A MODULAR PROOF METHODOLOGY FOR ACL2

A Case Study: The Fundamental Theorem of Calculus Using Non-standard Analysis

Matt Kaufmann
EDS CIO Services

98 San Jacinto Blvd. Suite 500
Austin, TX 78701 USA
kaufmann@cio2000.eds.com

March, 1999

OUTLINE

- A MODULAR PROOF METHODOLOGY
  - Proof development
    - *Facilitates organization*
    - *Can eliminate proof replay problems*
  - Proof presentation
- CASE STUDY: THE FUNDAMENTAL THEOREM OF CALCULUS
  Proof uses Ruben Gamboa's ACL2(r) (and its associated books), which supports reasoning using the method of *non-standard analysis*.

A TYPICAL HIGH-LEVEL PROOF PROCESS

1. GOAL:
   To prove theorem `THM`.
2. MAIN REDUCTION:
   Should suffice to prove lemmas `L1` and `L2`.
3. SUPPORT:
   Need to include a library `"lib.lisp"`.
4. PROOF HACKING:
   Need a few additional lemmas.

PROOF SKELETON

```
(include-book "lib")    ; 3. SUPPORT

(defaxiom L1 ....)      ; 2. MAIN REDUCTION
(defaxiom L2 ....)

<Miscellaneous lemmas>  ; 4. PROOF HACKING

(defthm THM ....)       ; 1. GOAL
```

CORRESPONDING BOOK

```
(include-book "lib")

(encapsulate ()
 (local (include-book "L1"))
 (defthm L1 ....))

(encapsulate ()
 (local (include-book "L2"))
 (defthm L2 ....))

<Miscellaneous lemmas>

(defthm THM ....)
```


PROPOSED METHODOLOGY: TWO TYPES OF BOOKS

- *LEMMA* BOOK:
  Book name is the same as name of final theorem proved in the book.
- *LIBRARY* BOOK:
  Other than a lemma book; typically contains definitions and useful lemmas (especially, rewrite rules).

PROPOSED METHODOLOGY: PROOF DEVELOPMENT

MAIN FOCUS:
Develop lemma books using the process just shown.

- Main lemmas are postponed to be proved in subsidiary lemma books, replaced by `defaxiom` in the present book.
- Useful lemmas may be moved into lemma books from time to time.
- A makefile based on Bishop Brock's IHS library makefile automates proof replay and also outline creation (explained below).

PROPOSED METHODOLOGY: PROOF PRESENTATION

- Preliminary tool assists in proof presentation.
- It creates top-down outlines of the proof to each depth of at least 2.

The next slides show part of the depth-3 outline that was generated mechanically from the books for the Fundamental Theorem of Calculus.

TOOL OUTPUT EXAMPLE (p. 1)

```
Main.  fundamental-theorem-of-calculus.
 (IMPLIES (AND (REALP A) (REALP X))
          (EQUAL (INTEGRAL-RCFN-PRIME A X) (RCFN X)))
using lemmas:
  ("split-integral-by-subintervals" "ftoc-lemma")
and library books:
  ("integral-rcfn" "nsa-lemmas" "integral-rcfn-lemmas" "riemann-lemmas"
   "make-partition" "riemann-defuns")
==============================
Main.1.  split-integral-by-subintervals.
 (IMPLIES (AND (REALP A) (REALP B) (REALP C))
          (EQUAL (INTEGRAL-RCFN A C)
                 (+ (INTEGRAL-RCFN A B) (INTEGRAL-RCFN B C))))
using lemmas:
  ("riemann-sum-approximates-integral"
   "integral-rcfn-equal-if-i-close")
and library books:
  ("integral-rcfn-lemmas" "nsa-lemmas" "integral-rcfn" "riemann-lemmas"
   "riemann-defuns")
==============================
```


