increase in truth is caused by an increase of positive evidence and decrease of negative evidence. Knowledge increases if any of evidences increases, no matter whether positive or negative.

Intuitions

Consider $\langle x, y \rangle$ to be $\langle p^+, p^- \rangle$:

- $\langle \emptyset, \emptyset \rangle = \langle \text{FALSE}, \text{FALSE} \rangle \stackrel{\text{def}}{=} \text{UNKNOWN}$ (nothing positive or negative is known)
- $\langle \text{All}, \text{All} \rangle = \langle \text{TRUE}, \text{TRUE} \rangle \stackrel{\text{def}}{=} \text{INCONSISTENT}$ (everything both positive and negative is known)
- $\langle \emptyset, \text{All} \rangle = \langle \text{FALSE}, \text{TRUE} \rangle \stackrel{\text{def}}{=} \text{FALSE}$ (nothing positive and all negative is known)
- $\langle \text{All}, \emptyset \rangle = \langle \text{TRUE}, \text{FALSE} \rangle \stackrel{\text{def}}{=} \text{TRUE}$ (all positive and nothing negative is known).

The bilattice corresponding to Belnap's logic



More generally

In the above lattice we considered pairs $\langle P, N \rangle$, where P, N contain either nothing (\emptyset) or everything (All).

Let now $P, N \subseteq All$ and let $\langle P, N \rangle$ be the positive and the negative region of a relation R , respectively. Then:

- for all objects $u \in P - N$ we have that $R(u) = \text{TRUE}$
- for all objects $u \in N - P$ we have that $R(u) = \text{FALSE}$
- for all objects $u \in -(P \cup N)$ we have that $R(u) = \text{UNKNOWN}$
- for all objects $u \in P \cap N$ we have that $R(u) = \text{INCONSISTENT}$.

Many valued logic of Lukasiewicz

Lukasiewicz and Tarski (1930) have generalized L_3 by allowing valuations to take any value in $[0, 1]$.

Let PV be the set of propositional variables.

An *L-valuation* is a mapping $e : PV \rightarrow [0, 1]$.

It is extended to all formulas by setting:

- $e(\neg A) \stackrel{\text{def}}{=} 1 - e(A)$
- $e(A \vee B) \stackrel{\text{def}}{=} \max[e(A), e(B)]$
- $e(A \wedge B) \stackrel{\text{def}}{=} \min[e(A), e(B)]$
- $e(A \rightarrow B) \stackrel{\text{def}}{=} \begin{cases} 1 & \text{when } e(A) \leq e(B) \\ 1 - e(A) + e(B) & \text{otherwise.} \end{cases}$
- $e(A \equiv B) \stackrel{\text{def}}{=} \begin{cases} 1 & \text{when } e(A) = e(B) \\ 1 - e(B) + e(A) & \text{when } e(A) < e(B) \\ 1 - e(A) + e(B) & \text{otherwise.} \end{cases}$

We define logics $L_n, L_{\mathbb{N}_0}$ and $L_{\mathbb{R}}$ as follows:

- a formula A is a tautology of L_n iff $e(A) = 1$ for every L -valuation $e : PV \rightarrow \left\{ \frac{m}{n-1} : 0 \leq m \leq n-1 \right\}$
- a formula A is a tautology of $L_{\mathbb{N}_0}$ iff $e(A) = 1$ for every L -valuation $e : PV \rightarrow \{r \in [0, 1] : r \text{ is a rational number}\}$
- a formula A is a tautology of $L_{\mathbb{R}}$ iff $e(A) = 1$ for every L -valuation e .

Fact

1. $A \vee B \equiv (A \rightarrow B) \rightarrow B$
2. $A \wedge B \equiv \neg(\neg A \vee \neg B)$. □

Theorem

1. The set of tautologies of L_2 is that of the classical propositional logic
2. $L_{n+1} \neq L_n$
3. $L_{\mathbb{N}_0} = L_{\mathbb{R}} \subseteq L_n$, for any n . □

Many valued logic of Gödel

The *logics of Gödel*, G_n and G_∞ are defined by either a finite set of reals within $[0, 1]$,

$$W_n \stackrel{\text{def}}{=} \left\{ \frac{m}{n-1} : 0 \leq m \leq n-1 \right\}$$

or the whole interval $[0, 1]$ as the truth degree set. The degree 1 is the only designated truth degree.

Let PV be the set of propositional variables.

Any valuation $e : PV \rightarrow W_n$ or $e : PV \rightarrow [0, 1]$ is extended to all formulas by setting:

- $e(\neg A) \stackrel{\text{def}}{=} \begin{cases} 1 & \text{for } e(A) = 0 \\ 0 & \text{otherwise} \end{cases}$
- $e(A \vee B) \stackrel{\text{def}}{=} \max\{e(A), e(B)\}$
- $e(A \wedge B) \stackrel{\text{def}}{=} \min\{e(A), e(B)\}$
- $e(A \rightarrow B) \stackrel{\text{def}}{=} \begin{cases} 1 & \text{when } e(A) \leq e(B) \\ e(B) & \text{otherwise.} \end{cases}$
- $e(A \equiv B) \stackrel{\text{def}}{=} \begin{cases} 1 & \text{when } e(A) = e(B) \\ \min\{e(A), e(B)\} & \text{otherwise.} \end{cases}$

Fuzzy sets

In the classical sense set membership is two-valued (an element belongs or does not belong to a set). We can then consider any set, say A , via its membership function

$$A : \text{DOM} \rightarrow \{\text{TRUE}, \text{FALSE}\}.$$

Zadeh (1965) proposed to generalize the *membership function* to return a number from the closed interval $[0, 1]$ of reals rather than from $\{\text{TRUE}, \text{FALSE}\}$:

$$A : \text{DOM} \rightarrow [0, 1].$$

An intuitive meaning of A is now the degree to which a given element belongs to the set A .

Example

Denote by *highSpeed* the set consisting of high speeds of a car on a highway. Then, e.g.,

- $\text{highSpeed}(3 \frac{\text{km}}{\text{h}}) = 0$
- $\text{highSpeed}(110 \frac{\text{km}}{\text{h}}) = 0.72$
- $\text{highSpeed}(160 \frac{\text{km}}{\text{h}}) = 1$

— can we provide an intuitive classical definition (with two-valued membership function) of *highSpeed*?

Fuzzy sets operations and inclusion

Let A and B be fuzzy sets (given by their membership functions). The membership function is extended to operations on sets and set inclusion as follows (other definitions are possible but those below are most common):

- for all $x \in \text{DOM}$, $(\neg A)(x) \stackrel{\text{def}}{=} 1 - A(x)$
- for all $x \in \text{DOM}$, $(A \cup B)(x) \stackrel{\text{def}}{=} \max\{A(x), B(x)\}$
- for all $x \in \text{DOM}$, $(A \cap B)(x) \stackrel{\text{def}}{=} \min\{A(x), B(x)\}$
- $A \subseteq B \stackrel{\text{def}}{=} \text{for all } x \in \text{DOM}, A(x) \leq B(x).$

Example

Consider sets of young persons and tall persons, which are subsets of $\{\text{Anne}, \text{Eve}, \text{Jack}, \text{Mary}, \text{Steve}\}$:

person	Anne	Eve	Jack	Mary	Steve
young	0.6	0.5	0.7	0.4	0.9
tall	0.4	0.7	0.6	0.7	0.8

Then:

person	Anne	Eve	Jack	Mary	Steve
young \cup tall	0.6	0.7	0.7	0.7	0.9
young \cap tall	0.4	0.5	0.6	0.4	0.8
\neg tall	0.6	0.3	0.4	0.3	0.2

Example

Suppose we have a choice of five meals in a restaurant. Assume further that fuzzy sets “tasty” and “expensive” are given by

meal	1	2	3	4	5
tasty	0.3	0.8	0.7	0.4	0.8
expensive	0.2	0.6	0.9	0.6	0.9

Then we have (notice some drawbacks in the results):

meal	1	2	3	4	5
inexpensive (\neg expensive)	0.8	0.4	0.1	0.4	0.1
tasty and (\cap) inexpensive	0.3	0.4	0.1	0.4	0.1

Consider an alternative definition intersection:

- for all $x \in \text{DOM}$, $(A \cap B)(x) \stackrel{\text{def}}{=} A(x) * B(x)$

With the new definitions we have:

meal	1	2	3	4	5
tasty and (\cap) inexpensive	0.24	0.32	0.07	0.16	0.08

which is much better.

