

Contents

1	Introduction and Motivation	1
2	Riemann Integral: Definition of Partition	1
3	Definition of Riemann Sum	2
4	Definition of Riemann Integral	3
5	Linearity Properties	5
6	Step Functions	6
7	Fundamental Theorem of Calculus	10
8	Conclusion	11
A	Why Are PVS Proofs So Much Larger?	12
A.1	Formalizing Partitions	12
A.2	Step Functions	13
A.3	Complications Due To Working In Type Theory Rather Than Set Theory . .	14
B	Illustration	14
B.1	Continuous Function Is Integrable	15
B.2	Formal Proof Sketch of This Theorem	15
C	Example of Proof Deficiency in Rosenlicht	20
D	User Guide	20

1 Introduction and Motivation

There are several motivations for the development of the integral calculus library for the PVS theorem prover: (1) Increasingly our formal methods team is being called upon to develop analysis techniques that can demonstrate the safety of algorithms and systems used for air traffic management both on the ground and in the air [6, 4, 3]. This problem domain inevitably requires reasoning about the effect of the software on aircraft trajectories. As aircraft approach the airport and subsequently reach the final approach fix, they follow non-constant speed profiles that can require the use of calculus [4], (2) The power of the PVS theorem has been growing over the last 10 years. A particularly challenging problem is the formalization and mechanical verification of the classic proofs of integral calculus including the Fundamental Theorem of Calculus. The author has often wondered if theorem proving technology might ever reach the place where it can be a pedagogical aid to the mathematics student and eventually a tool of practical use to the mathematician. The formalization of the integral calculus in PVS should provide a basis for judging how close we are to this goal. (3) The goal of reducing all of mathematics to primitive logic has a long heritage. Russell and Whitehead sought to place all of the mathematics upon the foundation of set theory and classical logic. When the paradoxes were discovered at the beginning of the 20th century, they abandoned their effort. Although mathematicians have not demonstrated much interest in continuing the Russell and Whitehead program, computer scientists have [1]. The Lebesgue integral has been formalized in the Isabelle/Isar theorem prover [11] and the Gauge Integral in Isabelle/HOL as well [8]. Harrison describes the development of the theory of integration in the HOL theorem prover in [9]. L. Cruz-Filipe developed a constructive theory of analysis in the Coq theorem prover in his Ph.D dissertation [5]. Here, we are providing the same mathematical foundations for the PVS theorem prover. This work uses and extends the work done by Bruno Dutertre [7]. He developed the first version of the PVS analysis library which provided definitions and properties of limits, derivatives, and continuity. This work develops the theory of integration through the Fundamental Theorem of Calculus.

In this paper I will provide a summary of the formalization and a few illustrations of the mechanical proofs to emphasize the difference between the rigorous proofs provided by Rosenlicht [12] in his classic text and a mechanically checked proof in PVS [10, 13]. The PVS theories and proofs are available at NASA Langley's formal methods web site [2].

2 Riemann Integral: Definition of Partition

We begin our formalization of the integral with the definition of a partition. Rosenlicht defines a partition as follows:

Definition. Let $a, b \in \mathfrak{R}, a < b$. By a partition of the closed interval $[a, b]$ is meant a finite sequence of numbers x_0, x_1, \dots, x_N such that $a = x_0 < x_1 < x_2 \dots < x_N = b$.

In PVS a “finite sequence” is record with two fields:

```
finite_sequence: TYPE = [# length: nat, seq: [below[length] -> T] #]
```

To define a partition we create a predicate subtype of finite sequences with the appropriate properties:

```

integral_def[T: TYPE FROM real]: THEORY
BEGIN
  a,b,x: VAR T
  closed_interval(a:T, b:{x:T|a<x}): TYPE = { x | a <= x AND x <= b}

  partition(a:T,b:{x:T|a<x}): TYPE =
    {fs: finite_sequence[closed_interval(a,b)] |
      Let N = length(fs), xx = seq(fs) IN
      N > 1 AND xx(0) = a AND xx(N-1) = b AND
      (FORALL (ii: below(N-1)): xx(ii) < xx(ii+1))}

```

The width of this partition is defined by Rosenlicht as follows:

$$\max\{x_i - x_{i-1} : i = 1, 2, \dots, N\}$$

In PVS we have:

```

width(a:T, b:{x:T|a<x}, P: partition(a,b)): posreal =
  max({ l: real | EXISTS (ii: below(length(P)-1)):
    l = seq(P)(ii+1) - seq(P)(ii)})

```

In PVS it is necessary to include the endpoints of the interval a, b as the first arguments of `width`. Note the convenience of $\max\{x_i - x_{i-1} : i = 1, 2, \dots, N\}$ compared to the PVS formalization. In the PVS definition, there is an existential quantifier which is hidden by the traditional notation.

