Knowledge-Based Systems and Deductive Databases

Wolf-Tilo Balke
Christoph Lofi
Institut für Informationssysteme
Technische Universität Braunschweig
http://www.ifis.cs.tu-bs.de
3. Models

3.1 Logical Models
3.2 Deductive Systems
3.3 Horn Clauses
3.0 Summary of Last Lecture

- Short summary from last lecture
  - **Language** $\mathcal{L} = (\Gamma, \Omega, \Pi, \chi)$
    - $\Gamma$ constant symbols
    - $\Omega$ function symbols
    - $\Pi$ predicate symbols
    - $\chi$ variable symbols
  - Languages are **only syntax** and have absolutely no meaning.
  - Further building blocks of languages are **terms**
    - Will be interpreted as an entity of the universe of discourse
  - Predicates may be combined with terms into **formulas**
    - Formulas may be **quantified** or **concatenated** with connectives
Interpretation $I = (U, I_C, I_F, I_P)$

- $U$: universe of discourse
- $I_C$: constant symbol mapping
- $I_F$: functional symbol mapping
- $I_P$: predicate symbol mapping

Interpretations are needed to evaluate and interpret the individual components of a language.

Furthermore, we need variable assignment $\rho$

- Variable assignments may change very frequently within a single application session.
3.0 Summary of Last Lecture

• Again: What’s the trick with interpretations?
  – Consider $W \equiv \forall x \ (p(x, b, a) \rightarrow q(a, x))$
    • True or false? The interpretation determines!
  – Interpretation 1:
    • $I_C : \Gamma \rightarrow U, \ \{a \mapsto \text{Argo the Cat}, \ b \mapsto \text{Food}\}$
    • $I_P (p) := \{(m, n, o) \in U^3 \mid \text{“m gives n to o”} \} \subseteq U \times U \times U$
    • $I_P (q) := \{(m, n) \in U^2 \mid \text{“m loves n”} \} \subseteq U \times U$
    • “Argo the Cat loves everybody who gives him food” is true
  – Interpretation 2:
    • $I_C : \Gamma \rightarrow U, \ \{a \mapsto 10, \ b \mapsto 5\}$
    • $I_P (p) := \{(m, n, o) \in U^3 \mid m+n>o \} \subseteq U \times U \times U$
    • $I_P (q) := \{(m, n) \in U^2 \mid m<n \} \subseteq U \times U$
    • “$\forall x ((x+5> 10) \rightarrow (10< x))$” is obviously not true
Exercise 2.1

• Design a first order **language** for simple arithmetic’s on natural numbers. One should be able to **add** numbers, **subtract** numbers, **multiply** number, decide if a number is **equal** another number, and if a number is **greater** than another number.

  • $\Gamma := \{0, 1, 2, 3, \ldots\}$
  • $\Omega := \{+,-,\ast\}$
  • $\Pi := \{<,=\}$
  • $X := \{x,y,z\}$
Exercise 2.2

• Provide an interpretation

- \( \mathcal{L} = (\Gamma, \Omega, \Pi, X) \)
  - \( \Gamma := \{0, 1, 2, 3, \ldots\} \), \( \Omega := \{+, -, *\} \),
  - \( \Pi := \{<, =\} \), \( X := \{x, y, z\} \)

- \( I = (U, I_C, I_F, I_P) \)
  - \( U := \mathbb{N} \)
  - \( I_C : \Gamma \rightarrow U, \{0\mapsto0, 1\mapsto1, 2\mapsto2, 3\mapsto3, \ldots\} \)
  - \( I_F (+) : U \times U \rightarrow U, (n, m) \mapsto n + m \)
  - \( I_F (*) : U \times U \rightarrow U, (n, m) \mapsto n \times m \)
  - \( I_F (-) : U \times U \rightarrow U, (n, m) \mapsto n - m \)
  - \( I_P (<) := \{(n, m) \in U^2 \mid n < m\} \subseteq U \times U \)
  - \( I_P (=) := \{(n, m) \in U^2 \mid n = m\} \subseteq U \times U \)
Exercise 2.3

• We use infix notation in the following:
  – 5 is greater than 2: 5 > 2 (prefix: > (5, 2))
  – If x is greater than 0, then also x*y is greater than 0: x > 0 → x*y > 0
  – x is either greater than y, or x is equal to y, or x is smaller than y:
    x > y ∨ x = y ∨ y > x
  – The sum of any two numbers is always smaller than the product of the same two numbers
    ∀ x, y (x*y > x + y)
• Which statements are true? Provide an example substitution.