The choice of definitions is then application-dependent.

The definition of union, fitting the new definition of intersection may be:

- for all $x \in \text{DOM}$, $(A \cup B)(x) \stackrel{\text{def}}{=} A(x) + B(x) - A(x) * B(x).$

Linguistic modifiers

A *linguistic modifier*, is an operation that modifies the meaning of a term. From the perspective of fuzzy sets, linguistic modifiers are operations on fuzzy sets.

Examples

1. The modifier “very” which, e.g., transforms the statement “Jack is young” to “Jack is very young.” The modifier “very” is usually defined to be

$$\text{very}A(x) \stackrel{\text{def}}{=} A(x)^2.$$

If $\text{young}(\text{Jack}) = 0.7$ then:

- $\text{very young}(\text{Jack}) = 0.7^2 = 0.49$
- $\text{very } \underbrace{\text{very young}}_{0.49}(\text{Jack}) = 0.49^2 = 0.2401.$

2. The modifier “extremely” can be defined by

$$\text{extremely}A(x) \stackrel{\text{def}}{=} A(x)^3.$$

E.g., $\text{extremely young}(\text{Jack}) = 0.7^3 = 0.343.$

3. The modifier “slightly” can be defined by

$$\text{slightly}A(x) \stackrel{\text{def}}{=} \sqrt{A(x)}.$$

E.g., $\text{slightly young}(\text{Jack}) = \sqrt{0.7} \approx 0.837.$

Fuzzy reasoning

In fuzzy reasoning we are interested in truth degree of complex sentences on the basis of truth degrees of atomic sentences. Any formula can be viewed as a set of objects satisfying the formula.

As in the classical logic, conjunction and disjunction are defined like intersection and union on sets. The choice of implication is less obvious. One of common choices is to accept the Gödel’s implication (see slide 109). However, in applications like fuzzy controllers one frequently uses other definitions of implication.

Examples

1. Formula $\text{tall}(x)$ representing the set of tall persons has the truth value equal to the fuzzy membership of x in the (fuzzy) set tall . E.g., $\text{tall}(\text{John})$ might be 0.7 (if $\text{John} \in \text{tall}$ to the degree 0.7).

2. Consider formula

$$\text{walk}(x) \vee \text{cinema}(x)$$

and suppose that $\text{walk}(\text{Eve}) = 0.6$ and $\text{cinema}(\text{Eve}) = 0.5$. Then the truth degree of $\text{walk}(\text{Eve}) \vee \text{cinema}(\text{Eve})$ is

$$\max\{0.6, 0.5\} = 0.6.$$

Example

Consider sentence

“If the water level in a tank is high then the tank’s valve is opened widely”.

Assume that if the tank is filled in $p\%$ then the water level is high to degree $\frac{p}{100}$ and that the same characteristics applies to the term “widely”.

The sentence has a pattern $A \rightarrow B$, where

- A stands for “Water level in a tank is high”
- B stands for “Tank’s valve is opened widely”.

Suppose that:

- the tank is filled in 60% (i.e., A holds to the degree 0.6)
- the valve is opened in 50% (i.e., B holds to the degree 0.5).

Then, assuming Gödel’s implication, $A \rightarrow B$ is true to degree 0.5.

In the case when

- the tank is filled in 50% (i.e., A holds to the degree 0.5)
- the valve is opened in 60% (i.e., B holds to the degree 0.6).

Then, according to Gödel’s implication, $A \rightarrow B$ is true to the degree 1.0.

What you should have learnt from Lecture VII?

- What are many-valued logics?
- What is the set of designated truth values?
- 4-valued logic of Belnap.
- Knowledge ordering versus truth ordering.
- What are bilattices?
- Many-valued logic of Łukasiewicz.
- Many-valued logic of Gödel.
- Fuzzy sets and operations on fuzzy sets.
- Fuzzy implication.
- Fuzzy reasoning.

Intuitionism

Intuitionism (Brouwer, 1907) is based on the philosophy that the meaning of a statement is not given by truth conditions but by the means of proof or verification, which are to be constructive. Brouwer, Heyting, and Kolmogorov formulated the following principles:

- a proof of $A \wedge B$ is given by presenting a proof of A and a proof of B
- a proof of $A \vee B$ is given by presenting either a proof of A or a proof of B and indicating which proof it is
- a proof of $A \rightarrow B$ is a procedure which permits us to transform a proof of A into a proof of B
- the constant FALSE, a contradiction, has no proof
- a proof of $\neg A$ is a procedure that transforms any hypothetical proof of A into a proof of a contradiction
- a proof of $\exists x A(x)$ is a procedure that selects object a and provides a proof of $A(a)$, where a is a constant symbol denoting a
- a proof of $\forall x A(x)$ is a procedure which, for any object a of a given universe, constructs a proof of $A(a)$, where a is as above.

Heyting's axiom system

Heyting (1930) presented an axiom system which has been accepted as the *intuitionistic propositional calculus*:

• axioms

1. $A \rightarrow (A \wedge A)$
2. $(A \wedge B) \rightarrow (B \wedge A)$
3. $(A \rightarrow B) \rightarrow [(A \wedge C) \rightarrow (B \wedge C)]$
4. $[(A \rightarrow B) \wedge (B \rightarrow C)] \rightarrow (A \rightarrow C)$
5. $A \rightarrow (B \rightarrow A)$
6. $[A \wedge (A \rightarrow B)] \rightarrow B$
7. $A \rightarrow (A \vee B)$
8. $(A \vee B) \rightarrow (B \vee A)$
9. $[(A \rightarrow C) \wedge (B \rightarrow C)] \rightarrow [(A \vee B) \rightarrow C]$
10. $\neg A \rightarrow (A \rightarrow B)$
11. $[(A \rightarrow B) \wedge (A \rightarrow \neg B)] \rightarrow \neg A$

• proof rules

1. modus ponens

$$\frac{A, A \rightarrow B}{B}$$
2. adjunction

$$\frac{A, B}{A \wedge B}$$

Examples of derivations

1. transitivity of implication (trans): $\frac{A \rightarrow B, B \rightarrow C}{A \rightarrow C}$
 - $A \rightarrow B$ — premise (11)
 - $B \rightarrow C$ — premise (12)
 - $(A \rightarrow B) \wedge (B \rightarrow C)$ — adjunction on (11) and (12) (13)
 - $[(A \rightarrow B) \wedge (B \rightarrow C)] \rightarrow (A \rightarrow C)$ — axiom 4 (14)
 - $A \rightarrow C$ — modus ponens on (13) and (14)

2. $(A \wedge B) \rightarrow A$
 - $A \rightarrow (B \rightarrow A)$ — axiom 5 (15)
 - $(A \rightarrow (B \rightarrow A)) \rightarrow [(A \wedge B) \rightarrow ((B \rightarrow A) \wedge B)]$ (16)
 - axiom 3 with B replaced by $B \rightarrow A$
 - and C replaced by B
 - $(A \wedge B) \rightarrow ((B \rightarrow A) \wedge B)$ (17)
 - modus ponens on (15) and (16)
 - $((B \rightarrow A) \wedge B) \rightarrow (B \wedge (B \rightarrow A))$ (18)
 - axiom 2 with A replaced by $B \rightarrow A$
 - $[B \wedge (B \rightarrow A)] \rightarrow A$ (19)
 - axiom 5 with switched A and B
 - $(A \wedge B) \rightarrow A$ — (trans) on (17), (18) and (19)

3. $(A \rightarrow B) \rightarrow B$
 - $(A \wedge B) \rightarrow (B \wedge A)$ — axiom 3 (20)
 - $(B \wedge A) \rightarrow B$ — formula 2 with switched A and B (21)
 - $(A \wedge B) \rightarrow A$ — (trans) on (20) and (21)

4. $A \rightarrow A$
 - $A \rightarrow (A \wedge A)$ — axiom 1 (22)
 - $(A \wedge A) \rightarrow A$ — formula 2 with B replaced by A (23)
 - $A \rightarrow A$ — (trans) on (22) and (23)

5. $B \rightarrow (A \vee B)$
 - $B \rightarrow (B \vee A)$ — axiom 7 with switched A and B (24)
 - $(B \vee A) \rightarrow (A \vee B)$ — axiom 8 with switched A and B (25)
 - $B \rightarrow (A \vee B)$ — (trans) on (24) and (25)

Kripke semantics for the intuitionistic propositional calculus

The intuition behind Kripke semantics for the intuitionistic propositional calculus can be described in terms of a process of knowledge acquisition, where

- worlds of Kripke structures correspond to stages of knowledge acquisition. We require that knowledge reflects facts which are true in a given reality. Therefore, if a formula is known in stage w , then it holds in w , too, i.e., we accept that $\Box A \rightarrow A$, where $\Box A$ is read “ A is known” (thus reflexivity of the accessibility relation will be required)
- the accessibility relation between worlds provides a “temporal ordering” on stages. The ordering is not required to be linear and different branches reflect different ways to acquire knowledge
- atomic formulas true in a given stage represent knowledge acquired so far
- it is assumed that the acquired knowledge is never lost, i.e., a formula true in a given stage w remains true in all stages accessible from w , so that if a formula A is “known” then it is also known in all successive stages (this is reflected by modal axiom $\Box A \rightarrow \Box \Box A$. In consequence, the accessibility relation will be required to be transitive).

