A Query Language in Scheme

Mark Boady

August 1, 2013

Overview

- Lambda Calculus - Review of Question 3.39 Posted to website.
- Query Languages - Review of Query Language from SICP
- Metacircular Evaluator - Lab Introduction

Query Languages

- Query Languages Present a different method of writing programs.
- For most languages
  - We give a set of instructions and ask they be executed
- For a query language
  - We give a set of information and ask a question
- Terminology
  - Data base: Our set of known information
  - Query: A question we want answered
- Note: Similar but not exactly like Database Programming (SQL)

Example

- A data base in scheme (From SICP 4.4.1)
  - (job (Bitdiddle Ben) (computer wizard))
  - (job (Hacker Alyssa P) (computer programmer))
  - (job (Fect Cy D) (computer programmer))
  - (job (Tweakit Lem E) (computer technician))
  - (can-do-job (computer wizard) (computer programmer))
- A simple query
  - (job ?x (computer programmer))
- Find all the people who have the job computer programmer.
  - Fill in the blank ?x

```
;;; Query results:
(job (Hacker Alyssa P) (computer programmer))
(job (Fect Cy D) (computer programmer))
```

Queries

- To start the SICP Query Interpreter

```
(load "ch4-query.scm")
;This is with the Lab 3 files
(initialize-data-base microshaft-data-base)
(query-driver-loop)
```

- We can use mutiple variables

```
(job ?x ?y)
```

  - This would just list everyone's job

- We can also repeat variables

```
(supervisor ?x ?x)
```

  - This would give us all the people who are their own supervisor.

Partial Matches

- Partial Matches
  - This will find any person whose job starts with the word computer

    ```
    (job ?x (computer . ?type))
    ```

  - This will not find someone with the job (computer programmer trainee)

    ```
    (job ?x (computer ?type))
    ```

Compound Queries

- We can use “and” and “or” to make more complex queries
- I want to know the address of all the computer programmers

```
(and (job ?person (computer programmer))
     (address ?person ?where))
```

- I want to know all the people who have either Ben or Alyssa as their supervisor

```
(or (supervisor ?x (Bitdiddle Ben))
    (supervisor ?x (Hacker Alyssa P)))
```

Compound Queries

- The not command can be used to negate a statement
- I want to know all the people supervised by Ben who are not computer programmers

```
(and (supervisor ?x (Bitdiddle Ben))
     (not (job ?x (computer programmer))))
```

- lisp-value can be used if we need to compare values
- I want to know everyone who makes more then 30,000

```
(and (salary ?person ?amount)
     (lisp-value > ?amount 30000))
```

Adding New Rules

- We can add new rules to the query data base
- Two people are the same

```
(rule (same ?x ?x))
```

- Two people live near each other if they are from the same town but not the same person

```
(rule (lives-near ?person-1 ?person-2)

      (and (address ?person-1 (?town . ?rest-1))

           (address ?person-2 (?town . ?rest-2))
           (not (same ?person-1 ?person-2))))
```

- Now we can ask who lives near Ben

```
(lives-near ?x (Bitdiddle Ben))
```

Programming with Logic

- Note: The single quote after mydb will be broken in the pdf (smart-quotes)

```
(load "ch4-query.scm")
(define mydb '(
(rule (append-to-form () ?y ?y))


    (rule (append-to-form (?u . ?v) ?y (?u . ?z))
    (append-to-form ?v ?y ?z))

))
(initialize-data-base mydb)
(query-driver-loop)
(append-to-form (a b) (c d) ?z)
```

- For this version of append, the return value is placed in ?z
- In most query languages, we need to give a variable to place the solution

Programming with Logic

- What is the list generated by appending these two lists

  `(append-to-form (a b c) (d e f) ?z)`

- What is the list that append to make a specific final list

  `(append-to-form (a b) ?y (a b c d))`

- What are all the possible lists that merge to a target list

  `(append-to-form ?x ?y (a b c d))`

MetaCircular Evaluator

- A Scheme Interpreter written in Scheme
- Why?
  - Teaching!
  - Extending the Language or Adding Syntax
  - Debuggers
- Jikes RVM
  - A Java Research Virtual Machine Written in Java
  - The RVM runs on top of the JVM
  - Used For Research in:
    - Garbage Collection
    - Dynamic Parallelization
    - Machine Learning for Dynamic Compilation
    - Dynamic Typing

Running the mcEval

- Here is an example of how to load and run code

```
(load "ch4-mceval.scm")
(define the-global-environment (setup-environment))
(driver-loop)
(+ 4 5)
```

- Primatives
  - We want to reuse base functionality
    - car,cdr,cons, null?, +, -, =, *, etc
- We want to reimplement higher order functions
  - Anything that can be built from the primary functions
  - We would need to rewrite reverse