– \( 5 > 2 \): true
– \( x > 0 \rightarrow x \cdot y > 0 \): Possibly true; \( \rho(y) = 1 \ \rho(x) = 1 \)
– \( x > y \lor x = y \lor y > x \): true
– \( \forall x, y \ (x \cdot y > x + y) \): not true; \( \rho(y) = 1 \ \rho(x) = 1 \)
Which are formulas?

- \( \Gamma := \{a, b\} \), \( \Omega := \{f(x), g(x, y)\} \), \( \Pi := \{P, Q(x, y), R(x)\} \), \( X := \{x, y\} \)
- \( f(g(x, y)) \): no formula (it’s a term)
- \( P \): formula
- \( Q(x, y) \lor Q(a, b) \): formula
- \( Q(g(f(a), x), f(y)) \): formula
- \( \forall a (R(a)) \): no formula (\( a \) is constant)
- \( \exists x (f(x)) \): no formula (\( f(x) \) is no formula)
- \( R(x) \rightarrow \neg R(x) \): formula
- \( \neg R(\neg R(f(x))) \): no formula (predicate in predicate does not work)
3.1 Roadmap

• **Roadmap** for the immediate future…
  – Why do we need to bother with languages, interpretations, and formulae?

• Logic forms the **basic building blocks** of a knowledge base, because…
  – A knowledge base should be **storage efficient**
  – A knowledge base should be **easily extensible**

• **Deductive databases** implement these ideas
3.1 Example

• Consider an example: store a family tree
  – Important for finding genetic predispositions
  – E.g., Disease X is a risk, if two certain gene variants $Q_1$ and $Q_2$ are inherited from your parents
  – **Needed:** children names, all parent’s names, and the known possession of the specific gene variants
    • These are basic facts that cannot be derived from anything else
3.1 Example

– Store it in a relational database
  • Store the parents and their known genetic risk factors for all persons in a database
  • Is John at risk? Can we write some SQL query?

<table>
<thead>
<tr>
<th>Disease X</th>
<th>Name</th>
<th>Parent</th>
<th>Q₁</th>
<th>Q₂</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>John</td>
<td>Mary</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>John</td>
<td>Thomas</td>
<td>NULL</td>
<td>NULL</td>
</tr>
<tr>
<td></td>
<td>Mary</td>
<td>Peter</td>
<td>NULL</td>
<td>NULL</td>
</tr>
<tr>
<td></td>
<td>Mary</td>
<td>Karen</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>Thomas</td>
<td>George</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>Thomas</td>
<td>Sonja</td>
<td>NULL</td>
<td>NULL</td>
</tr>
</tbody>
</table>
Query for parents with predisposition

- (SELECT name FROM DiseaseX WHERE Q_1='Yes') INTERSECT (SELECT name FROM DiseaseX WHERE Q_2='Yes')

- But what if John could inherit from all ancestors?
Obviously this needs an extension of our model…

– Well, storing \((\text{Name}, \text{Ancestor}, Q_1, Q_2)\) would do the trick

• But this is not merely an extension, but would need a change of the database schema…

• And the actual extension needs to change the database content (who are ancestors?)

• And needs a lot more storage space…

• And opens the door for possible inconsistencies…
3.1 Relational vs. Deductive

• Relational databases may **not** be the prime choice for our problem set
  
  – Two kinds of knowledge
    • Static knowledge as given by tables
    • Derived knowledge as given by view mechanism
  
  – Queries in a **declarative** query language
  
  – Formal semantics is **relational algebra**
  
  – Class of completeness: **relational complete**
    • Especially: there is a problem with **recursive** views
We know rules to derive further knowledge from the basic knowledge about parentage

- **Deduction rules**
  - **Persons** have a name, a parent, and genetic predispositions
  - All parents of **Persons** are **Ancestors**.
  - All parents of **Ancestors** are **Ancestors**.
  - For all **Persons** there is a **Risk**, if some **Ancestor** has $Q_1$ and some **Ancestor** has $Q_2$

- These are **formulae** over the **predicates** Person, Ancestor and Risk
- Formulae represent **relationships** between real world objects
3.1 Relational vs. Deductive

• Predicates + formulae are the **database schema**

• Deductive databases consist of two major parts

  – **The extensional** database (EDB)
    • Fact collection as a (non-redundant) set of basic knowledge (facts, axioms)
    • The instance of data determines what further facts can be derived