By a *Kripke structure for intuitionistic propositional calculus* we mean a Kripke structure in the sense of definition given in slide 65, where the accessibility relation R is reflexive and transitive.

Let $\langle \mathcal{W}, R \rangle$ be a reflexive and transitive Kripke frame and $\langle \mathcal{K}, w, v \rangle$ be a Kripke structure. Then

- $\mathcal{K}, w, v \models A$, where A is a propositional variable, iff for all $z \in \mathcal{W}$ such that $R(w, z)$, we have that $v(A, z) = \text{TRUE}$
- $\mathcal{K}, w, v \models A \wedge B$ iff $\mathcal{K}, w, v \models A$ and $\mathcal{K}, w, v \models B$
- $\mathcal{K}, w, v \models A \vee B$ iff $\mathcal{K}, w, v \models A$ or $\mathcal{K}, w, v \models B$
- $\mathcal{K}, w, v \models A \rightarrow B$ iff for all $z \in \mathcal{W}$ such that $R(w, z)$, we have that $\mathcal{K}, z, v \not\models A$ or $\mathcal{K}, z, v \models B$
- $\mathcal{K}, w, v \models \neg A$ iff for all $z \in \mathcal{W}$ such that $R(w, z)$, we have that $\mathcal{K}, z, v \not\models A$.

Theorem

Heyting's axiomatization of the intuitionistic propositional calculus is sound and complete wrt Kripke semantics given above. \square

Examples

1. $\neg A \vee A$ is not intuitionistically valid.

As a counterexample, consider the Kripke structure, where

- the set of worlds: $\{w, z, y\}$
- the accessibility relation: $R(w, z), R(w, y)$ (plus reflexivity)
- valuation:
 $v(A, w) = \text{TRUE}, v(A, z) = \text{FALSE}, v(A, y) = \text{TRUE}$.

We now have that $\mathcal{K}, w, v \not\models A$ and $\mathcal{K}, w, v \not\models \neg A$.

Thus $\mathcal{K}, w, v \not\models \neg A \vee A$.

2. $\neg \neg A \rightarrow A$ is not intuitionistically valid.

As a counterexample, consider the Kripke structure, where

- the set of worlds: $\{w, z, y\}$
- the accessibility relation: $R(w, z), R(z, y)$ (plus reflexivity and transitivity)
- valuation:
 $v(A, w) = \text{FALSE}, v(A, z) = \text{FALSE}, v(A, y) = \text{TRUE}$.

We now have that $\mathcal{K}, w, v \models \neg \neg A$ and $\mathcal{K}, w, v \not\models A$.

Thus $\mathcal{K}, w, v \not\models \neg \neg A \rightarrow A$.

Translation of the intuitionistic propositional calculus into modal logic

Kripke semantics for the intuitionistic propositional calculus suggests its translation tr into modal logics S4 and S4Grz:

- $tr(A) \stackrel{\text{def}}{=} \Box A$, where A is a propositional variable
- $tr(A \wedge B) \stackrel{\text{def}}{=} tr(A) \wedge tr(B)$
- $tr(A \vee B) \stackrel{\text{def}}{=} tr(A) \vee tr(B)$
- $tr(A \rightarrow B) \stackrel{\text{def}}{=} \Box [tr(A) \rightarrow tr(B)]$
- $tr(\neg A) \stackrel{\text{def}}{=} \Box [\neg tr(A)]$.

If Γ is a set of formulas then $tr(\Gamma) \stackrel{\text{def}}{=} \{tr(A) \mid A \in \Gamma\}$.

Theorem

Let Γ be a set of formulas of the intuitionistic propositional calculus. Then:

1. $\Gamma \models A$ iff $tr(\Gamma) \models_{S4} tr(A)$
2. $\models A$ iff $\models_{S4Grz} tr(A)$. \square

Examples

1. Consider $A \vee \neg A$, where A is a propositional variable. Then

$$\begin{aligned} tr(A \vee \neg A) &= tr(A) \vee tr(\neg A) = \\ &= \Box A \vee \Box [\neg tr(A)] = \\ &= \Box A \vee \Box [\neg \Box A] \end{aligned}$$

— check whether the above formula is valid in S4.

2. Translation of $\neg\neg A \rightarrow A$:

$$\begin{aligned} tr(\neg\neg A \rightarrow A) &= \Box [tr(\neg\neg A) \rightarrow tr(A)] = \\ &= \Box [\Box [\neg tr(\neg A)] \rightarrow \Box A] = \\ &= \Box [\Box [\neg \Box \neg A] \rightarrow \Box A] = \\ &= \Box [\Box \Diamond A \rightarrow \Box A] \end{aligned}$$

3. Translation of $A \rightarrow \neg\neg A$:

$$tr(A \rightarrow \neg\neg A) = \Box [\Box A \rightarrow \Box \Diamond A]$$

— check whether the above formula is valid in S4.

Relation of intuitionistic propositional calculus to the classical logic

Theorem

1. Glivenko (1929)

A formula A is a classical propositional tautology iff the formula $\neg\neg A$ is an intuitionistic propositional tautology.

2. Gödel (1933)

If a formula contains no connectives except \wedge and \neg then it is a classical propositional tautology iff it is an intuitionistic propositional tautology.

3. If a formula contains no connectives except \wedge, \vee and \rightarrow then it is a classical propositional tautology iff it is an intuitionistic propositional tautology.

4. A formula $\neg A$ is a classical propositional tautology iff $\neg A$ is an intuitionistic propositional tautology.

5. A formula $A \rightarrow \neg B$ is a classical propositional tautology iff $A \rightarrow \neg B$ is an intuitionistic propositional tautology.

6. Classical propositional logic =

Heyting's axioms + $\{A \vee \neg A\}$ =

Heyting's axioms + $\{\neg\neg A \rightarrow A\}$ =

Heyting's axioms + $\{(A \rightarrow B) \rightarrow [(\neg A \rightarrow B) \rightarrow B]\}$. \square

Translation of the classical propositional calculus into the intuitionistic propositional calculus

We define translation tr_0 of the classical propositional calculus into the intuitionistic propositional calculus as follows:

- $tr_0(A) \stackrel{\text{def}}{=} A$, where A is a propositional variable
- $tr_0(A \wedge B) \stackrel{\text{def}}{=} tr_0(A) \wedge tr_0(B)$
- $tr_0(A \vee B) \stackrel{\text{def}}{=} \neg[\neg tr_0(A) \wedge \neg tr_0(B)]$
- $tr_0(A \rightarrow B) \stackrel{\text{def}}{=} tr_0(A) \rightarrow tr_0(B)$
- $tr_0(\neg A) \stackrel{\text{def}}{=} \neg tr_0(A)$.

If Γ is a set of formulas then $tr_0(\Gamma) \stackrel{\text{def}}{=} \{tr_0(A) \mid A \in \Gamma\}$.

