#6: EUSolver Discussion

Sankha Narayan Guria

EECS 700: Introduction to Program Synthesis

EUSolver: strengths

- Divide-and-conquer (aka condition abduction)
    - scales better on conditional expressions
    - but: they didn’t invent it
- Neat application of decision tree learning
    - leverages the structure of Boolean expressions
- Empirically does well, especially on PBE
    - why specifically on PBE?

EUSolver: weaknesses

- Only applies to conditional expressions
- Does not always generate the smallest expression
  - in the limit, can find the smallest solution
  - but unclear when to stop
- Only works for pointwise specifications
  - but so do ALL CEGIS-based approaches

EUSolver

- **Q3:** What does EUSolver use as behavioral constraints? Structural constraint? Search strategy?
  - First-order formula
  - Conditional expression grammar
  - Bottom-up enumerative with OE + pruning
- Why do they need the specification to be pointwise?
  - Example of a non-pointwise spec?
  - How would it break the enumerative solver?

EUSolver

- **Q4:** What are pruning/decomposition techniques EUSolver uses to speed up the search?
  - Condition abduction + special form of equivalence reduction
- Why does EUSolver keep generating additional terms when all inputs are covered?
- How is the EUSolver equivalence reduction differ from observational equivalence we saw in class?
  - How do they overcome the problem that it's not robust to adding new points?
- Can we discard a term that covers a subset of the points covered by another term?

EUSolver

- **Q5:** What would be a naive alternative to decision tree learning for synthesizing branch conditions?
  - Learn atomic predicates that precisely classify points
    - why is this worse?
    - is it as bad as ESolver?
  - Next best thing is decision tree learning w/o heuristics
    - why is this worse?
  - random forest, DNN, linear regression
  - Random sampling of conditions?
  - sequence of if-else based on examples?

Wrong!

Project Discussion