3 Definition of Riemann Sum

Definition. If f is a real-valued function on $[a, b]$ by a Riemann sum for f corresponding to the given partition is meant a sum

$$\sum_{i=1}^N f(x'_i)(x_i - x_{i-1})$$

where $x_{i-1} \leq x'_i \leq x_i$ for each $i = 1, 2, \dots, N$.

Originally the following formalization was attempted:

```

Riemann_sum(a:T,b:{x:T|a<x},P:partition(a,b),f:[T->real]): real =
  LET xx = seq(P), N = length(P)-1 IN
  sigma[upto(N)](1,N,(LAMBDA (n: upto(N)):
    f(x_in(xx(n-1),xx(n)))*(xx(n)-xx(n-1))))

```

using `sigma` from the `reals` library and defining `x_in` as

```
x_in(aa:T,bb:{x:T|aa<x}): {t: T | aa <= t AND t <= bb}
```

there were two problems with this formulation, one minor and one serious. First, this will not typecheck (unprovable TCC) because the typechecker does not know that `sigma` function will not evaluate `xx(n)` outside of the range 1 to `N`. The value of `n-1` in `xx(n-1)` will actually never go negative, but PVS has no way of knowing this. Second, this definition is deficient in that the value of x' on which `f` is evaluated is provided by `x_in`. Although it is in some sense *arbitrary*, there is no real quantifier here. The proof of the integral split theorem requires that we quantify over all possible values of x' . See section 6 for more details. Thus, we define the following predicate on partitions and sequences:

```
xis?(a:T,b:{x:T|a<x},P:partition(a,b))
  (fs: [below(length(P)-1) -> closed_interval(a,b)]): bool =
  (FORALL (ii: below(length(P)-1)):
    P(ii) <= fs(ii) AND fs(ii) <= P(ii+1))
```

Given a sequence of x values, this predicate is true iff the i th value is contained in the i th section of the partition.

Now we can define a Riemann sum as follows:

```
Rie_sum(a:T,b:{x:T|a<x},P:partition(a,b),
  xis: (xis?(a,b,P)),f:[T->real]): real =
  LET N = length(P)-1 IN
  sigma[below(N)](0,N-1,(LAMBDA (n: below(N)):
    (P(n+1) - P(n)) * f(xis(n))))
```

We note that the index is taken from 0 to `N-1` to solve the typechecking problem and the fourth parameter `xis` provides the values of x' on which the height of each rectangle can be calculated: $f(x')$.

It is also convenient to define a predicate that checks whether a particular value equals `Rie_sum(a,b,P,xis,f)`:

```
Riemann_sum?(a:T,b:{x:T|a<x},P:partition(a,b),f:[T->real])(S:real): bool =
  (EXISTS (xis: (xis?(a,b,P))): LET N = length(P)-1 IN
    S = Rie_sum(a,b,P,xis,f))
```

4 Definition of Riemann Integral

Maxwell Rosenlicht provide the following definition of an integral:

Definition. Let $a, b \in \mathfrak{R}, a < b$. Let f be a real-valued function on $[a, b]$. We say that f is Riemann integrable on $[a, b]$ if there exists a number $A \in \mathfrak{R}$ such that, for any $\epsilon > 0$, there exists a $\delta > 0$ such that $|S - A| < \epsilon$ whenever S is a Riemann sum for f corresponding to any partition of $[a, b]$ of width less than δ . In this case A is called the Riemann Integral of f between a and b and is denoted

$$\int_a^b f(x) dx$$

In other words, in order to establish that $\int_a^b f(x) dx = A$, we must show that for any given ϵ there exists a δ and a real number A such that no matter how we partition the interval, if the width of that partition is less than δ and S is the Riemann sum corresponding to that partition, we have $|S - A| < \epsilon$.