Theorem (Gentzen 1936)

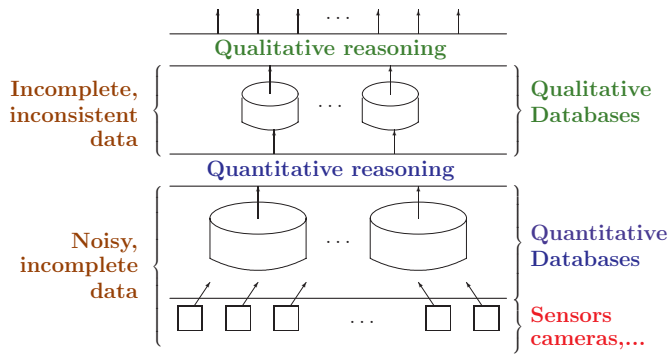
Let Γ be a set of formulas of the classical propositional calculus. Then:

$$\Gamma \models_{\text{classically}} A \text{ iff } tr_0(\Gamma) \models_{\text{intuitionistically}} tr_0(A). \quad \square$$

What you should have learnt from Lecture VIII?

- what is a proof from the perspective of intuitionism?
- Heyting's axiom system for the intuitionistic propositional logic.
- Kripke semantics for the intuitionistic propositional calculus.
- Translation of the intuitionistic propositional calculus into modal logics.
- Relationships between the intuitionistic propositional calculus and the classical logic.
- Translation of the classical propositional calculus into the intuitionistic propositional calculus.

From sensors to qualitative reasoning



Similarity spaces

By a *similarity space* we understand a triple $S = \langle \text{DOM}, f, p \rangle$, where:

- DOM is a nonempty set of objects, called a *domain*
- $f : \text{DOM} \times \text{DOM} \rightarrow [0, 1]$ is a function, called the *similarity function* of S
- $p \in [0, 1]$ is called the *similarity threshold* of S .

Examples

1. Consider cars, where each car is characterized by its speed, measured by a radar. The speed of a car c is given by a function $S(c)$, whose value is in $[0, 200]$.

The perceptual capability of a given agent can be modelled by a similarity space $\Sigma = \langle C, \sigma, 0.9 \rangle$, where C is the set of considered cars and

$$\sigma(c_1, c_2) \stackrel{\text{def}}{=} \frac{|S(c_1) - S(c_2)|}{200}$$

2. Assume that a robot is asked to recognize heaps made of grains of sand. The robot's subjective reality can be modeled in terms of a perception filter given, e.g., by the similarity space $\langle \omega, \sigma', 0.8 \rangle$, where

- ω is the set of natural numbers
- $\sigma'(k, m) \stackrel{\text{def}}{=} 1 - \frac{|k - m|}{\max\{1, k, m\}}$.

The similarity function σ' reflects the intuition that heaps, say, of 1000 and 1001 grains are more similar than heaps of, e.g., 100 and 101 grains.

Neighborhoods and approximations

Let $S = \langle \text{DOM}, \tau, p \rangle$ be a similarity space.

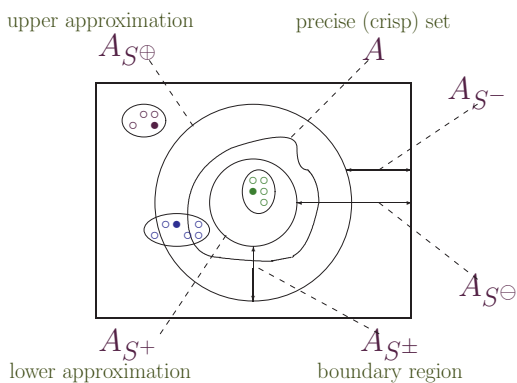
1. By a *neighborhood of u* wrt S , we understand

$$n^S(u) \stackrel{\text{def}}{=} \{v : f(u, v) \geq p\}.$$

2. Let $A \subseteq \text{DOM}$. The *lower and upper approximation of A* wrt S , denoted by A_{S^+} and A_{S^\oplus} , respectively, are defined by

$$A_{S^+} \stackrel{\text{def}}{=} \{u \in \text{DOM} : n^S(u) \subseteq A\},$$

$$A_{S^\oplus} \stackrel{\text{def}}{=} \{u \in \text{DOM} : n^S(u) \cap A \neq \emptyset\}.$$



Let $S = \langle \text{DOM}, f, p \rangle$ be a similarity space. Then $\sigma(u, v) \stackrel{\text{def}}{=} v \in n^S(u)$ is called the *similarity relation induced by S* .

Fact

Let $S = \langle \text{DOM}, f, p \rangle$ be a similarity space and let $A \subseteq \text{DOM}$. Then:

$$A_{S^+} = \{u \in A \mid \forall v [\sigma(u, v) \rightarrow v \in A]\}$$

$$A_{S^\oplus} = \{u \in A \mid \exists v [\sigma(u, v) \wedge v \in A]\}.$$

□

Correspondences between approximations and similarity

There is a close relationship between correspondences between approximations and similarities and those considered in modal correspondence theory, when:

- modal \Box is considered as A_{S^+}
- modal \Diamond is considered as A_{S^\oplus} .

Examples of correspondences

Let S be a similarity space and σ be the similarity relation induced by S . Then:

1. inclusion $A_{S^+} \subseteq A_{S^\oplus}$ is equivalent to the requirement of seriality of σ ($\forall x \exists z \sigma(x, z)$)
2. inclusion $A_{S^+} \subseteq A$ is equivalent to the requirement of reflexivity of σ ($\forall x \sigma(x, x)$)
3. inclusion $A \subseteq (A_{S^\oplus})_{S^+}$ is equivalent the requirement of symmetry of σ ($\forall x, y [\sigma(x, y) \rightarrow \sigma(y, x)]$)
4. inclusion $A_{S^+} \subseteq (A_{S^+})_{S^+}$ is equivalent to the requirement of transitivity of σ ($\forall x, z, u [(\sigma(x, z) \wedge \sigma(z, u)) \rightarrow \sigma(x, u)]$).

Exercises

Provide examples of reasonable similarity relation, which is:

1. non-transitive
2. non-symmetric
3. non-reflexive.

Approximate database considerations

Based on the results above, it is important that the use of an approximate database is consistent with the approximation and similarity constraints required by the particular application.

The approximation and similarity constraints play the role of integrity constraints in crisp databases.
Yet, enforcing these constraints is not as straightforward.

Example: rough databases

In rough set theory σ is required to be an equivalence relation.

We then have to enforce that:

1. $A_{S^+} \subseteq A$ (reflexivity)
2. $A \subseteq (A_{S^\oplus})_{S^+}$ (symmetry)
3. $A_{S^+} \subseteq (A_{S^+})_{S^+}$ (transitivity).

Conditions 1-2 are not representable by database integrity constraints, since the crisp set A is **not explicitly represented in the database**.

Meta-constraints

One often needs to enforce *meta-constraints* that have to be ensured by database designers, and which cannot explicitly be represented or computed in an approximate database.

Sometimes one can apply known facts from the area of modal logics, e.g., axiom 5:

$$A_{S^\oplus} \rightarrow (A_{S^\oplus})_{S^+},$$

can replace both reflexivity and transitivity, since $KT5$ is $S5$.

Axiom 5 is expressed by referring to approximations only!

Weakest sufficient and strongest necessary conditions

By a *sufficient condition of a formula α on the set of relation symbols P under theory T* we shall understand any formula ϕ containing only symbols in P such that

$$T \models \phi \rightarrow \alpha.$$

It is the *weakest sufficient condition*, denoted by $WSC(\alpha; T; P)$ if, additionally, for any sufficient condition ψ of α on P under T , $T \models \psi \rightarrow \phi$ holds.

Fact

$WSC(\alpha; T; P) \equiv \forall \bar{\Phi}. [T \rightarrow \alpha]$, where $\bar{\Phi}$ consists of relations to be projected out (outside of P). \square

By a *necessary condition of a formula α on the set of relation symbols P under theory T* we shall understand any formula ϕ containing only symbols in P such that

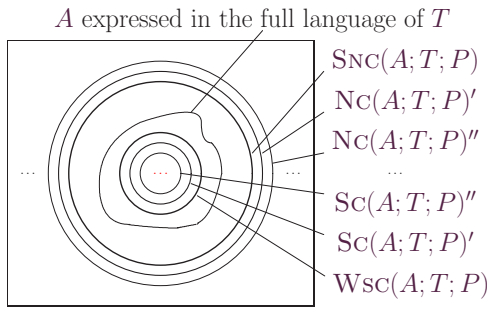
$$T \models \alpha \rightarrow \phi.$$

It is the *strongest necessary condition*, denoted by $SNC(\alpha; T; P)$ if, additionally, for any necessary condition ψ of α on P under T , $T \models \phi \rightarrow \psi$ holds.

