Rule-based expert systems

overview and concepts

Henning Christiansen

Expert systems

- One of most applied and successful "intelligent" sort of technology
- Basic idea:
  - take one or more experts in a specific field
  - drain their brains for knowledge
  - formalize it into rules that are put into system that can reason using them
    - "expert" themselves may use system — or fire the experts and hire lower educated people...
- 15-20 y's ago "knowledge engineering" was thought to become a new major trend
  - books-books-books
  - courses-courses-courses
  - proposals for educational programs in K.E.

History and applications (sketch)

- MYCIN, Mid 1970'ies (Stanford Univ. USA)
- successful rule-based system for medical diagnosis
- EMYCIN, around 1980, "shell" for developing expert systems
- Quite many commercial systems and developer shells
- Recent overview paper:
  - Expert System in Real World Applications
    K.S. Wai, A. Latif, B.A. Rahman, H.F. Zaiyadi, A.A. Aziz (2005)
    http://www.generation5.org/content/2005/Expert_System.asp
  - Agriculture, education, environmental management, medicine
- Also: Repairing complicated machinery (cars, trains, planes ...)

Basic principles

- If-then rules
  - forward and backward chaining
  - often with probabilities, fuzzy, other (ad-hoc) uncertainty measures
  - often with dialogue with the user

Example:

- Prolog seen as rule-based expert system shell with backward chaining and no uncertainty and very little dialogue

A rule, forward and backward chaining

IF rains AND go-out AND NOT umbrella THEN get-wet

Forward:

Bottom-up, data-driven

From known facts **rains, go-out, NOT umbrella**
   conclude **get-wet**

Backward:

Bottom-up, goal-driven

For testing hypothesis **get-wet**, you need to check
   whether **rains, go-out** and **NOT umbrella** holds

Typically sequences of steps, maybe mixing both
Dialogue with user
   System:Did you bring umbrella?
   User: What if I Brought umbrella?

5

Demonstrations

- Prolog as backward chaining
- Version that requires user to edit the program
   ```
   ?- ['my_kb0Prolog.txt'].
   ```
- Prolog extended with ask-user facilities
   ```
   ?- ['expert0.txt'].
   ?- ['my_kb0.txt'].
   ```
   (order is important)

6