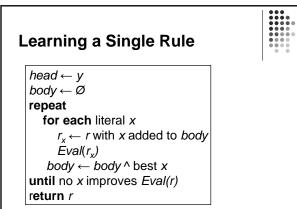
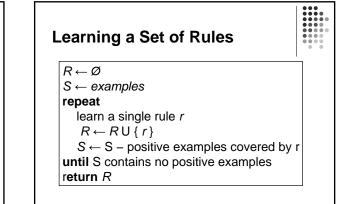


• Eval(r): Accuracy, info gain, coverage, support, etc.





First-Order Rule Induction (a.k.a. Inductive Logic Programming)



- *y* and *x_i* are now predicates with arguments E.g.: *y* is Ancestor(x,y), *x_i* is Parent(x,y)
- Literals to add are predicates or their negations
- Literal to add must include at least one variable already appearing in rule
- Adding a literal changes # groundings of rule
 E.g.: Ancestor(x,z) ^ Parent(z,y) ⇒ Ancestor(x,y)
- *Eval(r)* must take this into account E.g.: Multiply by # positive groundings of rule still covered after adding literal

MLN Structure Learning

- Generalizes feature induction in Markov nets
- Any inductive logic programming approach can be used, but . . .
- Goal is to induce any clauses, not just Horn
- Evaluation function should be likelihood
- Requires learning weights for each candidate
- Turns out not to be bottleneck
- Bottleneck is counting clause groundings
- Solution: Subsampling

MLN Structure Learning

- Initial state: Unit clauses or hand-coded KB
- Operators: Add/remove literal, flip sign
- Evaluation function: Pseudo-likelihood + Structure prior
- Search: Beam search, shortest-first search