The 30 Years War (1618–1648)

- reduction in German population 15–30%
- in some terrirories 3/4 of the population died
- male population reduced by almost half
- population of Czech lands reduced by 1/3

Gottfried Wilhelm Leibniz (1646-1716)

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

Logic in Practice

1. given $\exists$, can introduce new constant
2. given sentence with ground expresion, can introduce $\exists$
3. given $\forall$, can introduce new constant
4. given sentence, can introduce $\forall$ over new free variable

$\wedge$ **elimination/introduction:**

$\vee$ **introduction:**

$\neg\neg$ **elimination:**

Inference Rules

Modus Ponens:

Resolution:

Abduction:

Induction:

mathematical induction $\neq$ inductive reasoning

Alfred Horn (1951)

$$x \wedge y \rightarrow z$$

Alfred Horn (1951)

$x \wedge y \rightarrow z \equiv \neg x \vee \neg y \vee z$

at most one positive literal (eaxctly one = 'definite clause')

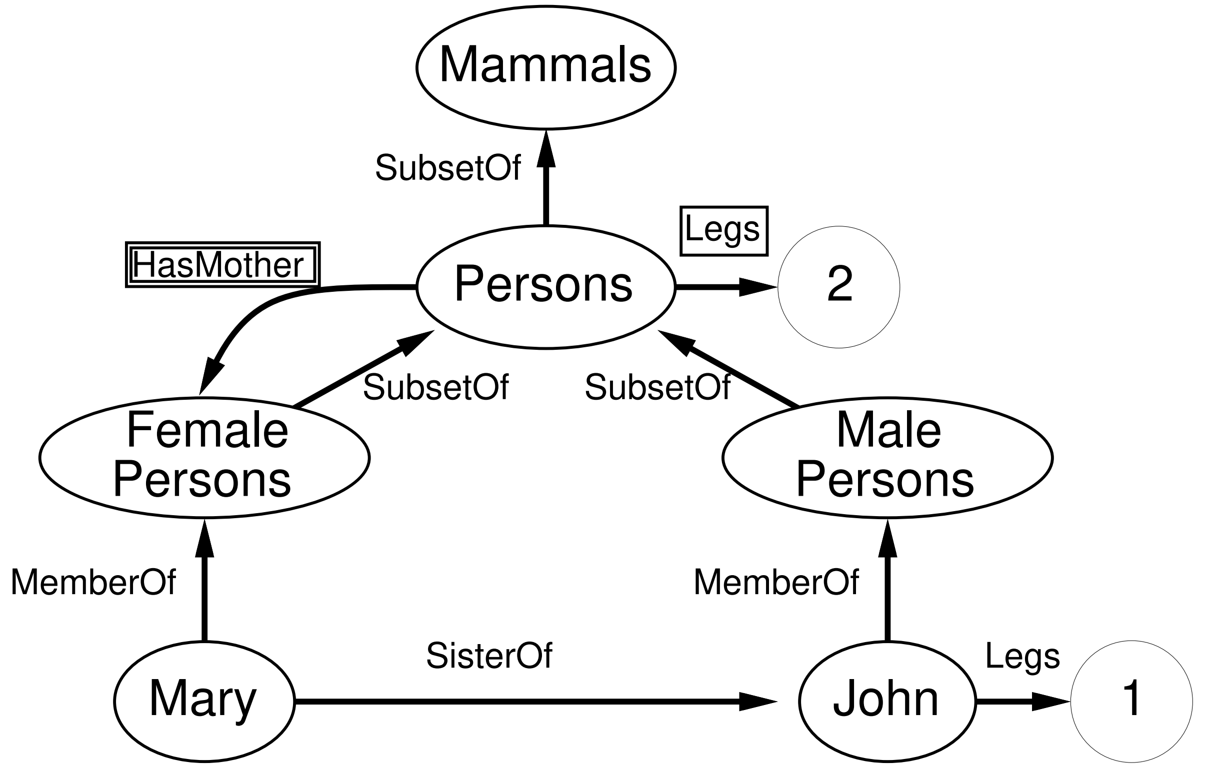
```
Cat(x) :- Furry(x), Meows(x).
Cat(y) :- Feline(y).
Furry(A).
Meows(A).
?  Cat(z).
```

Still semi-decidable 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



Semantic Networks

Multiple aspects:

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

Typically:

- Efficient indexing
- Precomputation
- Methods for defaults or typicality

Aka: frames, inheritence 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

Example DL

1. concepts (primitive and derived), instances
2. roles (definitional) and properties (assertional)
3. subsumption: *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

Break

- assts 6, 7
- preliminary proposals due next class
- share project ideas!

Logics of Action

Event Calculus

Events and fluents are reified:

$$Member(E23, Flyings) \wedge Agent(E23, John) \wedge Happens(E23, I7) \ldots$$

$$T(At(John, KN113), t_1) \wedge Terminates(E23, At(John, KN113), t_2) \ldots$$

World state (= situation) is reified:

$$Result(GoForward, s_0) = s_1$$

$$Result(Turn(right), s_1) = s_2$$

$$\forall s, a, b\; Clear(a, s) \wedge Clear(b, s) \rightarrow On(a, b, Result(PutOn(a, b), s))$$

Problems with Logic

Defaults: hard to have coherent semantics and efficient inference (default logics, answer set programming, probabilistic logic)

Ramification problem: choosing what to infer (specialized systems)

Retraction: when previous truth becomes false (truth maintenance systems)

Qualification problem: making rules correct (probabilistic logic)

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)

Inductive Logic Programming

Given: ground facts and background definitions
Find: short (almost Horn) clauses that cover positive examples and not negative ones

$$Background \land Hypothesis \land Descriptions \models Classifications$$

Input

Descriptions:

Father(Philip, Charles)	*Father(Philip, Anne)*
Mother(Mum, Margaret)	*Mother(Mum, Elizabeth)*
Married(Diana, Charles)	*Married(Elizabeth, Philip)*
Male(Philip)	*Male(Charles)*
Female(Beatrice)	*Female(Margaret)*

Classifications:

Grandparent(Mum, Charles)	*Grandparent(Elizabeth, Beatrice)*
$\neg$*Grandparent(Mum,Harry)*	$\neg$*Grandparent(Spencer,Peter)*

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

Target: $Grandparent(x,y) \leftrightarrow \exists\, z\; Parent(x,z) \wedge Parent(z,y)$

Given: ground facts and background definitions
Find: short (almost Horn) clauses that cover positive examples and not negative ones

Sequential covering ('FOIL')
rules ← { }
Until no remaining positives (or good enough):
    new ← empty rule
    While false positives (eg, covers any negatives):
        Add best single literal precondition
    Add *new* to *rules*
    Remove positive examples covered by *new*

Example

$\rightarrow$ *Grandfather(x,y)*

Example

$\rightarrow$ *Grandfather(x,y)*

Father(x,y) $\rightarrow$ *Grandfather(x,y)* (always wrong)
Parent(x,y) $\rightarrow$ *Grandfather(x,y)* (many false $+$)
Father(x,z) $\rightarrow$ *Grandfather(x,y)* (selected)

Example

$\rightarrow$ *Grandfather(x,y)*

Father(x,y) $\rightarrow$ *Grandfather(x,y)*	(always wrong)
Parent(x,y) $\rightarrow$ *Grandfather(x,y)*	(many false $+$)
Father(x,z) $\rightarrow$ *Grandfather(x,y)*	(selected)

Father(x,z) $\wedge$ *Parent(z,y)* $\rightarrow$ *Grandfather(x,y)* (target)

Specializing

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!