We begin the formulation of the Riemann integral, by defining the following predicate:

```
integral?(a:T,b:{x:T|a<x},f:[T->real],S:real): bool =
  (FORALL (epsi: posreal): (EXISTS (delta: posreal):
    (FORALL (P: partition(a,b)):
      width(a,b,P) < delta IMPLIES
        (FORALL (R: (Riemann_sum?(a,b,P,f))):
          abs(S - R) < epsi))))
```

From this definition we can construct a predicate `integrable?` and a function `integral` which is defined on `integrable?` functions:

```
integrable?(a:T,b:{x:T|a<x},f:[T->real]): bool =
  (EXISTS (S: real): integral?(a,b,f,S))

integral(a:T,b:{x:T|a<x}, ff: { f | integrable?(a,b,f)} ):
  {S: real | integral?(a,b,ff,S)}
```

The uniqueness of the integral was demonstrated in the proof of

```
integral_unique: LEMMA a < b AND integral?(a,b,f,A1) AND
  integral?(a,b,f,A2) IMPLIES A1 = A2
```

Thus, the return type of the function `integral` consists of only one possible value. From this we can easily prove

```
integral_def: LEMMA a < b IMPLIES
  ( (integrable?(a,b,f) AND integral(a,b,f) = s)
    IFF integral?(a,b,f,s) )
```

Next, we eliminate the restriction that $a < b$, as follows:

```
Integrable?(a:T,b:T,f:[T->real]): bool = (a = b) OR
  (a < b AND integrable?(a,b,f)) OR
  (b < a AND integrable?(b,a,f))
```

```
Integrable_funs(a,b): TYPE = { f | Integrable?(a,b,f)}
```

```
Integral?(a:T,b:T,f:[T->real],S:real): bool = (a = b AND S = 0) OR
  (a < b AND integral?(a,b,f,S))
```

```
Integral(a:T,b:T,f:Integrable_funs(a,b)): real =
  IF a = b THEN 0
  ELSIF a < b THEN integral(a,b,f)
  ELSE -integral(b,a,f)
  ENDIF
```

The names were capitalized to distinguish these functions from the more restricted ones.

All of the proofs were straight-forward. The total number of proof commands were just a little over 500 in number, including all of the typecheck condition proofs. The only long proof was `integral_unique` which required 102 proof steps.

5 Linearity Properties

Following Rosenlicht, the first properties of the integral that were proved were the linearity properties:

```
integral_const_fun: LEMMA a < b IMPLIES integrable?(a,b,const_fun(D))
                    AND integral(a, b, const_fun[T](D)) = D*(b-a)
```

```
integral_scal: LEMMA a < b AND integrable?(a,b,f) IMPLIES
              integrable?(a,b,D*f) AND
              integral(a,b,D*f) = D*integral(a,b,f)
```

```
integral_sum: LEMMA a < b AND integrable?(a,b,f) AND integrable?(a,b,g)
             IMPLIES
             integrable?(a,b,(LAMBDA x: f(x) + g(x))) AND
             integral(a,b,(LAMBDA x: f(x) + g(x))) =
             integral(a,b,f) + integral(a,b,g)
```

```
integral_diff: LEMMA a < b AND integrable?(a,b,f) AND integrable?(a,b,g)
             IMPLIES
             integrable?(a,b,(LAMBDA x: f(x) - g(x))) AND
             integral(a,b,(LAMBDA x: f(x) - g(x))) =
             integral(a,b,f) - integral(a,b,g)
```

These properties were then used to prove that non-negative functions have non-negative integrals:

```
integral_ge_0: LEMMA a < b AND integrable?(a,b,f) AND
              (FORALL (x: closed_interval(a,b)): f(x) >= 0) IMPLIES
              integral(a,b,f) >= 0
```

Size of proofs:

lemma	Proof Buffer Size
<code>integral_const_fun</code>	35 lines
<code>integral_scal</code>	85 lines
<code>integral_sum</code>	85 lines
<code>integral_diff</code>	37 lines
<code>integral_ge_0</code>	94 lines

All of these proofs were easy. However the following simple property (example 2 on page 114 of Rosenlicht):

```
integral_jump: LEMMA a < b AND a <= z AND z <= b AND f(z) = cc AND
  (FORALL x: x /= z IMPLIES f(x) = 0) IMPLIES
  integrable?(a,b,f) AND integral(a,b,f) = 0
```

required 602 proof lines. Lemmas `integral_sum` and `integral_jump` were then used prove the following lemma in 36 steps:

```
integral_chg_one_pt: LEMMA a < b IMPLIES
  FORALL y: a <= y AND y <= b AND
  integrable?(a,b,f)
  IMPLIES integrable?(a,b,f WITH [(y) := yv]) AND
  integral(a,b,f) = integral(a,b,f WITH [(y) := yv])
```

which shows that if you change a function at one point, then its integral does not change. Of course this could be generalized to show that one can change a countably infinite number of points and not change the value of the integral, but this proof has not been attempted in the PVS system.