TOOL OUTPUT EXAMPLE (p. 2)

```
Main.1.1.  riemann-sum-approximates-integral.
 (IMPLIES (AND (PARTITIONP P) (EQUAL A (CAR P))
               (EQUAL B (CAR (LAST P))) (< A B) (STANDARD-NUMBERP A)
               (STANDARD-NUMBERP B) (I-SMALL (MESH P)))
          (I-CLOSE (RIEMANN-RCFN P) (INTEGRAL-RCFN A B)))
using lemmas (NOT shown just below:  depth limit reached):
  ("riemann-sum-approximates-integral-1"
   "riemann-sum-approximates-integral-2")
and library books:
  ("integral-rcfn" "riemann-defuns")
==============================
Main.1.2.  integral-rcfn-equal-if-i-close.
 (IMPLIES (AND (REALP A) (STANDARD-NUMBERP A) (REALP B)
               (STANDARD-NUMBERP B) (< A B) (REALP Y) (REALP Z))
          (EQUAL (EQUAL (INTEGRAL-RCFN A B)
                        (+ (STANDARD-PART Y) (STANDARD-PART Z)))
                 (I-CLOSE (INTEGRAL-RCFN A B) (+ Y Z))))
using lemmas (NOT shown just below:  depth limit reached):
  ("standard-part-equal-if-i-close")
and library books:
  ("integral-rcfn-lemmas" "riemann-lemmas" "integral-rcfn"
   "riemann-defuns")
```


TOOL OUTPUT EXAMPLE (p. 3)

```
==============================
Main.2.  ftoc-lemma.
 (IMPLIES (AND (REALP EPS) (NOT (EQUAL EPS 0)) (I-SMALL EPS) (REALP X)
               (STANDARD-NUMBERP X))
          (EQUAL (STANDARD-PART (/ (INTEGRAL-RCFN X (+ X EPS)) EPS))
                 (RCFN X)))
using lemmas:
  ("integral-rcfn-quotient-between-non-classical"
   "between-i-close-implies-i-close")
and library books:
  ("min-x-and-max-x-lemmas" "../nsa/realp" "defaxioms" "integral-rcfn"
   "max-and-min-attained" "nsa-lemmas" "integral-rcfn-lemmas"
   "riemann-lemmas" "make-partition" "riemann-defuns")
==============================
Main.2.1.  integral-rcfn-quotient-between-non-classical.
```

.... and so on.

TOOL OUTPUT EXAMPLE: Avoiding restatement

```
Main.1.1.  riemann-sum-approximates-integral.
 (IMPLIES .....) .....
==============================
Main.1.1.1.  riemann-sum-approximates-integral-1.
 (IMPLIES .....) .....
==============================
Main.1.1.1.1.  partitions-give-i-close-riemann-sum.
 (IMPLIES .....) .....
==============================
.....
==============================
Main.1.1.2.  riemann-sum-approximates-integral-2.
 (IMPLIES .....) .....
==============================
Main.1.1.2.1.  riemann-sum-approximates-integral-1.
 <See Main.1.1.1 for details.>
==============================
Main.1.2.  integral-rcfn-equal-if-i-close.
 (IMPLIES .....) .....
```


BRIEF INTRODUCTION TO NON-STANDARD ANALYSIS

See also Keisler's Calculus book and Nelson's paper (cf. Ruben's dissertation)

- `(STANDARD-NUMBERP` x`)`: "Ordinary" real number.
- `(I-SMALL` x`)`: $|x| < r$ for all standard $r > 0$.
- `(I-CLOSE` x y`)`: $|y - x| < r$ for all standard $r > 0$.
- `(I-LIMITED` x`)`: $|x| < r$ for some standard r.
- `(I-LARGE` x`)`: Not `(I-LIMITED` x`)`.
- `(STANDARD-PART` x`)`: For `(I-LIMITED` x`)`, that unique standard r such that `(I-CLOSE` x r`)`.
- `(I-LARGE-INTEGER)`: A particular `I-LARGE` integer.

