LAAI - M2 - Homework

Michele Milesi

March 2022

Abstract

Exercise 1

Exercise 1.4

Observe that our model of evaluation allows for combinations whose operators are compound expressions. Use this observation to describe the behavior of the following procedure:

```
(define (a-plus-abs-b a b)
  ((if (> b 0) + -) a b))
```

Solution

Exercise 1.5

Ben Bitdiddle has invented a test to determine whether the interpreter he is faced with is using applicative-order evaluation or normal-order evaluation. He defines the following two procedures:

```
(define (p) (p))

(define (test x y)
  (if (= x 0)
      0
      y))
```

Then he evaluates the expression

```
(test 0 (p))
```

What behavior will Ben observe with an interpreter that uses applicative-order evaluation? What behavior will he observe with an interpreter that uses normal-order evaluation? Explain your answer. (Assume that the evaluation rule for the special form `if` is the same whether the interpreter is using normal or applicative order: the predicate expression is evaluated first, and the result determines whether to evaluate the consequent or the alternative expression).

Solution

Exercise 2

Exercise 1.35

Show that the golden ratio φ is a fixed point of the transformation $x \mapsto 1 + \frac{1}{x}$, and use this fact to compute φ by means of the `fixed-point` procedure.

Solution

Exercise 1.36

Modify `fixed-point` so that it prints the sequence of approximations it generates, using the `newline` and `display` primitives shown in Exercise 1.22. Then find a solution to $x^x = 1000$ by finding a fixed point of $x \mapsto \frac{\log(1000)}{\log(x)}$. (Use Scheme's primitive `log` procedure, which computes natural logarithms). Compare the number of steps this takes with and without average damping. (Note that you cannot start `fixed-point` with a guess of 1, as this would cause division by $\log(1) = 0$).

Solution

Exercise 1.37

a. An infinite *continued fraction* is an expression of the form

$$f = \cfrac{N_1}{D_1 + \cfrac{N_2}{D_2 + \cfrac{N_3}{D_3 + \dots}}}$$

As an example, one can show that the infinite continued fraction expansion with the N_i and the D_i all equal to 1 produces $\dfrac{1}{\varphi}$, where φ is the golden ratio. One way to approximate an infinite continued fraction is to truncate the expansion after a given number of terms. Such a truncation – a so-called *k-term finite continued fraction* – has the form

$$\cfrac{N_1}{D_1 + \cfrac{N_2}{\ddots + \cfrac{N_k}{D_k}}}$$

Suppose that `n` and `d` are procedures of one argument (the term index i) that return the N_i and D_i of the terms of the continued fraction. Define a procedure `cont-frac` such that evaluating `(cont-frac n d k)` computes the value of the k-term finite continued fraction. Check your procedure by approximating $\frac{1}{\varphi}$ using

```
(cont-frac (lambda (i) 1.0)
           (lambda (i) 1.0)
           k)
```

for succesive values ok `k`. How large must you make `k` in order to get an approximation that is accurate to 4 decimals places?

b. If your `cont-frac` procedure generates a recursive process, write one that generates an iterative process. If it generates an iterative process, write one that generates a recursive process.

Solution

Exercise 1.38

In 1737, the Swiss mathematician Leonhard Euler published a memoir *De Fractionibus Continuis*, which included a continued fraction expansion for $e-2$, where e is the base of the natural logarithms. In this fraction, the N_i are all 1, and the D_i are successively $1, 2, 1, 1, 4, 1, 1, 6, 1, 1, 8, \dots$. Write a program that uses your `cont-frac` procedure from Exercise 1.37 to approximate e, based on Euler's expansion.