  – **The intensional** database (IDB)
    • Rule collection as a (non-redundant) set of ways to derive new knowledge
    • The instance of rules determines how further facts can be derived
3.1 Models

• A valid question is which **interpretation** and **variable substitution** make a formula true?
  – Well, there are **unlimited** possible interpretations and variable substitutions
    • Should we try them all?
    • Does the computation ever end?
  – To make it easier: if the formula is **closed**, we can abstract from the specific variable substitution, only the interpretation matters
3.1 Models

• An interpretation \( I \) is called a model of a closed formula \( W \), if it evaluates to true with respect to \( I \)
  – Analogously, an interpretation \( I \) is called a model of a set of closed formulas \( \mathcal{W} \), if \( I \) is a model of all \( W \in \mathcal{W} \)

• Example
  – \( W \equiv \forall x \exists y \ (P(x, y)) \)
    • Let \( I \) be an interpretation which maps \( P \) to \(<\) on \( \mathbb{N} \)
      Then \( I \) is a model of \( W \): \( W \) is also called a fact with respect to \( I \)
  – \( W \equiv \exists x \forall y \ (P(x, y)) \)
    • Let \( I \) be then same interpretation mapping \( P \) to \(<\) on \( \mathbb{N} \)
      Then \( I \) is not a model of \( W \)
3.1 Models

• Now an interesting question arises for the evaluation of a set of closed formulas $\mathcal{W}$

  – Given a set of formulas, does it have a model?
    • $\mathcal{W}$ is called **satisfiable** (or consistent, contradiction-free), iff $\mathcal{W}$ has a model
    • $\mathcal{W}$ is called **unsatisfiable** (or inconsistent, contradictive), iff $\mathcal{W}$ does not have any model

  – We can immediately **stop the evaluation** of any unsatisfiable set
3.1 Models

• What is the connection between satisfiability of a set of formulae and inference?
  – Remember Aristotle’s principle of the indirect proof (reductio ad absurdum)
    • We want to prove (infer) a statement $W$ using a set of propositions $𝓦$
    • If we assume that $(¬W)$ holds and show a contradiction to some statement in $𝓦$, the proof is complete
    • That means $𝓦 \cup \{¬W\}$ is unsatisfiable
3.1 Semantic Equivalence

- Remember: we want to define concepts over basic fact data
- Natural question: do two concepts describe the same idea?
  - Two closed formulas $W_1$ and $W_2$ are semantically equivalent, iff $I(W_1) = I(W_2)$ for all $I$
  - It does not matter what interpretation we use, the evaluation of the two formulas is always the same
3.1 Semantic Conclusions

- Another natural question: can a certain fact be deduced from some given fact set?
  - A formula $W$ is a semantic conclusion of a set of formulas $\mathcal{W}$, iff every model of $\mathcal{W}$ is also a model of $W$
    - $\mathcal{W}$ may contain additional or broader concepts, but every interpretation that makes $\mathcal{W}$ true, also makes the ‘smaller’ concept of $W$ true
  - This is denoted by $\mathcal{W} \models W$ (W follows from $\mathcal{W}$)
Both questions are important for retrieval efficiency

- We aim at creating a deductive system which starts with a small set of facts to avoid inconsistencies
  - All derived knowledge will be generated at query time
- But we also want to describe all necessary concepts with a small set of rules to speed up response time
  - All rules need to be evaluated, redundant rules waste time
3.1 Test for Unsatisfiability

• Lemma:
  – If it can be deduced from $\mathcal{W}$ that the opposite of $W$ follows ($\mathcal{W} \models \neg W$), then $\mathcal{W} \cup \{W\}$ is unsatisfiable (and vice-versa)

• Thus, **unsatisfiability** of a set of closed formulas $\mathcal{W}$ can be proven by finding a single formula $W$ from the set such that it’s opposite follows from the remaining formulas
  – Test them all?! Seems a rather theoretical result…
3.1 Tautologies

• Finally there even are formulas for which every interpretation is a model
  – $\mathcal{W}$ is called universal, iff every interpretation is a model of $\mathcal{W}$ (denoted by $\models \mathcal{W}$)
  – $\mathcal{W}$ then is a referred to as tautology
Now, what are tautologies?