The following lemma (named Lemma 1 on page 118 of Rosenlicht) required 338 PVS proof lines:

```
integrable_lem: THEOREM a < b IMPLIES
  (integrable?(a,b,f) IFF
    (FORALL (epsi: posreal): (EXISTS (delta: posreal):
      (FORALL (P1,P2: partition(a,b)):
        width(a,b,P1) < delta AND
        width(a,b,P2) < delta IMPLIES
          (FORALL (RS1: (Riemann_sum?(a,b,P1,f)),
            RS2: (Riemann_sum?(a,b,P2,f)))):
            abs(RS1 - RS2) < epsi )))))
```

This was Rosenlicht's first major building block for the more difficult theorems. His next step was to develop the necessary apparatus to integrate step functions.

6 Step Functions

Establishing the key properties for integrals involving step functions proved more difficult than was expected. Surprisingly, some of the most difficult challenges occurred in places where visually the proofs were easy to see.

The first step was to provide a definition for a step function. This was accomplished by exploiting the machinery we had already constructed for partitions. We define a predicate that returns true iff the function is constant on the sub-intervals of the partition:

```

step_function_on?(a:T,b:{x:T|a<x},f:[T->real], P: partition[T](a,b)): bool =
  Let N = length(P), xx = seq(P) IN
  (FORALL (ii: below(N-1)): (EXISTS (fv: real):
    (FORALL (x: open_interval[T](xx(ii),xx(ii+1))):
      f(x) = fv)))

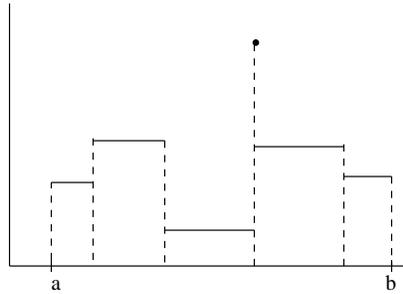
```

Then we define a “step function” to be a function for which there exists a partition for which `step_function_on?` holds:

```

step_function?(a:T,b:{x:T|a<x},f:[T -> real]): bool
  = (EXISTS (P: partition(a,b)): step_function_on?(a,b,f,P))

```



The first step function that was solved was the simple “square wave”:

```

Example_3: LEMMA a <= x1 AND x1 < xh AND xh <= b AND
  (FORALL z: (IF x1 < z AND z < xh THEN f(z) = 1
    ELSE f(z) = 0 ENDIF))
  IMPLIES integrable?(a,b,f) AND
    integral(a,b,f) = xh-x1

```

This was example 3 on page 114. The proof of this lemma in Rosenlicht’s text was 27 lines, the PVS proof was surprisingly difficult requiring 640 steps.

By showing that a step function is equivalent to a finite sum of these square wave functions, Rosenlicht’s Lemma 2 (page 119) was proved:

Lemma. *A step function is integrable. In particular, if x_0, x_1, \dots, x_n is a partition of the interval $[a, b]$, if $c_1, \dots, c_n \in \mathfrak{R}$ and if $f : [a, b] \rightarrow \mathfrak{R}$ is such that $f(x) = c_i$, if $x_{i-1} < x < x_i$ for $i = 1, \dots, n$, then*

$$\int_a^b f(x) dx = \sum_{i=1}^n c_i (x_i - x_{i-1})$$

In PVS we have:

```

step_function_integrable?: LEMMA a < b AND step_function?(a,b,f) IMPLIES
  integrable?(a,b,f)

```

```

step_function_on_integral: LEMMA a < b IMPLIES
  FORALL (P: partition[T](a,b)):

```

```

      step_function_on?(a,b,f,P) IMPLIES
integral(a,b,f) =
  LET N = length(P) IN
  sigma(0,N-2,(LAMBDA (i: below(N-1)):
    val_in(a,b,P,i,f)*(P(i+1) - P(i)))) ;

```

where $\text{val_in}(a,b,P,i,f)$ is just the value of f in the i th section of the partition P . In PVS this was defined as follows:

```

pick(a:T,b:{x:T|a<x},(P: partition[T](a,b)),j: below(length(P)-1)):
  {t:T | seq(P)(j) < t AND t < seq(P)(j + 1)} =
  choose({t:T | seq(P)(j) < t AND t < seq(P)(j + 1)})

val_in(a:T,b:{x:T|a<x},(P: partition[T](a,b)),j: below(length(P)-1),f): real
  = f(pick(a,b,P,j))

```

Establishing `step_function_on_integral` required the proof of 8 supporting lemmas and over 2500 proof steps.