Solution

Exercise 3

Exercise 3.2

Explain why (in terms of the evaluation process) these two programs give different answers (i.e. have different distributions on return values):

```
(define foo (flip))
(list foo foo foo)

(define (foo) (flip))
(list (foo) (foo) (foo))
```

Solution

Exercise 3.5

Here is a modified version of the tug of war game. Instead of drawing strength from the continuous Gaussian distribution, strength is either 5 or 10 with equal probability. Also the probability of laziness is changed from 1/4 to 1/3. Here are four expressions you could evaluate using this modified model:

```
(define strength (mem (lambda (person) (if (flip) 5 10))))
(define lazy (lambda (person) (flip (/ 1 3))))

(define (total-pulling team)
  (sum
   (map (lambda (person)
          (if (lazy person)
              (/ (strength person) 2)
              (strength person)))
        team)))

(define (winner team1 team2)
  (if (< (total-pulling team1) (total-pulling team2))
      team2
      team1))

(winner '(alice) '(bob))                        ;; expression 1

(equal? '(alice) (winner '(alice) '(bob)))      ;; expression 2

(and (equal? '(alice) (winner '(alice) '(bob))) ;; expression 3
     (equal? '(alice) (winner '(alice) '(fred))))

(and (equal? '(alice) (winner '(alice) '(bob))) ;; expression 4
     (equal? '(jane) (winner '(jane) '(fred))))
```

a. Write down the sequence of expression evaluations and random choices that will be made in evaluating each expression.

b. Directly compute the probability for each possible return value from each expression.

c. Why are the probabilities different for the last two? Explain both in terms of the probability calculations you did and in terms of the "causal" process of evaluating and making random choices.

Solution

Exercise 3.6

Use the rules of probability, described above, to compute the probability that the geometric distribution defined by the following stochastic recursion returns the number 5.

```
(define (geometric p)
  (if (flip p)
      0
      (+ 1 (geometric p))))
```

Solution

Exercise 3.7

Convert the following probability table to a compact Church program:

A	B	P(A, B)
F	F	0.14
F	T	0.06
T	F	0.4
T	T	0.4

Hint: fix the probability of A and then define the probability of B to depend on whether A is true or not. Run your Church program and build a histogram to check that you get the correct distribution.

```
(define a ...)
(define b ...)
(list a b)
```

Solution

Exercise 4

Exercise 4.1

What are `(bernoulli-dist p)`, `(normal-dist` μ σ`)` exactly? Are they real numbers (produced in a random way)? We have seen that flip is a procudere with a probabilistic behaviour. Is, e.g., `(normal-dist` μ σ`)` something similar? Try to evaluate `(normal-dist 0 1)`

Solution

Exercise 4.2

Evaluate

```
(dist? (normal-dist 0 1))

(dist? (bernoulli-dist 0.5))

(dist? flip)
```

What is the difference between flip and (bernoulli-dist 0.5)?

Solution

Exercise 5

Exercise 5.4

Probabilistic Models of Cognition - Exercise 4

Solution

Exercise 6

Exercise 6.1

To see the problems of rejection sampling, consider the following variation of the previous example:

```
(define baserate 0.1)
(define (take-sample)
  (rejection-sampler
   (define A (if (flip baserate) 1 0))
   (define B (if (flip baserate) 1 0))
   (define C (if (flip baserate) 1 0))
   (define D (+ A B C))
   (observe/fail (>= D 2))
   A))
```

Try to see what happens when you lower the basesate. What happens if we set it to 0.01? And to 0.001?

Solution

Exercise 7

Exercise 7.1

Proposition. *The functional and matrix-based definitions of a MC are equivalent.*

Proof sketch. Given $c : X \to D(X)$, with $X = \{x_1, ..., x_n\}$, we construct the matrix P_c as $P_c(i,j) = c(x_i)(x_j)$. Vice versa, given P, we define $c_P(x_i)(x_j) = P(i,j)$. □

Instructions *Complete the above proof. Prove, in particular that for any $x \in X$, $c_P(x)$ is indeed a distribution; that c_P is a stochastic matrix; and that $P_{c_P} = P$ and $c_{P_c} = c$.*

Solution

Exercise 7.2

Prove that $c(x) = c^*(\delta_x)$.

Solution

Exercise 7.3

Prove that $c^*(\psi) = \psi(P_c)$.

Solution

Exercise 7.4

Prove that f ψ satisfied DBC, then ψ is stationary for P.

Solution