- Tautologies are **always true**, whatever interpretation is used
  - Thus, they are true independently of their actual **content**
- The **set of all tautologies** is thus very interesting, as it contains all universal statements
  - Those are also true for any specific, given interpretation and may thus form a great tool for reasoning

Example for tautologies

- \( W \lor \neg W \)
- \( W_1 \land W_2 \rightarrow W_1 \)
- \( (W_1 \rightarrow W_2) \land (W_2 \rightarrow W_3) \rightarrow (W_1 \rightarrow W_3) \)
- “To be or not to be”
3.1 Tautologies

- Tautologies can be used to derive semantic equivalences which can be used as transformation rules
  - Proof by truth diagram...
  - $A \equiv \neg \neg A$
  - $A \land B \equiv B \land A$
    - $A \lor B \equiv B \lor A$
  - $A \land (B \lor C) \equiv (A \land B) \lor (A \land C)$
    - $A \lor (B \land C) \equiv (A \lor B) \land (A \lor C)$
3.1 Tautologies

\(- \neg(A \land B) \equiv (\neg A) \lor (\neg B)\)

• \(-\neg(A \lor B) \equiv (\neg A) \land (\neg B)\)

\(- A \rightarrow B \equiv (\neg A) \lor B\)

• \(A \land B \equiv \neg (A \rightarrow (\neg B))\)

• \(A \lor B \equiv (\neg A) \rightarrow B\)

\(- A \leftrightarrow B \equiv (A \rightarrow B) \land (B \rightarrow A)\)

\(- \forall x (P(x)) \equiv \neg \exists x (\neg P(x))\)

• \(\exists x (P(x)) \equiv \neg \forall x (\neg P(x))\)
• Is there a way to find the set of all tautologies?
  – Thus, finding all universal truth?
  – Also, this can be used to prove if a statement is universally true.

• There are two (equivalent) approaches
  – Model-theoretical: Is a formula true in all possible worlds, i.e. is any interpretation a model?
    • We did that before and will continue after the detour
  – Proof-theoretical: Can the truthfulness of a formula be proven by some rules and axioms?
3.2 Deduction Systems

• In this detour, we will focus on the second approach in form of **proof systems** and **deductive systems**
  
  – Made popular by **David Hilbert** during his efforts to formalize all math
  
  – Is a “mechanical” system for proving and generating of universally true statements from axioms and rules
3.2 Deduction Systems

• Who is David Hilbert?
  – Probably one of the most influential mathematicians of the early 20th century
  – Significant pioneer work in proof theory, logics, meta-mathematics
    • Main interest: Stronger focus on formalization, understandability and provability
  – Born 1862 in Königsberg, in 1895 became chair of the Math Department in Göttingen
  – Around 1910, Hilbert moved to theoretical physics
    • ... and brought them the joy of logics and formalism
• **Göttingen** was the most renowned University for Mathematics at that time
  
  – Brought to fame by Carl Friedrich **Gauss** and Bernhard **Riemann**
  
  – Most fundamental work in modern math was performed there
  
  – Just some people around in Hilbert’s later years: Emmy Noether, Alonzo Church, John von Neumann, Wilhelm Ackermann, …
  
  – Unfortunately, in 1933 most of the department fell victim to a Nazi swipe
The Hilbert Program

- Started by Hilbert around 1920
- Idea:
  - Formalize all existing theories to finite, complete set of axioms
  - Proof that these axioms are consistent
- Goals
  - Preciseness: Use precisely defined formalisms and mechanisms
  - Completeness: Show that all math can be proved by the system
  - Consistency: No contradictions will show up in the system
  - Decidability: For every statement, an algorithm can decide if it is true or not
- But we remember: The Gödel incompleteness theorem made the Hilbert program impossible in this form in 1933
  - Slight changes to the mission statement lead it to success.
  - Tools still remain
3.2 Deduction Systems

- So, now we also want to create a **deductive system in Hilbert style**
- First, we need some theorems:
- **Deduction theorem**
  - $\mathcal{W} \cup \{W_1\} \vdash W_2$ holds if and only if $\mathcal{W} \vdash W_1 \rightarrow W_2$
  - $W_2$ follows from $W_1$ and $\mathcal{W}$ iff $W_1 \rightarrow W_2$ follows from $\mathcal{W}$
  - The deduction theorem is considered a "**fundamental**" meta-rule which is true in each deductive theorem, but is not a theorem within the system itself
3.2 Deduction Systems

• Modus Ponens
  – Already introduced by Aristotle
  – “mode that affirms by affirming”
  – \( \{W_1, W_1 \rightarrow W_2 \} \vdash W_2 \)
  – If \( W_2 \) follows from \( W_1 \) and \( W_1 \) is true, also \( W_2 \) is true
  – Example:
    • Rule: “If it is Tuesday, then there is a KBS lecture.”
    • Fact: “Today is Tuesday.”
    • Derived fact: “Thus, today is a KBS lecture.”
One can prove that Modus Ponens is universally sound – i.e. it never generates incorrect knowledge.

