The 30 Years War

■ reduction in German population 15–30%
■ in some territories 3/4 of the population died
■ male population reduced by almost half
■ population of Czech lands reduced by 1/3
Leibniz's dream:

"a general method in which all truths of the reason would be reduced to a kind of calculation. At the same time this would be a sort of universal language or script, but infinitely different from all those projected hitherto; for the symbols and even the words in it would direct reason; and errors, except those of fact, would be mere mistakes in calculation."

If controversies were to arise, "there would be no more need of disputation between two philosophers than between two accountants. For it would suffice to take their pencils in their hands, and say to each other: Let us calculate."

*Dissertio de Arte Combinatoria*, 1666
<table>
<thead>
<tr>
<th>Leibniz</th>
<th>Logic in Practice</th>
<th>Satisfiability</th>
<th>ILP</th>
</tr>
</thead>
</table>

| 1 handout: slides |
| 730W journal entries were due |
Logic in Practice

- Horn Clauses
- Semantic Nets
- Description Logic
- Example DL
- Satisfiability
- ILP
\[ x \land y \rightarrow z \]
\[ x \land y \rightarrow z \equiv \neg x \lor \neg y \lor z \]

at most one positive literal (exactly one = ‘definite clause’)

\[
\text{Cat}(x) :- \text{Furry}(x), \text{Meows}(x).
\text{Cat}(y) :- \text{Feline}(y).
\text{Furry}(A).
\text{Meows}(A).
\text{? Cat}(z).
\]

Still undecidable in first-order case.

Propositional: Unit resolution (Modus Ponens) is sound and complete in linear time for Horn theories: ‘forward chaining’.
Each rule ‘fires’ at most once, each variable ‘processed’ at most once

‘expert systems’
Semantic Networks

Leibniz Logic in Practice
- Horn Clauses
- Semantic Nets
- Description Logic
- Example DL

Satisfiability

ILP

Mammals

Persons

Female Persons

Male Persons

Mary

John

SubsetOf

MemberOf

HasMother

SisterOf

Legs

SubsetOf

SubsetOf

SubsetOf

SubsetOf

1

2

Wheeler Ruml (UNH)
Multiple aspects:

- A visual notation
- A restricted logic
- A set of implementation tricks

Typically:

- Efficient indexing
- Precomputation
- Methods for defaults or typicality

Aka: frames, inheritance networks, semantic graphs, description logics, terminological logics, ontologies
computing categories and membership
including:
1. subsumption
2. classification
3. inheritance

missing:
1. negation
2. disjunction
3. nested functions
4. existentials
5. intractability
1. concepts (primitive and derived), instances
2. roles (definitional) and properties (assertional)
3. subsumption: \( \text{subsumes} \ (x, y) \) iff
   
   (a) \( x \) is a concept, and
   
   (b) same primitive concept ancestor, and
   
   (c) for each role of \( x \) with restriction \( r_x \)
       
       i. \( y \) has same role with restriction \( r_y \), and
       
       ii. \( r_x \) subsumes \( r_y \)
Model of $P$: an interpretation in which $P$ is true
Satisfiable: $\exists$ a model
Entailment: if $Q$ is true in every model of $P$, then $P \models Q$
Valid: true in any interpretation
Given a formula of boolean logic, is there any assignment of T/F to its variables that makes the entire formula true?

\[(a \lor b \lor c) \land (\neg a \lor b \lor \neg c) \land (\neg a \lor \neg b \lor c) \land (\neg a \lor \neg b \lor \neg c)\]
<table>
<thead>
<tr>
<th>Leibniz</th>
<th>Logic in Practice</th>
</tr>
</thead>
<tbody>
<tr>
<td>Satisfiability</td>
<td></td>
</tr>
<tr>
<td>■ Terminology</td>
<td>■ SAT</td>
</tr>
<tr>
<td>■ DPLL</td>
<td>■ Break</td>
</tr>
<tr>
<td>■ GSAT</td>
<td>■ ILP</td>
</tr>
</tbody>
</table>

Wheeler Ruml (UNH)
boolean logic is a CSP: clause $\rightarrow$ nogood
boolean logic is a CSP: clause $\rightarrow$ nogood

**DPLL($\alpha$):**
- if $\alpha$ has no clauses, return true
- if $\alpha$ has an empty clause, return false
- if $\alpha$ contains a unit clause, return DPLL(Simplify($\alpha$, literal))
  - $v \leftarrow$ choose a variable in $\alpha$
  - if DPLL(Simplify($\alpha$, $v$)) is true, return true
  - else, return DPLL(Simplify($\alpha$, $\neg v$))

