1

Knowledge Based Systems

(CM0377)

Lecture 12

(Last modified 2nd May 2002)

2

Reasoning with incomplete information

- Consider:
  - 'If I turn the key, the engine will start' *(default reasoning - assumes no fault)*
  - 'The engine doesn't start, so the starter motor must be faulty' *(abduction - finding explanations for observed facts. Note this isn't the* only *explanation)*
  - 'The sky is full of dark clouds: it's going to precipitate' *(induction from past experience)*
- In each case, the information is *incomplete*

3

The Closed World Assumption

- 'Everything that is not known to be true, must be false'
- Recall that the Herbrand Universe *U* of a program *P* is the set of terms that can be generated from the constants and functors of *P;* Herbrand Base *B* of program *P* is set of ground atoms that can be generated from predicates in *P* and ground terms in *U*.
- Then for every ground atom A belonging to *B* that is not a logical consequence of *P,* we assume that A is false.

4

CWA closure

- If A is false then :-A is true.
- The CWA-closure of a program *P* is the program with :-A added to it for all A in the Herbrand base that are not logical consequences of *P:*

  $CWA(P) = P \cup \{\text{:-A} \mid \text{A} \in B \text{ and } P \nvDash \text{A}\}$

5

Example

- Suppose *P* is:

```
likes(S, andrew) :- student_of(S, andrew).
student_of(john, andrew).
```

- Then ground atoms which are logical consequences of *P* are:

```
likes(john, andrew).
student_of(john, andrew).
```

- Add *negation* of remaining atoms in Herbrand base to obtain *CWA(P)* ...

6

Example (ctd.)

```
likes(S, andrew) :- student_of(S, andrew).
student_of(john, andrew).
:-likes(andrew, andrew).
:-likes(andrew, john).
:-likes(john, john).
:-student_of(john, john).
:-student_of(andrew, andrew).
:-student_of(andrew, john).
```

- Note there is only one model, namely:

```
{likes(john, andrew), student_of(john, andrew)}
```

- Whereas there are lots of models of *P.* This model is the *intended model* of *P.*

7

Non-monotonicity

- Now suppose we add the clause *C*=`likes(X, Y)` to *P*. Then *CWA*(*P*∪{C}) is:

```
likes(S, andrew) :- student_of(S, andrew).
likes(X, Y).
student_of(john, andrew).
:-student_of(john, john).
:-student_of(andrew, andrew).
:-student_of(andrew, john).
```

- and it is no longer true, for example, that `:-likes(andrew, john)` can be derived. Thus new information can invalidate previous conclusions.

8

Monotonicity and non-monotonicity

- Deductive reasoning is *monotonic* in the sense that:

$Theory \vdash Conclusion \Rightarrow$

$Theory \cup \{AnyFormula\} \vdash Conclusion$

- Our closed world assumption makes reasoning non-monotonic in that by adding new clauses we can make *negative* conclusions no longer derivable.

9

Another way of looking at monotonicity

- If we define the *deductive closure* of a theory as set of conclusions that can be derived from it:

$Closure(Theory) = \{Conclusion \mid Theory \vdash Conclusion\}$

- Then montonicity can be expressed thus:

$Theory1 \subseteq Theory2 \Rightarrow$

$Closure(Theory1) \subseteq Closure(Theory2)$

- (Monotonic in the sense that as the theory gets bigger, so does the closure)

10

Notes

- CWA is limited to definite clauses. For a program *P* comprising only definite clauses then there is a unique model of *CWA*(*P*).
- See Flach pp. 156-159, if interested, for information about *Predicate Completion* which applies to a program containing general clauses that has a particular property, namely that it's *'stratified'*.

11

Next time

- The last lecture!!
- Default reasoning:
  - using *not* and negation as failure
  - defining special default rules
- Basic elements of abduction and induction