In contrast consider the popular abduction inference rule

\[ \{W_1 \rightarrow W_2, W_2\} \vdash W_1 \]

Abduction often useful, but not sound

Example

- Rule: “If it has rained, the street is wet.”
- Fact: “The street is wet.”
- Derived fact: “Thus, it has rained”

Example

- Fact: “A patient has red dots in the face and high fever…”
3.2 Deduction Systems

• An Hilbert-Style deductive system for a language $\mathcal{L}$ consist of
  
  – A set of formulas of $\mathcal{L}$ called **logical axioms**
    • All other statements can be followed from the axioms
    • It cannot be proved within the system if they are true or not, they are just “given”
    • If you want to prove or deduce only tautologies, also your axioms need to be tautologies
    • Hilbert system use extreme numbers of axioms, thus they are also called **axiomatic systems**
  
  – A set of **inference rules**
    • Rules transform one statement into a new one
• **Example: deductive system**
  – **Axioms:** axioms are all well-formed formulae of $\mathcal{L}$ which are instances of one of the following *schemas*
    • A1: $A \rightarrow (B \rightarrow A)$
    • A2: $(A \rightarrow (B \rightarrow C)) \rightarrow ((A \rightarrow B) \rightarrow (A \rightarrow C))$
    • A3: $(\neg A \rightarrow \neg B) \rightarrow (B \rightarrow A)$
    • Thus, all axioms are *tautologies*
  – This system can be extended with additional axioms types to also cater for predicates and quantifiers
  – Thus, there are an *unlimited* number of axioms
    • Frog(Hector)→(Lake(Hector)→Frog(Hector)) (Type A1)
    • $\neg A \rightarrow (\neg \neg A \rightarrow \neg A)$ (also Type A1)
3.2 Deduction Systems

– As the only rule, we use modus ponens
  • \(\{A, A \rightarrow B\} \models B\)

– Modus ponens is usually enough for all axiomatic deductive systems
  • It can be shown that additional rules do not provide additional expressiveness
  • …but may be used for convenience
3.2 Deduction Systems

- The axioms and rules contain only formulas using $\rightarrow$ and $\neg$
  - But by using the equivalence rules, all other formulas with $\land$, $\lor$, or $\leftrightarrow$ can be \textit{transformed} to only use $\rightarrow$
    - $A \land B \equiv \neg (A \rightarrow (\neg B))$
    - $A \lor B \equiv (\neg A) \rightarrow B$
    - $A \leftrightarrow B \equiv (A \rightarrow B) \land (B \rightarrow A)$
• Are those axioms really **tautologies**?

– **A1**: \( A \rightarrow (B \rightarrow A) \)

\[
\begin{array}{cccc}
A & B & B \rightarrow A & A \rightarrow (B \rightarrow A) \\
0 & 0 & 1 & 1 \\
0 & 1 & 0 & 1 \\
1 & 0 & 1 & 1 \\
1 & 1 & 1 & 1 \\
\end{array}
\]

– **A3**: \( (\neg A \rightarrow \neg B) \rightarrow (B \rightarrow A) \)

\[
\begin{array}{ccccc}
A & B & \neg A \rightarrow \neg B & B \rightarrow A & A3 \\
0 & 0 & 1 & 1 & 1 \\
0 & 1 & 0 & 0 & 1 \\
1 & 0 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 \\
\end{array}
\]
Deductive systems now generate **proofs**

- If you want to prove that a statement \( A \) is satisfiable or a tautology, you construct a prove which ends with statement \( A \)

A proof from a set \( \mathcal{W} \) to \( A \) in a deductive system is a finite sequence \( W_1, \ldots, W_n \) of formulas of \( \mathcal{L} \) such that \( W_i \) is either an **axiom**, is in \( \mathcal{W} \), or follows from one of the previous \( B_j \) by the **inference rules**

- \( \mathcal{W} \) is the set of **hypothesis** from which \( A \) follows
3.2 Deduction Systems

• Example Proof:

  – Is \( \neg B \rightarrow (B \rightarrow A) \) a tautology?
    • i.e. \( \vdash \neg B \rightarrow (B \rightarrow A) \) ?
  