Fact

$SNC(\alpha; T; P) \equiv \exists \bar{\Phi}. [T \wedge \alpha]$, where $\bar{\Phi}$ consists of relations to be projected out (outside of P). \square

Snc's and Wsc's as approximations



Snc's and Wsc's are as strong as rough approximations

Fact

Let $A_{(T,P)^+} \stackrel{\text{def}}{=} \text{WSC}(A; T; P)$ and $A_{(T,P)^{\oplus}} \stackrel{\text{def}}{=} \text{SNC}(A; T; P)$.

Then:

1. $A_{(T,P)^+} \rightarrow A$ -reflexivity
2. $A \rightarrow (A_{(T,P)^{\oplus}})_{(T,P)^+}$ -symmetry
3. $A_{(T,P)^+} \rightarrow (A_{(T,P)^+})_{(T,P)^+}$ -transitivity. \square

Applications of Snc's and Wsc's

1. hypotheses generation and abduction (via Wsc's)
2. theory approximation (knowledge compilation)
3. security in knowledge databases (how much can one discover about hidden relations)
4. building communication interfaces between agents using heterogenous ontologies
5. computing weakest preconditions and strongest postconditions for actions
6. reasoning with data sets reduced by dropping some attributes.

Example: theory approximation

The concept of approximating more complex theories by simpler theories has been studied mainly in the context of approximating arbitrary propositional theories by propositional Horn clauses. Note that strongest necessary and weakest sufficient conditions relativized to a subset of relation symbols provide us with approximations of theories expressed in a richer language by theories expressed in a less expressive language.

Consider the following theory, denoted by T :

- (26) $(\text{CompSci} \wedge \text{Phil} \wedge \text{Psych}) \rightarrow \text{CogSci}$
- (27) $\text{ReadsMcCarthy} \rightarrow (\text{CompSci} \vee \text{CogSci})$
- (28) $\text{ReadsDennett} \rightarrow (\text{Phil} \vee \text{CogSci})$
- (29) $\text{ReadsKosslyn} \rightarrow (\text{Psych} \vee \text{CogSci})$.

Reasoning with this theory is quite complicated due to the large number of cases. On the other hand, one would like to check, for instance, whether a computer scientist who reads Dennett and Kosslyn is also a cognitive scientist.

One can substantially reduce the theory and make the reasoning more efficient.

In the first step one notices that Phil and Psych are not used in the query, thus they might appear redundant in the reasoning process. On the other hand, these notions appear in disjunctions in clauses (28) and (29). In this context we might consider

$$\text{SNC}(\text{CompSci} \wedge \text{ReadsDennett} \wedge \text{ReadsKosslyn}; T; -\{\text{Phil}, \text{Psych}\}) \tag{30}$$

where, as usual, $-\{\text{Phil}, \text{Psych}\}$ denotes all symbols in the language, other than Phil and Psych . Performing simple calculations one obtains the following formula equivalent to (30):

$$\begin{aligned} (27) \wedge [\text{CompSci} \wedge \text{ReadsDennett} \wedge \text{ReadsKosslyn}] \\ \wedge [(\text{CompSci} \wedge (\text{ReadsDennett} \wedge \neg \text{CogSci}) \\ \wedge (\text{ReadsKosslyn} \wedge \neg \text{CogSci})) \rightarrow \text{CogSci}] \end{aligned} \tag{31}$$

which easily reduces to

$$(27) \wedge \text{CompSci} \wedge \text{ReadsDennett} \wedge \text{ReadsKosslyn} \wedge (\neg \text{CogSci} \rightarrow \text{CogSci}). \tag{32}$$

Thus the strongest necessary condition for the original formula is

$$\text{CompSci} \wedge \text{ReadsDennett} \wedge \text{ReadsKosslyn}$$

which implies CogSci and, consequently, the formula also implies CogSci .

Assume that one wants to compute the weakest sufficient condition of being a computer scientist in terms of

$$\{ReadsDennett, ReadsKosslyn, ReadsMcCarthy, CogSci\}.$$

We then consider

$$WSC(CompSci; T; -\{Phil, Psych, CompSci\}). \quad (33)$$

After eliminating quantifiers over *Phil*, *Psych*, *CompSci* from the second-order formulation of the weakest sufficient condition, one obtains the following formula equivalent to (33):

$$ReadsMcCarthy \wedge \neg CogSci.$$

Thus the weakest condition that, together with theory *T*, guarantees that a person is a computer scientist is that the person reads McCarthy and is not a cognitive scientist.

Example: abduction

Abduction is an explanation of facts that are observed, but do not follow from a given theory about the world. The weakest sufficient condition corresponds to the weakest abduction.

Consider the theory

$$T = \{\forall x.[HasWheels(x) \rightarrow CanMove(x)], \\ \forall x.[Car(x) \rightarrow HasWheels(x)]\}.$$

Assume one wants to check what could be the reason for an object to be able to move. There are three interesting cases:

1. the target language is $\{HasWheels\}$; and we consider

$$WSC(CanMove(x); T; \{HasWheels\}),$$

which is equivalent to $\forall CanMove, Car.[T \rightarrow CanMove(x)]$

2. the target language is $\{Car\}$ and we consider

$$WSC(CanMove(x); T; \{Car\}),$$

which is equivalent to $\forall HasWheels, Car.[T \rightarrow CanMove(x)]$

3. the target language is $\{HasWheels, Car\}$ and we consider

$$WSC(CanMove(x); T; \{HasWheels, Car\}),$$

which is equivalent to $\forall CanMove.[T \rightarrow CanMove(x)]$.

We then obtain the following results:

1. $WSC(CanMove(x); T; \{HasWheels\}) \equiv HasWheels(x)$
2. $WSC(CanMove(x); T; \{Car\}) \equiv Car(x)$
3. $WSC(CanMove(x); T; \{HasWheels, Car\}) \equiv \\ \forall x.[Car(x) \rightarrow HasWheels(x)] \rightarrow HasWheels(x).$

The first two conditions are rather obvious.

The third one might seem a bit strange, but observe that

$$\forall x.[Car(x) \rightarrow HasWheels(x)]$$

is an axiom of theory *T*.

Thus, in the third case, after simplification we have

$$WSC(CanMove(x); T; \{HasWheels, Car\}) \equiv \\ HasWheels(x).$$

Example: agents using heterogeneous ontologies

Assume an agent *A* wants to obtain from agent *B* the list of all persons *x* such that $Q(x) \stackrel{\text{def}}{=} High(x) \wedge Silny(x)$ holds.

Assume that both agents know the terms

$$L = \{High, Muscular, Sound\},$$

and that agent *B* does not know the term *Silny*.

(In Polish “*Silny*” means “*Strong*”, but the database agent doesn’t know the Polish language.)

Suppose that *A* lives in a world satisfying the conditions:

$$Th \stackrel{\text{def}}{=} \left\{ \forall x.[Silny(x) \rightarrow Sound(x)], \\ \forall x.[Muscular(x) \rightarrow Silny(x)] \right\}.$$

The best approximation of the original query *Q*, under given assumptions, is:

$$Q_{(Th,L)^+}(x) \equiv WSC(High(x) \wedge Silny(x); Th; L) \\ Q_{(Th,L)^\oplus}(x) \equiv SNC(High(x) \wedge Silny(x); Th; L)$$

According to characterizations of weakest sufficient and strongest necessary conditions provided in slide 136, these conditions are equivalent to:

$$Q_{(Th,L)^+}(x) \equiv [High(x) \wedge Muscular(x)] \\ Q_{(Th,L)^\oplus}(x) \equiv [High(x) \wedge Sound(x)].$$

Tolerance spaces

Tolerance spaces are similarity spaces, where the similarity function is required to satisfy

$$\forall x [f(x, x) = 1]$$

$$\forall x, y [f(x, y) = f(y, x)].$$

Fact

The induced similarity relation of any tolerance space is reflexive and symmetric. \square

In the case of tolerance spaces similarity function is then required to satisfy conditions:

1. $A_{S^+} \subseteq A$ (reflexivity)
2. $A \subseteq (A_{S^{\oplus}})_{S^+}$ (symmetry).

What you should have learnt from Lecture IX?

- What are similarity spaces?
- What are neighborhoods?
- What are approximations?
- What are correspondences between similarities and approximations?
- What are weakest sufficient and strongest necessary conditions?
- What are applications of weakest sufficient and strongest necessary conditions?
- What is abduction?
- What are tolerance spaces?