The next key result proved in Rosenlicht is the following proposition from page 120:

Lemma. : *The real-valued function f on the interval $[a, b]$ is integrable on $[a, b]$ if and only for each $\epsilon > 0$ there exists step functions f_1, f_2 on $[a, b]$ such that*

$$f_1(x) \leq f(x) \leq f_2(x) \text{ for each } x \in [a, b]$$

and

$$\int_a^b (f_2(x) - f_1(x)) dx < \epsilon$$

This result, which he simply labeled as a proposition, required $2\frac{1}{2}$ pages in his book. The PVS proof was accomplished in two steps. First the forward direction was established:

```

step_to_integrable: LEMMA a < b AND      % Rosenlicht pg 120 forward direction
  (FORALL (eps: posreal):
    (EXISTS (f1,f2: [T -> real]):
      step_function?(a,b,f1) AND step_function?(a,b,f2)
      AND (FORALL (xx: closed_interval(a,b)):
        f1(xx) <= f(xx) AND f(xx) <= f2(xx))
      AND integrable?(a,b,f2-f1)
      AND integral(a,b,f2-f1) < eps))
  IMPLIES integrable?(a,b,f)

```

This proof was straight-forward and required only 146 PVS proof steps. However, the reverse direction was very difficult. The proof of the reverse direction took about 3 man weeks of effort. First it was necessary to establish that an integrable function is bounded (600 proof steps):

```

integrable_bounded: LEMMA a < b AND % Rosenlicht pg 122
                    integrable?(a,b,f)
                    IMPLIES bounded_on?(a, b, f)

```

where `bounded_on?(a, b, f)` was defined as

```

bounded_on?(a,b,f): bool = (EXISTS (B: real):
    (FORALL (x: (closed_intv(a,b))): abs(f(x)) <= B))

```

Then an additional 960 proof steps were necessary to finish the reverse direction.

The next main result proved was that a continuous function is integrable (Pg 123 Rosenlicht)

Theorem. *If f is a continuous real-valued function on the interval $[a, b]$ then $\int_a^b f(x) dx$ exists.*

The PVS version is:

```

continuous_integrable: LEMMA a < b AND
    (FORALL (x: closed_interval(a,b)): continuous(f,x))
    IMPLIES integrable?(a,b,f)

```

The proof given in Rosenlicht is 16 lines while the formal PVS proof is over 1200 lines long. See the appendix for discussion of why there is such a large difference between PVS proofs and traditional mathematical rigor.

The integral split theorem was probably the most difficult result to achieve in PVS. Here is its statement in Rosenlicht (page 123):

Theorem. *Let $a, b, c \in \mathfrak{R}$, $a < b < c$, and let f be a real-valued function on $[a, c]$. The f is integrable on $[a, c]$ if and only if it is integrable on both $[a, b]$ and $[b, c]$, in which case*

$$\int_a^b f(x) dx + \int_b^c f(x) dx = \int_a^c f(x) dx$$

This theorem is easily stated in PVS as follows:

```

integral_split: THEOREM a < b AND b < c AND
    integrable?(a,b,f) AND
    integrable?(b,c,f)
    IMPLIES integrable?(a,c,f) AND
    integral(a,b,f) + integral(b,c,f) = integral(a,c,f)

```

Although the rigorous proof in Rosenlicht was only 2 pages, the PVS proof required the use of 131 lemmas, whose proofs required over 4000 PVS proof commands. As explained in section 3, the Riemann sum was originally defined using `x_in`:

```

x_in(aa:T,bb:{x:T|aa<x}): {t: T | aa <= t AND t <= bb}

```

to select the x_i s within each subinterval in the partition. Using this definition all of the theorems were completed except the integral split theorem. In the first version of the library, the integral split theorem was included as an axiom, because the time required to prove this lemma was deemed prohibitive at that time. Using this axiom and the other proven lemmas the fundamental theorem was completed. The library was released with the expectation that this integral split lemma would be proved later.

The first indication of a problem with this definition of a Riemann sum, was in an email from David Lester of Manchester University. He pointed out that this definition does not allow you to establish that an integrable function was bounded¹. After receiving his email, the definition was revised, and all of the other lemmas were reproved using the new definition, and finally this integral split theorem was also completed. There are now no axioms in this PVS library.