  – By using the deduction theorem, we get
    • \( \neg B \vdash (B \rightarrow A) \)
  
  – \( W_1 \equiv \neg B \) (Hypothesis)
  – \( W_2 \equiv \neg B \rightarrow (\neg A \rightarrow \neg B) \) (Axiom 1)
  – \( W_3 \equiv \neg A \rightarrow \neg B \) (MP of \( W_1 \) and \( W_2 \))
  – \( W_4 \equiv (\neg A \rightarrow \neg B) \rightarrow (B \rightarrow A) \) (Axiom 3)
  – \( W_5 \equiv B \rightarrow A \) (MP of \( W_3 \) and \( W_4 \))
3.2 Deduction Systems

• Fun Proof:

- \(-\neg\neg A \models A\) ?
- \(W_1 \equiv \neg\neg A\) (Hypothesis)
- \(W_2 \equiv \neg\neg A \rightarrow (\neg\neg\neg A \rightarrow \neg\neg A)\) (Axiom I)
- \(W_3 \equiv \neg\neg A \rightarrow \neg\neg A\) (MP \(W_1 \& W_2\))
- \(W_4 \equiv (\neg\neg\neg A \rightarrow \neg\neg A) \rightarrow (\neg A \rightarrow \neg\neg\neg A)\) (Axiom 3)
- \(W_5 \equiv \neg A \rightarrow \neg\neg\neg A\) (MP \(W_3 \& W_4\))
- \(W_6 \equiv (\neg A \rightarrow \neg\neg\neg A) \rightarrow (\neg\neg A \rightarrow A)\) (Axiom 3)
- \(W_7 \equiv \neg\neg A \rightarrow A\) (MP \(W_5 \& W_6\))
- \(W_8 \equiv A\) (MP \(W_1 \& W_7\))
• Hilbert-style deduction has several drawbacks
  
  – Few rules, but many axioms
    • This is quite the opposite of what we want in a deductive database (e.g. the system of this detour has a unlimited, enumerable number of axioms…)
  
  – Finding a proof is very tricky
    • It’s hard to see when which axioms are needed to complete the proof
    • Thus, often we just end up doing trial & error
      – This is not what we want to a database
  
  – Feels unnatural
    • Many people felt that this kind of deduction is very unnatural and does not resemble the way how a mathematician would perform a proof
Better Idea: **Natural Deduction**

– **Use more rules, but a limited set of axioms**

– Most famous natural deduction calculus introduced by the Göttinger mathematician **Gerhard Gentzen**

  * **Gentzen Sequence Calculus**, developed in 1938
  * "Ich wollte zunächst einmal einen Formalismus aufstellen, der dem wirklichen Schließen möglichst nahe kommt. So ergab sich ein 'Kalkül des natürlichen Schließens'"

– These calculi have, in modified from, later been adapted by deductive databases
3.2 Deduction Systems

• Wonderful example for Gentzen Calculus goes here

• If you can see this slide, please re-download in a couple of days and hope that the content has been provided.
3.3 Test for Unsatisfiability

• Back to our topic…
  – A user starts extending concepts for his intensional database $\mathcal{W}$ formula by formula (i.e. closed formulae!)
  – For each new formula $W$ we need to test whether $(\mathcal{W} \cup \{W\})$ is unsatisfiable
    • Using our lemma, this can be done by showing that the formula $\neg W$ already follows from the set of formulas $\mathcal{W}$
    • Which means that every model of $\mathcal{W}$ is also a model of $\neg W$
    • Which means that all possible interpretations have to be tested..?!
    • We are back into the model-theoretical world
3.3 Test for Unsatisfiability

• Obviously there is an \textit{unlimited number} of possible interpretations…

• Idea: use interpretations that are \textit{representative} for the \textit{entire class} of all interpretations!
  – Are there such interpretations?
  – For what type of closed formulae?
  – For \textit{clauses} (certain type of closed formulae) the \textit{Herbrand interpretations} are representative
3.3 Clauses

• Basically **clauses** consist of **literals**
  
  – The set of **literals** $L_L$ consists of all atomic formulae $A \in A_L$ and the respective negated atomic formulae $\neg A$
    