Non-monotonic reasoning

Traditional logics are monotonic, i.e., adding new premises (axioms) will never invalidate previously inferred conclusions (theorems), or, equivalently, the set of conclusions non-decreases monotonically with the set of premises.

A logic is *monotonic* iff it satisfies the condition that for any sets of premises S and S' ,

$$S \subseteq S' \text{ implies } Cn(S) \subseteq Cn(S'),$$

where Cn denotes the semantic consequence operator of a given logic, i.e., the operator which to each set of formulas S assigns the set $Cn(S)$ of all formulas which are consequences of S in the logic.

Logics that are not monotonic are called *non-monotonic*.

Example

Assume that you are planning to make a trip by car. To begin with, you have to decide where your car actually is. Given no evidence to the contrary, it is reasonable to conclude that your car is located where you last parked it. Of course, the above conclusion may turn out to be incorrect. The car may have been towed away or stolen, but such instances would be uncommon. In order to plan and act, it is essential that such weak conclusions can be drawn.

Non-monotonic formalisms: an overview

There are two basic types of non-monotonic reasoning: default reasoning and autoepistemic reasoning.

By the *default reasoning* we mean the drawing of a rational conclusion, from less than conclusive information, in the absence of evidence leading to the contrary.

By *autoepistemic reasoning*, we understand reaching a conclusion, from an incomplete representation of complete information, under the assumption that if the conclusion were false, its negation would be explicitly represented.

Examples

1. A classical example of default reasoning is the rule stating: "In the absence of evidence to the contrary, assume that a bird flies."
2. A typical example of autoepistemic reasoning is the rule stating: "If your name is not on a list of winners, assume that you are a loser." The motivation for using this rule follows from the observation that the number of losers is usually much greater than the number of winners, so explicitly keeping all the losers would be impractical.

Default logic

By a *default rule* we understand

$$\frac{A(\bar{x}) : B(\bar{x})}{C(\bar{x})} \quad (34)$$

where $A(\bar{x})$, $B(\bar{x})$ and $C(\bar{x})$ are first-order formulas whose free variables are among those of \bar{x} . In rule (34):

- $A(\bar{x})$ is called the *prerequisite*
- $B(\bar{x})$ is called the *justification*
- $C(\bar{x})$ is called the *consequent*

of the default.

If $A(\bar{x}) = \text{TRUE}$, the default is called *prerequisite-free* and it is written as

$$\frac{:B(\bar{x})}{C(\bar{x})}$$

The default (34) has the following interpretation: for all individuals represented by \bar{x} , if $A(\bar{x})$ is sure and $B(\bar{x})$ is possible, then $C(\bar{x})$ is assumed to be sure.

Normal defaults are of the form

$$\frac{A(\bar{x}) : B(\bar{x})}{B(\bar{x})}$$

and *semi-normal defaults* are of the form

$$\frac{A(\bar{x}) : B(\bar{x}) \wedge C(\bar{x})}{B(\bar{x})}$$

Examples

1. Consider a fact that birds usually fly. This can be expressed as a rule stating that “in the absence of evidence to the contrary, assume about any particular bird that it flies”:

$$\frac{Bird(x) : Flies(x)}{Flies(x)}$$

(this is a normal default).

2. What do the following default rules express?

$$\frac{Republican(x) : \neg Pacifist(x)}{\neg Pacifist(x)}$$

$$\frac{Quaker(x) : Pacifist(x)}{Pacifist(x)}$$

(these are normal defaults).

3. The following semi-normal default, expresses the property that “typically, an adult who is not a high school dropout is employed”:

$$\frac{Adult(x) : Employed(x) \wedge \neg Dropout(x)}{Employed(x)}$$

Default theories

A *default theory* is a pair $T = \langle W, D \rangle$, where W is a set of first-order sentences, *axioms* of T , and D is a set of defaults.

Intuitively, a default theory is viewed as a representation of one or more aspects of the world under consideration. The axioms in a theory are intended to contain all information known to be true. The default rules extend this information by supporting plausible conclusions.

A set of beliefs about the world represented by a theory T is called an *extension* of T , defined as follows:

for any set of sentences S , let $\Gamma(S)$ be the smallest set of sentences satisfying the following properties:

1. $\Gamma(S) = Cn(\Gamma(S))$
2. $W \subseteq \Gamma(S)$
3. If $\frac{A : B}{C} \in D$, $A \in \Gamma(S)$ and $\neg B \notin \Gamma(S)$, then $C \in \Gamma(S)$.

A set of sentences E is an *extension* of T iff $E = \Gamma(E)$, i.e., iff E is a fixed point of Γ .

Approximations and default rules

Reasoning with default theories as defined before is not tractable. In the presence of a similarity space, say S , a tractable semantics can be provided. Namely, a default rule of the form (34) can be understood as:

$$\frac{A_{S^+}(\bar{x}) : B_{S^\oplus}(\bar{x})}{C_{S^+}(\bar{x})},$$

meaning that if $A(\bar{x})$ is sure and $B(\bar{x})$ is possible then conclude that $C(\bar{x})$ is sure.

Examples

1. If a fruit is yellow and might be a lemon then conclude that it is a lemon:

$$\frac{yellow_{S^+}(x) : lemon_{S^\oplus}(x)}{lemon_{S^+}(x)}$$

2. If a car's speed is classified to be high and the road might be slippery then conclude that the situation is surely dangerous:

$$\frac{highSpeed_{S^+}(x) : slippery_{S^\oplus}(x)}{danger_{S^+}(x)}$$

Circumscription

Circumscription (McCarthy, 1980) is a powerful non-monotonic formalism centered around the idea: the objects (tuples of objects) that can be shown to satisfy a certain relation or property are conjectured to be *all* the objects (tuples of objects) satisfying that relation or property.

Examples

1. Consider a phone book. It represents a relation

$tel(person, number)$.

We assume that those items which are in the phone book satisfy relation tel and, moreover, that these are *all* items satisfying tel .

2. More generally, circumscription can be considered as a generalization of the closed world assumption in databases (a database table in relational databases are assumed to contain all facts true about a given reality).

Some definitions

Let \mathcal{L} be a fixed first-order language with equality. By *theories* we understand here finite sets of sentences stated in \mathcal{L} . Since each such set is equivalent to the conjunction of its members, a circumscriptive theory may always be viewed as a single sentence.

If A and B are relation expressions of the same arity, then

$$A \leq B \stackrel{\text{def}}{=} \forall \bar{x}. (A(\bar{x}) \rightarrow B(\bar{x}))$$

$$\bar{A} \leq \bar{B} \stackrel{\text{def}}{=} \bigwedge_{i=1}^n [A_i \leq B_i]$$

$$\bar{A} = \bar{B} \stackrel{\text{def}}{=} (\bar{A} \leq \bar{B}) \wedge (\bar{B} \leq \bar{A})$$

$$\bar{A} < \bar{B} \stackrel{\text{def}}{=} (\bar{A} \leq \bar{B}) \wedge \neg(\bar{B} \leq \bar{A})$$

Definition of circumscription

Let $\bar{P} = \langle P_1, \dots, P_n \rangle$, $\bar{S} = \langle S_1, \dots, S_m \rangle$ be disjoint tuples of distinct relation symbols (the language does not have to be restricted to those symbols only).

Let $T(\bar{P}, \bar{S})$ be a theory.

The *circumscription* of \bar{P} in $T(\bar{P}, \bar{S})$ with varied \bar{S} , written $CIRC(T; \bar{P}; \bar{S})$, is the sentence

$$T(\bar{P}, \bar{S}) \wedge \forall \bar{X} \forall \bar{Y}. \{ [T(\bar{X}, \bar{Y}) \wedge [\bar{X} \leq \bar{P}]] \rightarrow [\bar{P} \leq \bar{X}] \},$$

which is an abbreviation for

$$T(\bar{P}, \bar{S}) \wedge \forall \bar{X} \forall \bar{Y}. \left\{ \left[T(\bar{X}, \bar{Y}) \wedge \bigwedge_{i=1}^n \forall \bar{x}. (X_i(\bar{x}) \rightarrow P_i(\bar{x})) \right] \rightarrow \bigwedge_{i=1}^n \forall \bar{x}. (P_i(\bar{x}) \rightarrow X_i(\bar{x})) \right\}.$$

Remark

Observe that the meaning of $CIRC(T; \bar{P}; \bar{S})$ is that relations \bar{P} are minimized (wrt inclusion) where relations \bar{S} can be varied.