7 Fundamental Theorem of Calculus

The culmination of this work was the completion of the Fundamental Theorem of Calculus in the PVS theorem prover. The statement of this theorem in PVS is:

```

fundamental: THEOREM continuous(f) AND
              (FORALL x: F(x) = Integral(a,x,f))
              IMPLIES derivable(F) AND deriv(F) = f

```

where `derivable` and `deriv` are defined in the differential calculus part of the analysis library that had been previously developed by Bruno Dutertre of the Royal Holloway & Bedford New College (now at SRI International). These define derivability (i.e. differentiability) and the derivative respectively.

The PVS proof chain analyzer reports that the final completed proof of fundamental depends upon 978 proven lemmas and theorems. The following corollaries were also completed:

```

fundamental2: THEOREM continuous(f)
              IMPLIES (EXISTS F: derivable(F) AND deriv(F) = f)
fundamental3: THEOREM derivable(F) AND deriv(F) = f AND continuous(f)
              IMPLIES Integral(a,b,f) = F(b) - F(a)

```

Next the concept of the antiderivative was formulated as follows:

```

antiderivative?(F,f): bool = derivable(F) AND deriv(F) = f

antiderivative_lem: LEMMA antiderivative?(F,f) AND derivable(G) AND
                      deriv(G) = f
                      IMPLIES (EXISTS (c: real): F = G + const_fun(c))

```

¹Here you have an integrable function and want to establish that f is bounded. It is not enough to pick an arbitrary x_i value in a interval i to compute the height of the rectangle $f(x_i)$. You need to exploit the fact that no matter what value of x_i you chose, $f(x_i)$ is sufficiently small if indeed the function is integrable. This requires explicit quantification of each x_i for all intervals in the partition.

A functional that returns the antiderivative was also provided:

```
antideriv(f: continuous_fun[T]): { gg: [T -> real] |  
                                   derivable(gg) AND deriv(gg) = f }
```

8 Conclusion

A formalization of the integral calculus in the PVS theorem prover has been completed. The theory and proofs were based on Rosenlicht's classic text on real analysis and follow the traditional epsilon-delta method. The goal of this work was to provide a practical set of PVS theories that could be used for verification of hybrid systems that arise in air traffic management systems and other aerospace applications. All of the basic linearity, integrability, boundedness, and continuity properties of the integral calculus were proven. The work culminated in the proof of the Fundamental Theorem Of Calculus. There is a brief discussion about why mechanically checked proofs are so much longer than standard mathematics textbook proofs.

A Why Are PVS Proofs So Much Larger?

Obviously providing a complete script of every formal step of one of the key lemmas in this library would require hundreds of pages and the details would be of no interest to the reader². Furthermore, these proof steps are easily viewed using PVS on our publically available libraries.

Therefore, I thought it might be of interest to the reader to offer some simple examples of how a completely formal proof differs from the traditional rigorous proof offered in a standard mathematics text. This presentation is in no way comprehensive – it highlights only a fraction of the complexities one faces in a completely formal theorem prover.

A.1 Formalizing Partitions

In a classic text book, the following suffices to define a partition of an interval:

$$a = x_0 < x_1 < x_2 \dots < x_N = b$$

But in a formal system this must be represented as a finite sequence of real numbers:

```
partition(a:T,b:{x:T|a<x}): TYPE =
  {fs: finite_sequence[closed_interval(a,b)] | .... }
```

Formally, x_i is $\text{seq}(P)(i)$ where P is a partition. But that is only a superficial difference. The real problem comes from all of the properties one implicitly knows about these x_i s:

```
seq(P)(0) = a AND
seq(P)(N-1) = b AND
(FORALL (ii: below(N-1)): seq(P)(ii) < seq(P)(ii+1))}
```

which are defined in the “...” part of the definition above. From $i < j$ it is obvious that $x_i < x_j$, but in PVS whenever one needs this property it has to be brought into the proof manually as a lemma:

```
parts_order : LEMMA FORALL (P: partition(a,b), ii,jj: below(length(P))):
  ii < jj IMPLIES seq(P)(ii) < seq(P)(jj)
```

Even the trivial property that if $a \leq x \leq b$, then x must be in one of the subintervals, say i , requires the use of the following lemma

```
part_in : LEMMA FORALL (P: partition(a,b)):
  a < b AND a <= x AND x <= b IMPLIES
  (EXISTS (ii: below(length(P)-1)):
    seq(P)(ii) <= x AND x <= seq(P)(ii+1))
```

which first must be proved by induction. If you need the trivial property that if x is inside subinterval i , then it is not in another subinterval j , you must bring in the following lemma:

²A typical proof of say 100 proof steps, produces over 4000 lines of proof trace when each step is replayed.

```

parts_disjoint: LEMMA FORALL (P: partition(a,b), ii,jj: below(length(P)-1)):
    seq(P)(ii) < x AND x < seq(P)(1 + ii) AND
    seq(P)(jj) < x AND x < seq(P)(1 + jj)
    IMPLIES
        jj = ii

```

If you have established that something is true for an arbitrary subinterval i , and you want to conclude that it is therefore true for all of $[a, b]$, you need to reference:

```

Prop: VAR [T -> bool]
part_induction: LEMMA (FORALL (P: partition(a,b)):
    (FORALL ( x: closed_interval(a,b)):
        LET xx = seq(P), N = length(P) IN
        (FORALL (ii : below(N-1)):
            xx(ii) <= x AND x <= xx(ii+1) IMPLIES
                Prop(x))
        IMPLIES Prop(x) ) )

```

and manually instantiate the property of interest. Clearly, this adds a tremendous amount of time-consuming, tedious work.

In general constructs such as x_0, x_1, \dots, x_N inevitably lead to inductions, the details of which mathematicians such as Rosenlicht rarely delve into.

A.2 Step Functions

There are many properties of step functions that are obviously true from a visual viewpoint, but require fairly time-consuming proofs in a mechanical theorem prover. For example, the property that if you add two step-functions, you get another step function is assumed without proof in Rosenlicht. However, the proof in PVS was surprisingly tedious:

```

sum_step_is_step: LEMMA a < b AND
    step_function?(a, b, f) AND
    step_function?(a, b, g)
    IMPLIES
        step_function?(a, b, f + g)

```

This lemma required over 350 proof steps and the construction of a function

```

UnionPart(a:T,b:{x:T|a<x},P1,P2: partition[T](a,b)): partition[T](a,b) =
    set2part(union(part2set(a, b, P1), part2set(a, b, P2)))

```

that generates a new partition containing all of the x_i s from the two step functions being added together. All of the trivial properties such as the fact that if an x_i is a discontinuity point on one of the original step functions then it is also one of the discontinuity points in the generated one must be manually introduced into the proof in order to be used. The obvious property that the n th sub interval of the new partition must be contained within some sub interval of the original partitions is

```

Union_lem: LEMMA FORALL (a:T, b: {x:T|a<x}, P1,P2: partition[T](a, b),
                        n: below(length(UnionPart(a,b,P1,P2))-1)):
  in_sect?(a,b,UnionPart(a,b,P1,P2),n,x)
  IMPLIES
  (EXISTS (k: below(length(P1)-1)):
    seq(P1)(k) <= UnionPart(a,b,P1,P2)(n) AND
    UnionPart(a,b,P1,P2)(n+1) <= seq(P1)(k+1) )

```

This property requires a tricky proof using the following maximum:

```

"max[length(P1!1) - 1] ({k: below(length(P1!1) - 1) |
  seq(P1!1)(k) <= UnionPart(a!1, b!1, P1!1, P2!1)'seq(n!1)})"

```

i.e, the largest subinterval index less than n . Similar proofs were needed for the difference of two partitions, the concatenation of two partitions and several other constructions involving step functions.

A.3 Complications Due To Working In Type Theory Rather Than Set Theory

One of the most disturbing things about working in type theory rather than set theory is that standard operators such as Σ , are not unique. There are different versions depending upon the domain of the function being summed. For example the summation operator over functions from `[nat -> real]` is `sigma[nat]` whereas the operator for functions from `[upto[N] -> real]` is `sigma[upto[N]]` and they are not interchangeable even though `upto[N]` is a subtype of `nat`.

Also restrictions of function domains to subdomains can lead to ugliness involving the PVS prelude `restrict/extend` functions For example

```

continuous[closed_interval[T](seq(PP)(ii!1), seq(PP)(1 + ii!1))]
  (restrict[T,
    closed_interval[T](seq(PP)(ii!1), seq(PP)(1 + ii!1)),
    real]
  (f!1),
  x!1)

```

in addition to a proliferation of different versions of `continuous` e.g. `continuous[T](f)`, `continuous[closed_interval(a,b)](f)`, and `continuous[closed_interval[T](seq(PP)(ii!1), seq(PP)(1 + ii!1))]`.