    • The atomic formulae are called **positive literals**
    
    • The negated atomic formulae are called **negative literals**
    
    • If some atomic formula does not contain variables, it is called a **ground literal**
  
  – e.g.:
    
    • $A, \neg A, \text{Frog}(\text{Hector}), \neg \text{Frog}(\text{Hector}), \text{isGreen}(x), \neg \text{isGreen}(x), \ldots$
3.3 Clauses

• A **clause** is the universal closure of a **disjunction** of literals
  
  \[ \forall (L_1 \lor L_2 \lor \ldots \lor L_n), \quad L_i \in L_L \]

• A **Horn clause** is a clause that only contains at most a **single positive literal**
  
  \[ \text{e.g. } \forall (\neg A_1 \lor \neg A_2 \lor \ldots \lor \neg A_{n-1} \lor A_n), \quad A_i \in A_L \]
  
  – Horn clauses without a positive literal are called **goal clauses**
  
  – Horn clauses with exactly one positive literal are called **definite clauses**
  
  – Horn clauses with one positive but no negative literals are called **fact clauses**
• So, what is special about horn clauses?
  – Remember the transformation rule for semantic equivalence: $(\neg A) \lor B \equiv A \rightarrow B$
    • Thus, definite Horn clauses actually represent an implication
    • $\neg A_1 \lor \neg A_2 \lor \ldots \lor \neg A_{n-1} \lor A_n \equiv (A_1 \land A_2 \land \ldots \land A_{n-1}) \rightarrow A_n$
• Back to the topic: Define a **representative interpretation** which can replace any other
  – So called **Herbrand Interpretation**

• The **Herbrand interpretation** of an language $\mathcal{L}$ is based on
  – **Herbrand Universe** $\mathcal{U}_\mathcal{L}$, consisting of all ground terms
  – **Herbrand Base** $\mathcal{B}_\mathcal{L}$, consisting of all ground atoms
3.3 The Herbrand Universe

• How to construct the **Herbrand base** \( B_\mathcal{L} \)?
  – Take all the terms of the Herbrand universe and apply the predicates of the language \( \mathcal{L} \) to them
  – For each predicate symbol there is a (usually infinite) number of terms that can be used as argument
    • For every \( P \in \Pi \) as a \( n \)-ary predicate symbol all combinations of \( n \) terms \( t_i \) from the universe \( U_\mathcal{L} \) are used
    • \( P(t_1, ..., t_n) \subseteq B_\mathcal{L} \) with \( t_i \in U_\mathcal{L} \)
Example

- Given is the language $\mathcal{L}$
  - $\Gamma := \{a, b\}$, $\Omega := \{f, g\}$, $\Pi := \{P\}$, $\mathcal{X} := \{\}$
- The **Herbrand universe** thus is
  - $U_{\mathcal{L}} = \{a, f(a), g(a), f(f(a)), f(g(a)), \ldots\} \cup \{b, f(b), g(b), f(f(b)), f(g(b)), \ldots\}$
  - All **terms** which can be generated by using the function and constant symbols
- The **Herbrand base** is
  - $B_{\mathcal{L}} = \{P(a), P(f(a)), P(g(a)), P(f(f(a))), P(f(g(a))), \ldots\} \cup \{P(b), P(f(b)), P(g(b)), P(f(f(b))), P(f(g(b))), \ldots\}$
  - All **ground atoms** which can be generated using the universe
    - i.e. no variables allowed here
3.3 The Herbrand Universe

- Given a language $\mathcal{L}$ which allows following statements and $\Gamma := \{\text{Hector, green}\}$, $\Omega := \{\text{hasColor}\}$, $\Pi := \{\text{Frog, equals}\}$
  
  - $W \equiv \forall x \ (\text{Frog}(x) \rightarrow \text{equals(hasColor}(x), \text{green}))$
  
  - $U_\mathcal{L} = \{\text{Hector, hasColor(Hector)}, \text{hasColor(hasColor(Hector))}, \ldots\} \cup \{\text{green, hasColor(green), hasColor(hasColor(green))}, \ldots\}$
  
  - $B_\mathcal{L} = \{\text{Frog(Hector), Frog(hasColor(Hector))}, \ldots\} \cup \{\text{Frog(green), Frog(hasColor(green))}, \ldots\} \cup \{\text{equals(Hector, green), equals(hasColor(Hector), green)}, \ldots\}$
Finally, an **Herbrand interpretation** $I=(U, I_C, I_F, I_P)$ is given by

