CS1571 HW 3 Report

Tianjian Meng

11-13-2017

1. Implementation decisions

    1. I used a lot of sub-functions to implement the 'firing a rule' to keep the overall code clean, make the code reusable and avoid a lot of loops.
    2. To implement the assigning of constant to variable, I made the 'assign' function recursive to make sure that only one constant could be assigned to one variable at the same time and the search could be sound and complete.
    3. To keep track of the inference process, I used a global variable to log them when new facts are added into the knowledge base.

2. Incremental forward-chaining

    1. I used two global variables, new-inferred and last-inferred, to keep track of the inference process.
    2. In naive forward-chaining, I check every rules and fire them until no new fact could be inferred, which is not efficient.
    3. In Incremental forward-chaining, I could use last-inferred to maintain all the facts that were inferred in last round, so that I just need to check that if there are any atom in each rule that matches the facts I just inferred. Only if a rule matches at least one atom with the facts inferred in last round could have the chance to be fired in this round.
    4. By doing this, I don't need to fire all rules in each round and I can avoid a lot of redundant computation. So IFC is much more efficient than FC.