**Simplify($\alpha$, literal):**
- remove clauses in $\alpha$ where literal is positive
- remove $\neg$literal from clauses where it appears
- return new alpha

‘unit propagation’, model-finding
Leibniz
Logic in Practice
Satisfiability
- Terminology
- SAT
- DPLL
- Break
- GSAT
ILP

Break

- asst 2
- exam 1
- office hours
- final projects
1. Start with a random solution
2. Repeatedly flip variable to satisfy the most clauses
   (a) If same as previous, try second-best.
3. When tired, restart

*(Selman and Kautz (GSAT, WalkSAT), …)*
1. Start with a random solution
2. Repeatedly flip variable to satisfy the most clauses
   (a) If same as previous, try second-best.
3. When tired, restart

(Selman and Kautz (GSAT, WalkSAT), ...)

DPLL: 50 vars = 1.4 secs, 100 vars = 2.8 min, 140 vars = 4.7 hrs
1. Start with a random solution
2. Repeatedly flip variable to satisfy the most clauses
   (a) If same as previous, try second-best.
3. When tired, restart

(Selman and Kautz (GSAT, WalkSAT), …)

DPLL: 50 vars = 1.4 secs, 100 vars = 2.8 min, 140 vars = 4.7 hrs

GSAT: 100 vars = 6 secs, 140 vars = 14 secs, 500 vars = 1.6 hrs
Inductive Logic Programming
Three types:

**Supervised:** classification (= prediction of class)

**Unsupervised:** compression (= prediction of actual value)

**Reinforcement:** sequence of decisions with occasional reward

Each can be on-line (incremental) or off-line (batch).

**Terminology:**

1. Hypothesis space
2. Training data (vs test data, for off-line case)
3. Performance metric (often on validation data)
Given: ground facts and background definitions
Find: short (almost Horn) clauses that cover positive examples and not negative ones

Background \land Hypothesis \land Descriptions \models Classifications
Descriptions:
- Father(Philip, Charles)
- Mother(Mum, Margaret)
- Married(Diana, Charles)
- Male(Philip)
- Female(Beatrice)
- Father(Philip, Anne)
- Mother(Mum, Elizabeth)
- Married(Elizabeth, Philip)
- Male(Charles)
- Female(Margaret)

Classifications:
- Grandparent(Mum, Charles)
- ¬Grandparent(Mum, Harry)
- Grandparent(Elizabeth, Beatrice)
- ¬Grandparent(Spencer, Peter)

Background: \( Parent(x,y) \leftrightarrow Mother(x,y) \lor Father(x,y) \)

Target: \( Grandparent(x,y) \leftrightarrow \exists z \ Parent(x,z) \land Parent(z,y) \)
Given: ground facts and background definitions
Find: short (almost Horn) clauses that cover positive examples and not negative ones

**Sequential covering** (‘FOIL’)

\[ \text{rules } \leftarrow \{ \} \]

Until no remaining positives (or good enough):

\[ \text{new } \leftarrow \text{empty rule} \]

While false positives (e.g., covers any negatives):

Add best single literal precondition

Add \textit{new} to \textit{rules}

Remove positive examples covered by \textit{new}
Example

→ Grandfather(x, y)
Example

\[ \rightarrow \text{Grandfather}(x,y) \]

\[ \text{Father}(x,y) \rightarrow \text{Grandfather}(x,y) \quad \text{(always wrong)} \]

\[ \text{Parent}(x,y) \rightarrow \text{Grandfather}(x,y) \quad \text{(many false +)} \]

\[ \text{Father}(x,z) \rightarrow \text{Grandfather}(x,y) \quad \text{(selected)} \]
→ \textit{Grandfather}(x,y)

\textit{Father}(x,y) \rightarrow \textit{Grandfather}(x,y) \quad \text{ (always wrong)}

\textit{Parent}(x,y) \rightarrow \textit{Grandfather}(x,y) \quad \text{ (many false +)}

\textit{Father}(x,z) \rightarrow \textit{Grandfather}(x,y) \quad \text{ (selected)}

\textit{Father}(x,z) \land \textit{Parent}(z,y) \rightarrow \textit{Grandfather}(x,y) \quad \text{ (target)}
New literals:

1. Any predicate over any variables, where at least one of the variables is in previous literal or head
2. Equal(x, y), where x and y are already in rule
3. Negation of any of the above

Best: maximizes ‘information gain’

Clause must be shorter than positives it explains (cf Ockham’s razor).
ILP Applications

1. Mutagenesis
2. Toxicity
3. Rules of chess
4. Protein structure
5. Parsers
What question didn’t you get to ask today?
What’s still confusing?
What would you like to hear more about?

Please write down your most pressing question about AI and put it in the box on your way out.

Thanks!