Outline

- Two formulations for learning: Inductive and Analytical
- Perfect domain theories and Prolog-EBG
A Positive Example
The Inductive Generalization Problem

Given:
- Instances
- Hypotheses
- Target Concept
- Training examples of target concept

Determine:
- Hypotheses consistent with the training examples
The Analytical Generalization Problem

Given:
- Instances
- Hypotheses
- Target Concept
- Training examples of target concept
- *Domain theory for explaining examples*

Determine:
- Hypotheses consistent with the training examples *and the domain theory*
An Analytical Generalization Problem

Given:

• Instances: pairs of objects
• Hypotheses: sets of horn clause rules
• Target Concept: Safe-to-stack(x,y)
• Training Example: Safe-to-stack(OBJ1,OBJ2)

  On(OBJ1,OBJ2)
  Isa(OBJ1,BOX)
  Isa(OBJ2,ENDTABLE)
  Color(OBJ1,RED)
  Color(OBJ2,BLUE)
  Volume(OBJ1,.1)
  Density(OBJ1,.1)
...

• Domain Theory:

  Safe-To-Stack(x,y) :- Not(Fragile(y))
  Safe-To-Stack(x,y) :- Lighter(x,y)
  Lighter(x,y) :- Weight(x,wx), Weight(y,wy),
                  Less(wx,wy)
  Weight(x,w) :- Volume(x,v), Density(x,d),
                Equal(w, v*d)
  Weight(x,5) :- Isa(x, ENDTABLE)
...

Determine:

• Hypotheses consistent with training examples and domain theory
Assumes domain theory is *correct* (error-free)

- Prolog-EBG is algorithm that works under this assumption
- This assumption holds in chess and other search problems
- Allows us to assume explanation = proof
- Later we’ll discuss methods that assume *approximate* domain theories
Initialize hypothesis = {}

For each positive training example not covered by hypothesis:

1. **Explain** how training example satisfies target concept, in terms of domain theory

2. **Analyze** the explanation to determine the most general conditions under which this explanation (proof) holds

3. **Refine** the hypothesis by adding a new rule, whose preconditions are the above conditions, and whose consequent asserts the target concept
Explanation of a Training Example

Explanation:

Safe-to-Stack(OBJ1,OBJ2)

Lighter(OBJ1,OBJ2)

Weight(OBJ1, 0.6)

Weight(OBJ2,5)

Volume(OBJ1,2) Density(OBJ1,0.3) Equal(0.6, 2*0.3) Less-Than(0.6, 5) Type(OBJ2,ENDTABLE)

Training Example:
Computing the Weakest Preimage of Explanation

Safe-to-Stack(OBJ1,OBJ2)

Lighter(OBJ1,OBJ2)

Weight(OBJ1, 0.6)

Less-Than(0.6,5)

Weight(OBJ2,5)

Volume(OBJ1,2)

Density(OBJ1,0.3)

Equal(0.6,2*0.3)

Volume(OBJ1, 0.6)

Weight(x,wx)

Less-Than(wx,wy)

Weight(y,wy)

Density(x,dx)

Equal(wx,vx*dx)

Less-Than(wx,wy)

Type(OBJ2,ENDTABLE)

Weight(OBJ1, 0.6)

Volume(x,vx)

Density(x,dx)

Equal(wx,vx*dx)

Less-Than(wx,5)

Type(y,ENDTABLE)
Regression Algorithm

Regress(\text{Frontier}, \text{Rule}, \text{Expression}, U_{I,R})

\text{Frontier}: the set of expressions to be regressed through \text{Rule}
\text{Rule}: a horn clause.
\text{Expression}: the member of \text{Frontier} that is inferred by \text{Rule} in the explanation.
\text{U}_{I,R}: the substitution that unifies \text{Rule} to the training example in the explanation

Returns the list of expressions forming the weakest preimage of \text{Frontier} with respect to \text{Rule}

\begin{align*}
\text{let } & \text{Consequent} \leftarrow \text{Rule consequent} \\
\text{let } & \text{Antecedents} \leftarrow \text{Rule antecedents} \\
1. & \text{U}_{E,R} \leftarrow \text{most general unifier of } \text{Expression} \text{ with Consequent} \\
& \quad \text{such that there exists a substitution } S \text{ for which} \\
& \quad S(\text{U}_{E,R}(\text{Consequent})) = U_{I,R}(\text{Consequent}) \\
2. & \text{Return } \text{U}_{E,R}((\text{Frontier} \cdot \text{Consequent} + \text{Antecedent}))
\end{align*}

\textbf{Example:}

\begin{align*}
\text{Regress}\{&\text{Volume}(x,vs), \text{Density}(x,dx), \text{Equal}(wx,vx*dx), \\
&\text{Less-Than}(wx,wy), \text{Weight}(y,wy)\}, \\
&\text{Weight}(z,5) :- \text{Type}(z,\text{ENDTABLE}), \\
&\text{Weight}(y,wy), \\
&\text{\{OBJ2/z\}}
\end{align*}

\begin{align*}
\text{Consequent} & \leftarrow \text{Weight}(z,5) \\
\text{Antecedents} & \leftarrow \text{Type}(z,\text{ENDTABLE}) \\
\text{U}_{E,R} & \leftarrow \{y/z, 5/wy\}, \ (S = \{\text{OBJ2}/y\})
\end{align*}

\begin{align*}
\text{Result} & \leftarrow \{\text{Volume}(x,vs), \text{Density}(x,dx), \text{Equal}(wx,vx*dx), \\
&\text{Less-Than}(wx,5), \text{Type}(y,\text{ENDTABLE})\}
\end{align*}
Lessons from Safe-to-Stack Example

• Justified generalization from single example
• Explanation determines feature relevance
• Regression determines needed feature constraints
• Generality of result depends on domain theory
• Still require multiple examples
Perspectives on Prolog-EBG

- Theory-guided generalization from examples
- Example-guided operationalization of theories
- "Just" restating what learner already "knows"

Is it learning?
- Are you learning when you get better over time at chess?
  - Even though you already know everything in principle, once you know rules of the game...
- Are you learning when you sit in a mathematics class?
  - Even though those theorems follow deductively from the axioms you've already learned...