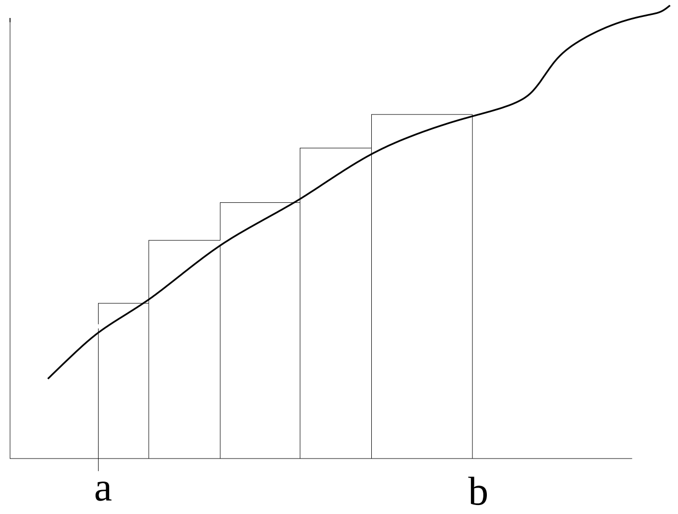
CONTINUITY FORMALIZED

```
(ENCAPSULATE ; from Ruben Gamboa

 ((RCFN (X) T)) ; real continuous function

 (LOCAL (DEFUN RCFN (X) X)) ; witness

 (DEFTHM RCFN-STANDARD
   (IMPLIES (STANDARD-NUMBERP X)
            (STANDARD-NUMBERP (RCFN X)))
   :RULE-CLASSES
   (:REWRITE :TYPE-PRESCRIPTION))
```


CONTINUITY FORMALIZED (continued)

```
(DEFTHM RCFN-REAL
  (IMPLIES (REALP X)
           (REALP (RCFN X)))
  :RULE-CLASSES
  (:REWRITE :TYPE-PRESCRIPTION))

(DEFTHM RCFN-CONTINUOUS
  (IMPLIES (AND (STANDARD-NUMBERP X)
                (REALP X)
                (I-CLOSE X Y)
                (REALP Y))
           (I-CLOSE (RCFN X) (RCFN Y)))))
```


FORMALIZING THE FTOC (1)



FORMALIZING THE FTOC (2)

```
(DEFUN DELTAS (P) ; successive differences
  (IF (AND (CONSP P) (CONSP (CDR P)))
      (CONS (- (CADR P) (CAR P))
            (DELTAS (CDR P)))
    NIL))

(DEFUN MAP-RCFN (P) ; map rcfn over the list p
  (IF (CONSP P)
      (CONS (RCFN (CAR P)) (MAP-RCFN (CDR P)))
    NIL))

(DEFUN RIEMANN-RCFN (P)
  (DOTPROD (DELTAS P) (MAP-RCFN (CDR P))))
```


FORMALIZING THE FTOC (3)

```
(DEFUN-STD INTEGRAL-RCFN (A B)
  (COND
   ((OR (NOT (REALP A))
        (NOT (REALP B)))
    0)
   ((< A B)
    (STANDARD-PART
     (RIEMANN-RCFN
      (MAKE-PARTITION A B
        (I-LARGE-INTEGER)))))
   ((< B A)
    (- (.... B A .....)))
   (T 0)))
```


FORMALIZING THE FTOC (4)

```
(DEFUN-STD INTEGRAL-RCFN-PRIME (A X)
   (IF (AND (REALP A) (REALP X))
       (LET ((EPS (/ (I-LARGE-INTEGER))))
         (STANDARD-PART
          (/ (- (INTEGRAL-RCFN A (+ X EPS))
                (INTEGRAL-RCFN A X))
             EPS)))
     0)) ; hints omitted
```


FORMALIZING THE FTOC (5)

```
(DEFTHM INTEGRAL-RCFN-PRIME-EXISTS
  (IMPLIES
   (AND (REALP EPS)
        (NOT (EQUAL EPS 0))
        (I-SMALL EPS)
        (REALP A) (STANDARD-NUMBERP A)
        (REALP X) (STANDARD-NUMBERP X))
   (EQUAL (STANDARD-PART
           (/ (- (INTEGRAL-RCFN A (+ X EPS))
                 (INTEGRAL-RCFN A X))
              EPS))
          (INTEGRAL-RCFN-PRIME A X))))
   ; hints omitted
```