Abnormality theories

Frequently one attempts to formalize “normal” behavior or situations. In the circumscription based approach one formalizes “abnormalities” and then minimizes those abnormalities.

Examples

1. “Those who work usually have income”:

$$\forall x \left[(works(x) \wedge \neg Ab_w(x)) \rightarrow hasIncome(x) \right]$$

2. “In a normal situation people escape from fire”:

$$\forall s, x \left[(fire(s) \wedge \neg Ab_f(x, s)) \rightarrow escapes(x, s) \right]$$

3. “In typical situations a car is in the place where it has been parked”:

$$\forall s, p, c \left[(parked(c, p) \wedge \neg Ab(s, c)) \rightarrow isIn(c, p) \right].$$

Example: birds usually fly

Let T consist of the following formulas:

$$\begin{aligned} & Bird(\mathbf{Tweety}) \\ & \forall x. [(Bird(x) \wedge \neg Ab(x)) \rightarrow Flies(x)]. \end{aligned}$$

Let $\bar{P} = \langle Ab \rangle$ and $\bar{S} = \langle Flies \rangle$.

$$\begin{aligned} CIRC(T; \bar{P}; \bar{S}) &= T(\bar{P}, \bar{S}) \wedge \\ & \forall X \forall Y. \{ [Bird(\mathbf{Tweety}) \wedge \forall x. [(Bird(x) \wedge \neg X(x)) \rightarrow Y(x)] \wedge \\ & \forall x. [X(x) \rightarrow Ab(x)]] \rightarrow \forall x. [Ab(x) \rightarrow X(x)] \}. \end{aligned}$$

In its basic form, the idea of circumscriptive reasoning is either to eliminate second-order quantifiers or to find relational expressions for X and Y that when substituted into the theory T , will result in strengthening the theory so additional inferences can be made.

For example, substituting $\lambda x. \text{FALSE}$ for X and $\lambda x. Bird(x)$ for Y , one can conclude that

$$CIRC(T; \bar{P}; \bar{S}) \models T \wedge A \quad (35)$$

where A is

$$\begin{aligned} & \{ \forall x. [(Bird(x) \wedge \neg \text{FALSE}) \rightarrow Bird(x)] \wedge \\ & \forall x. [\text{FALSE} \rightarrow Ab(x)] \} \rightarrow \forall x. [Ab(x) \rightarrow \text{FALSE}]. \end{aligned}$$

Since A can be simplified to the logically equivalent sentence

$$\forall x. [Ab(x) \rightarrow \text{FALSE}],$$

which in turn is equivalent to $\forall x. \neg Ab(x)$, one can infer by (35) that

$$CIRC(T; \bar{P}; \bar{S}) \models Flies(\mathbf{Tweety}).$$

Semantics of circumscription

Given a relational structure

$$M = \langle \text{DOM}, \{f_i^{\text{DOM}} : i \in I\}, \{R_j^{\text{DOM}} : j \in J\} \rangle,$$

we write $\text{DOM}(M)$ to denote the domain of M . If R_j is a relation symbol, then $M(R_j)$ stands for R_j^{DOM} .

The class of all models of a theory T is denoted by $\text{Mod}(T)$.

Let $\bar{P}, \bar{S}, T(\bar{P}, \bar{S})$ be as in slide 155.

Let M, N be models of T . We say that M is a (\bar{P}, \bar{S}) -submodel of N , written $M \leq^{(\bar{P}, \bar{S})} N$, iff

1. $\text{DOM}(M) = \text{DOM}(N)$
2. $M(R) = N(R)$, for any relation symbol R not in $\bar{P} \cup \bar{S}$
3. $M(R) \subseteq N(R)$, for any relation symbol R in \bar{P} .

We write $M <^{(\bar{P}, \bar{S})} N$ iff $M \leq^{(\bar{P}, \bar{S})} N$, but not $N \leq^{(\bar{P}, \bar{S})} M$.

A model M of T is (\bar{P}, \bar{S}) -minimal if and only if T has no model N such that $N <^{(\bar{P}, \bar{S})} M$.

We write $\text{Mod}^{(\bar{P}, \bar{S})}(T)$ to denote the class of all (\bar{P}, \bar{S}) -minimal models of T .

Theorem For any T, \bar{P} and \bar{S} ,

$$\text{Mod}(CIRC(T; \bar{P}; \bar{S})) = \text{Mod}^{(\bar{P}, \bar{S})}(T).$$

Example: birds usually fly revisited

Let us reconsider the theory T from slide 157:

$$\begin{aligned} & Bird(\mathbf{Tweety}) \\ & \forall x. [(Bird(x) \wedge \neg Ab(x)) \rightarrow Flies(x)]. \end{aligned}$$

It should be emphasized that the sentence $Flies(\mathbf{Tweety})$ is not derivable using circumscription if the relation symbol $Flies$ is not varied. That is

$$CIRC(T; Ab; \langle \rangle) \not\models Flies(\mathbf{Tweety}).$$

To understand the reason why, consider a relational structure M such that:

- $\text{DOM}(M) = \{\mathbf{Tweety}\}$
- $M(Bird) = M(Ab) = \{\mathbf{Tweety}\}$
- $M(Flies) = \emptyset$.

It is clear that M is a model of T .

However, M is also an $(Ab, \langle \rangle)$ -minimal model of T (we cannot make $M(Ab)$ smaller while preserving the truth of T and $M(Flies)$).

Thus, since $Flies(\mathbf{Tweety})$ is false in M , this formula cannot be derived by circumscribing Ab in T without varying relations (in view of Theorem from slide 158).

Example: Nixon diamond

Let T consist of the following:

$$\begin{aligned} & R(\mathbf{Nixon}) \wedge Q(\mathbf{Nixon}) \\ & \forall x. [(R(x) \wedge \neg Ab_1(x)) \rightarrow \neg P(x)] \\ & \forall x. [(Q(x) \wedge \neg Ab_2(x)) \rightarrow P(x)]. \end{aligned}$$

This is a well-known “Nixon diamond” theory with R, P, Q standing for *Republican, Pacifist* and *Quaker*, respectively.

Observe that we use two abnormality relations here, namely Ab_1 and Ab_2 . This is because being abnormal with respect to pacifism as a quaker is a different notion than being abnormal with respect to pacifism as a republican. Let M and N be models of T such that

$$\begin{aligned} \text{DOM}(M) &= \text{DOM}(N) = \{\mathbf{Nixon}\}, \\ M(R) &= N(R) = \{\mathbf{Nixon}\}, \quad M(Q) = N(Q) = \{\mathbf{Nixon}\} \end{aligned}$$

and

$$\begin{aligned} M(P) &= \{\mathbf{Nixon}\} & N(P) &= \emptyset \\ M(Ab_1) &= \{\mathbf{Nixon}\} & N(Ab_1) &= \emptyset \\ M(Ab_2) &= \emptyset & N(Ab_2) &= \{\mathbf{Nixon}\}. \end{aligned}$$

For any \bar{S} , both M and N are (\bar{Ab}, \bar{S}) -minimal models of T , where $\bar{Ab} = \langle Ab_1, Ab_2 \rangle$. Moreover, $M \models P(\mathbf{Nixon})$ and $N \models \neg P(\mathbf{Nixon})$. Therefore it can not be inferred whether \mathbf{Nixon} is a pacifist or not.

The best that can be obtained is the disjunction

$$\neg Ab_1(\mathbf{Nixon}) \vee \neg Ab_2(\mathbf{Nixon}),$$

stating that \mathbf{Nixon} is either normal as a republican or as a quaker.

Autoepistemic reasoning

Autoepistemic logic (Moore, 1985) is built over a base logic (e.g., the classical propositional or first-order logic) by adding modality \Box to the language. The semantics of modality \Box is usually given by modal logics S5 or K45.

Typical forms of autoepistemic reasoning are based on rules “if a fact A is not known then conclude that $\neg A$ holds”, i.e.,

$$\neg\Box A \rightarrow \neg A$$

— this is not an axiom of the logic (that would lead to the trivial modal logic), but is accepted for certain formulas A .

Examples

1. Recall the autoepistemic rule “if x is not on a list of winners, assume that x is a loser.”. It can be expressed as:

$$\neg\Box\text{winner}(x) \rightarrow \neg\text{winner}(x),$$

where $\Box\text{winner}(x) = \text{TRUE}$ when x is on the list of winners.