- $U = U_\mathcal{L}$
  - The Herbrand universe is used as universe
- $I_C(c) := c$
  - Thus, any constant symbol $c \in \Gamma$ is interpreted by itself
- $I_F(f): U_\mathcal{L} \times \ldots \times U_\mathcal{L} \to U_\mathcal{L}$, $f(t_1,\ldots,t_n) \mapsto f(t_1,\ldots,t_n)$
  - Any functional symbol $f \in \Omega$ is interpreted by itself
- Each language entity is mapped to an equivalent universe symbol
  - Thus, we create a completely **symbolic interpretation** without a specific real-world semantics
• Example:
  – An Herbrand Interpretation evaluates the term $f(g(a))$ to $f(g(a)) \in U_\mathcal{L}$
  – Given a substitution $\rho(x) = g(f(b))$, the term $f(x)$ evaluates to $f(g(f(b))) \in U_\mathcal{L}$

  – Keep in mind that the term $f(g(a))$ and the universe element $f(g(a))$ are not the same although they look the same!
    • One actually means something, the other is just a symbol
3.3 The Herbrand Universe

• $U$, $I_C$, and $I_F$ are the same for all Herbrand interpretations

• Herbrand interpretations only differ with respect to the predicate interpretation $I_P$
  
  – For two different Herbrand interpretations, $P(a)$ might be true in one and false in another
  
  – Thus, Herbrand interpretation can be defined by listing all atoms from the base which evaluate to true

• A Herbrand interpretation can identified with a subset of the Herbrand base and vice versa

• e.g. Herbrand Interpretation $I_1 = \{P(a), P(f(a))\}$, Herbrand Interpretation $I_2 = \{P(g(a)), P(g(b))\}$
3.3 The Herbrand Universe

• A Herbrand Model of a set of formulas $\mathcal{W}$ is a Herbrand interpretation, which is a model of $\mathcal{W}$

• Example: $W \equiv \forall x,y(\text{loves}(x, y) \rightarrow \text{loves}(y, x))$
  
  – ... language $\mathcal{L}$ is implicitly given
  
  – $I_1 := \{\text{loves}(\text{Tarzan}, \text{Jane}), \text{loves}(\text{Jane}, \text{Tarzan})\}$
    • $I_1$ is a Herbrand Model (remember, closed world!)
  
  – $I_2 := \{\text{loves}(\text{Tarzan}, \text{Jane}), \text{loves}(\text{Jane}, \text{Paul D’Arnot})\}$
    • $I_2$ is not a Herbrand Model
Lemma

- Given a set of clauses $\mathcal{W}$
  - $\mathcal{W}$ has a model, if and only if $\mathcal{W}$ has a Herbrand model
  - $\mathcal{W}$ is unsatisfiable, if and only if $\mathcal{W}$ has no Herbrand model

- That means that all symbols in a (set of) clause(s) can be interpreted in a purely syntactical way
  - If there is a syntactic possibility to satisfy the clause(s), there will also be some (more or less useful) semantic interpretation
• Using this lemma, we can finally test the unsatisfiability of $\mathcal{W} \cup \{W\}$
  – Remember: we have to show $\mathcal{W} \models \neg W$
  – But now, we just have to show the existence/nonexistence of a single Herbrand model instead testing all existing models

• But careful, this lemma only works for clauses, not for general closed formulas
3.3 The Herbrand Universe

• **So... How do Herbrand models help?**
  – They are just a *syntactical interpretation* without any relation to the real world…?
  – Can’t I always construct a Herbrand model for a satisfiable formula?

• **Consider this:**
  – We want to build a deductive *database*.
  – So, we need rules how to *use the data* within a database to *construct* Herbrand interpretations!
  – If a Herbrand interpretation constructed by the symbolic data of a DB is also a *model*, it can be used to for further evaluation and querying!
• **Herbrand Theory**
  – Why and how do Herbrand interpretations work?

• **Database Clauses**
  – How does data relate to models, interpretations, and rules?

• **Datalog**
  – How can we work with deduction in a database?

\[
\begin{align*}
\text{ancestor}(X,Y) & : = \text{parent}(X,Y) . \\
\text{ancestor}(X,Y) & : = \text{ancestor}(X,Z) , \text{ancestor}(Z,Y)
\end{align*}
\]