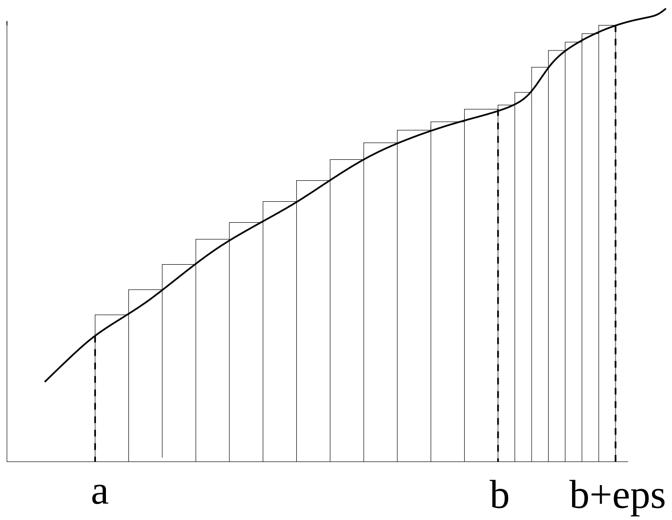
FORMALIZING THE FTOC (6)

```
(DEFTHM-STD FUNDAMENTAL-THEOREM-OF-CALCULUS
  (IMPLIES (AND (REALP A)
                (REALP X))
           (EQUAL (INTEGRAL-RCFN-PRIME A X)
                  (RCFN X)))) ; hints omitted
```

The theorem above was "MAIN" on our outline. Let us use the proof outline to explore the proof.


```
Main.1.  split-integral-by-subintervals.
 (IMPLIES (AND (REALP A) (REALP B) (REALP C))
  (EQUAL (INTEGRAL-RCFN A C)
         (+ (INTEGRAL-RCFN A B)
            (INTEGRAL-RCFN B C))))

Main.2.  ftoc-lemma.
 (IMPLIES
  (AND (REALP EPS) (NOT (EQUAL EPS 0))
       (I-SMALL EPS)
       (REALP X) (STANDARD-NUMBERP X))
  (EQUAL (STANDARD-PART
          (/ (INTEGRAL-RCFN X (+ X EPS)) EPS))
         (RCFN X)))
```


PROOF OF Main.1 (1)




```
Main.1.1.  riemann-sum-approximates-integral.
 (IMPLIES
  (AND (PARTITIONP P) (EQUAL A (CAR P))
       (EQUAL B (CAR (LAST P))) (< A B)
       (STANDARD-NUMBERP A)
       (STANDARD-NUMBERP B)
       (I-SMALL (MESH P)))
  (I-CLOSE (RIEMANN-RCFN P)
           (INTEGRAL-RCFN A B)))
```

Our "P" will be the result of gluing together the partitions for the two sub-integrals: A to C, C to B.

Below, think of `Y` and `Z` as being the Riemann sums formed from those two partitions.

```
Main.1.2.  integral-rcfn-equal-if-i-close.
 (IMPLIES
  (AND (REALP A) (STANDARD-NUMBERP A)
       (REALP B) (STANDARD-NUMBERP B)
       (< A B) (REALP Y) (REALP Z))
  (EQUAL (EQUAL (INTEGRAL-RCFN A B)
                (+ (STANDARD-PART Y)
                   (STANDARD-PART Z)))
         (I-CLOSE (INTEGRAL-RCFN A B)
                  (+ Y Z))))
```

But to apply the first lemma, this "`(+ Y Z)`" must be the Riemann sum for a single partition.


```
Main.1.  split-integral-by-subintervals.
Main.1.1.  riemann-sum-approximates-integral.
Main.1.2.  integral-rcfn-equal-if-i-close.
```

We could have put this in its own lemma book.

```
(DEFTHM SPLIT-RIEMANN-RCFN-BY-SUBINTERVALS
  (IMPLIES
   (AND (PARTITIONP P1)
        (PARTITIONP P2)
        (EQUAL (CAR (LAST P1)) (CAR P2)))
   (EQUAL (+ (RIEMANN-RCFN P1)
             (RIEMANN-RCFN P2))
          (RIEMANN-RCFN
           (APPEND P1 (CDR P2))))))
```


CONCLUSION

A proof methodology based on

- LEMMA BOOKS and
- LIBRARY BOOKS

has helped me in the FTOC effort in these ways:

- STAYING ORGANIZED DURING THE PROOF
- AVOIDING NEEDLESS PROOF RE-WORK
- EXPLAINING THE PROOF

I would be interested in discussing extensions and improvements to this methodology and its proof outlining tool.