2. Consider “if x does not know that (s)he has a sister, conclude that x has no sister” (otherwise x would know about it):

$$\neg\Box\text{hasSister}(x) \rightarrow \neg\text{hasSister}(x).$$

What you should have learnt from Lecture X?

- What are non-monotonic logics?
- What is default and autoepistemic reasoning?
- What is circumscription?
- What are abnormality theories?
- What is autoepistemic logic?

Index

3-valued logic, 82
 of Bočvar, 92
 of Hallden, 94
 of Kleene, 90
 of Lukasiewicz, 83
 4-valued logic
 of Belnap, 99
 abduction, 142
 abnormality theory, 156
 accept all, 102, 103
 accessibility relation, 65
 actual world, 63, 65
 adjunction, 118
 alethic logic, 55
 approximation
 lower, 131
 ordering, 100
 upper, 131
 atomic
 query, 27
 autoepistemic
 logic, 161
 reasoning, 148, 161
 auxiliary symbol, 7
 axiom, 15
 4, 60
 5, 60

B, 60
 D, 60
 Dum, 60
 E, 60
 G, 60
 Grz, 60
 H, 60
 M, 60
 T, 60
 Tr, 60
 W, 60

Barcan, 74
 formula, 74, 75
 Belnap, 98, 99
 bilattice, 103
 product, 104
 Bočvar, 92
 body of a rule, 28
 boundary region, 131
 Brouwer, 61, 62, 117

circumscription, 153, 155
 closed world assumption, 26
 closure ordinal, 40
 complete proof system, 19
 completeness, 19
 complexity, 33

class of queries, 33
 conceptual models for quantified
 modal logics, 72
 connective, 7
 consensus, 102, 103
 consequence, 2
 relation, 17
 consequent of a default, 149
 constant, 7
 converse of Barcan formula, 75
 correct
 argument, 2
 proof system, 19
 reasoning, 2
 correctness, 19
 correspondence, 66
 CWA, 26

data complexity, 33
 of Datalog queries, 35, 50
 of first-order queries, 34
 database, 21, 24
 Datalog, 25
 query, 30
 DCA, 26
 deductive database, 22
 default
 logic, 149
 reasoning, 148
 rule, 149

theory, 151
 deontic logic, 55
 deontic variant of S4, 62
 deontic variant of S5, 62
 deontic variant of T, 62
 designated truth values, 98
 domain
 closure axiom, 26
 dual
 fixpoint operator, 42
 modality, 54
 Dummett, 61

epistemic logic, 55
 expression complexity, 33
 of Datalog queries, 35
 of first-order queries, 35
 extension of a default theory, 151
 extensional
 database, 22
 relation, 52
 external negation, 96

fact, 22
 in Datalog, 27
 Feys, 61, 62
 first-order
 logic, 10
 variable, 10
 fixed domains, 74
 fixpoint

calculus, 45
 logic, 45
 of a mapping, 37
 formula, 5, 9
 free logic, 71
 minimal, 71
 function symbol, 7
 fuzzy
 reasoning, 114
 set, 110
 Gödel, 62, 98, 109, 126
 Geach, 61
 Gentzen, 127
 GFP, 37
 Glivenko, 126
 greatest fixpoint, 37
 ground
 formula, 9
 term, 8
 Grzegorzcyk, 62
 Guillaume, 63
 Hallden, 94
 head of a rule, 28
 Heyting, 117, 118
 Hintikka, 61, 63
 indefinite, 90
 individual
 constant, 7
 variable, 10
 information ordering, 100
 integrity constraint, 22
 intensional
 database, 22
 logics, 54
 relation, 52
 internal negation, 96
 interpretation, 12
 intuitionism, 117
 intuitionistic propositional calculus, 118
 join, 103
 justification, 149
 Kanger, 63
 Kleene, 90, 98
 Knaster, 39
 Knaster-Tarski Theorem, 39
 knowledge ordering, 100
 Kolmogorov, 117
 Kripke, 63
 frame, 65
 semantics, 63, 65
 for L_3 , 87
 structure, 65
 for intuitionistic propositional calculus, 122
 for L_3 , 87

L-valuation, 107
 Löb, 61, 62
 language, 5
 lattice, 103
 least fixpoint, 37
 Lemmon, 59
 Lewis, 61
 LFP, 37
 linguistic modifier, 113
 literal, 43
 logic, 2, 17
 alethic, 55
 deontic, 55
 epistemic, 55
 G_n , 109
 G_∞ , 109
 K, 59
 K_3 , 90
 K45, 62
 KD, 62
 KD4, 62
 L_3 , 83
 L_n , 108
 L_{\aleph_0} , 108
 L_{\aleph} , 108
 of Belnap, 99
 of Bočvar, 92
 of Brouwer, 62
 of Grzegorzcyk, 62
 of Hallden, 94
 of Kleene, 90
 of Löb, 62
 of Prior, 62
 of provability, 55
 of Lukasiewicz, 83, 107
 Q1, 74
 Q1-S, 74
 Q1R, 76
 Q1R-S, 76
 Q2, 77
 QC, 77
 S4, 62
 S4.1, 62
 S4.2, 62
 S4.3, 62
 S5, 62
 semantical definition, 4
 syntactical definition, 4
 T, 62
 temporal, 55
 logical
 constant, 7
 language, 4, 5
 operators, 7
 logics of programs, 55
 lower approximation, 131
 Lukasiewicz, 83, 98, 107

many-valued
 logic, 98
 many-valued logic
 of Gödel, 109
 of Lukasiewicz, 107
 McCarthy, 153
 McKinsey, 61
 meaning, 12
 meaningless, 94
 meet, 103
 membership function, 110
 meta-property, 19
 minimal
 free logic, 71
 model, 158
 modal
 logic, 54
 operators, 54
 modality, 54
 modus ponens, 118
 monotone
 fixpoint logic, 45
 formula wrt an atomic formula, 43
 mapping, 38
 monotonic logic, 147
 Moore, 161
 multimodal logics, 67
 n-th order
 logic, 10
 variable, 10
 necessary condition, 136
 negation
 external, 96
 internal, 96
 negative
 formula
 wrt an atomic formula, 43
 literal, 43
 neighborhood, 131
 nested domains, 76
 neutral, 83
 non-monotonic
 logic, 147
 reasoning, 147
 nonsense, 92
 normal default, 149
 objectual
 interpretation, 76
 models for quantified modal logics, 72
 objectual interpretation, 76
 overdefined world, 102
 positive
 fixpoint logic, 45
 formula
 wrt an atomic formula, 43
 literal, 43

possible world, 63
 Post, 98
 POW, 37
 powerset, 37
 prerequisite, 149
 prerequisite-free default, 149
 principle of nested domains, 75
 Prior, 62
 proof
 rule, 15
 system, 15
 propositional
 logic, 10
 modal logic, 54
 variable, 10
 provability logic, 55
 quantified modal logic, 69
 query recognition problem, 33
 relation symbol, 7
 rigid terms semantics, 73
 rule, 15
 in Datalog, 28
 second-order
 logic, 10
 variable, 10
 semantical approach, 12, 13, 56
 semi-normal
 default, 149
 Sheffer, 88
 stroke, 88
 similarity
 function, 129
 space, 129
 threshold, 129
 SNC, 136
 sound
 argument, 2
 proof system, 19
 soundness, 19
 strong Kleene logic, 90
 strongest necessary condition, 136
 submodel, 158
 sufficient condition, 136
 syntactical approach, 12, 15, 56
 Tarski, 39, 107
 temporal logic, 55
 of programs, 78
 term, 8
 theorem
 of Gödel, 126
 of Gentzen, 127
 of Glivenko, 126
 of Knaster and Tarski, 39
 of Wajsberg, 86
 theory, 154
 third-order
 logic, 10

variable, 10
tolerance space, 145
trivial modal logic, 62
truth ordering, 100
truth value, 17
UNA, 26
underdefined world, 102
unique names axiom, 26
upper approximation, 131
variable, 7
 propositional, 10
varied relation symbol, 155
Von Wright, 62
Wajsberg, 86
weakest sufficient condition, 136
Webb, 88
Wesselkamper, 89
 operator, 89
world, 65
 actual, 63
 overdefined, 102
 possible, 63
 underdefined, 102
world-relative model, 76
WSC, 136
Zadeh, 98, 110
zero-order
 logic, 10