B Illustration

As noted before, the complete presentation of a PVS proof of a lemma would require dozens of pages and likely to be of no real interest to the reader. Therefore in this section I merely provide a proof sketch of a theorem from page 123 of Rosenlicht.

B.1 Continuous Function Is Integrable

Theorem. *If f is a continuous real-valued function on the interval $[a, b]$ then $\int_a^b f(x) dx$ exists.*

Proof. We shall prove this theorem by showing that the criterion of the preceding lemma obtains. Since f is uniformly continuous on $[a, b]$, given any $\epsilon > 0$ we can find a δ such that whenever $x, x'' \in [a, b]$ and $|x' - x''| < \delta$ then $|f(x') - f(x'')| < \epsilon/(b-a)$. Choose any partition x_0, x_1, \dots, x_N of $[a, b]$ of width less than δ . For each $i = 1, \dots, N$ choose $x'_i, x''_i \in [x_{i-1}, x_i]$ such that the restriction of f to $[x_{i-1}, x_i]$ attains a minimum at a'_i and a maximum at x''_i . Define step functions f_1, f_2 on $[a, b]$ by

$$f_1(x) = \begin{cases} f(x'_i) & \text{if } x_{i-1} < x < x_i, i = 1, \dots, N \\ f(x) & \text{if } x = x_i, i = 0, 1, \dots, N \end{cases}$$

$$f_2(x) = \begin{cases} f(x''_i) & \text{if } x_{i-1} < x < x_i, i = 1, \dots, N \\ f(x) & \text{if } x = x_i, i = 0, 1, \dots, N \end{cases}$$

Then $f_1(x) \leq f(x) \leq f_2(x)$ for all $x \in [a, b]$. Furthermore for each $i = 1, \dots, N$ we have $|x'_i - x''_i| \leq x_i - x_{i-1} < \delta$, so that $|f(x'_i) - f(x''_i)| < \epsilon/(b-a)$ and therefore $f_2(x) - f_1(x) \leq \epsilon/(b-a)$ for all $x \in [a, b]$. Therefore

$$\begin{aligned} \int_a^b (f_2(x) - f_1(x)) dx &< \max\{f_2(x) - f_1(x) : x \in [a, b]\} \cdot (b - a) \\ &< \frac{\epsilon}{b - a} \cdot (b - a) = \epsilon \end{aligned}$$

□

B.2 Formal Proof Sketch of This Theorem

The Informal proof is 16 lines in Rosenlicht. The Formal PVS proof is over 1200 lines long (not counting the auxiliary lemmas and TCCs). The PVS proof script (i.e. M-x edit-proof) is 417 command lines and the proof trace is over 8000 lines long. Here are some highlights of this formal proof. The formal proof begins with

```
{-1}  a!1 < b!1
{-2}  FORALL (x: closed_interval(a!1, b!1)): continuous(f!1, x)
      |-----
{1}   integrable?(a!1, b!1, f!1)
```

We use a lemma that establishes that $f!1$ is uniformly continuous and obtain

```
[-1]  uniformly_continuous?(LAMBDA (s: closed_interval[T] (a!1, b!1)):
      f!1(s),
      LAMBDA (t: real):
      IF T_pred(t) AND a!1 <= t AND t <= b!1
```

```

        THEN TRUE
        ELSE FALSE
    ENDIF)

```

```

[-2]  a!1 < b!1
[-3]  FORALL (x: closed_interval(a!1, b!1)): continuous(f!1, x)
      |-----
[1]   integrable?(a!1, b!1, f!1)

```

We rewrite with the lemma `step_to_integrable` and the goal becomes:

```

[1]   EXISTS (f1, f2: [T -> real]):
      step_function?(a!1, b!1, f1) AND step_function?(a!1, b!1, f2)
      AND FORALL (xx: closed_interval[T](a!1, b!1)):
          f1(xx) <= f!1(xx) AND f!1(xx) <= f2(xx)
      AND integrable?(a!1, b!1, f2 - f1)
      AND integral(a!1, b!1, f2 - f1) < eps!1

```

We instantiate `f1` and `f2` with `fmin` and `fmax` defined as follows

```

min_x(a:T,b:{x:T|a<x}, f: fun_cont_on(a,b)):
  {mx: T | a <= mx AND mx <= b AND
    (FORALL (x: T): a <= x AND x <= b IMPLIES
      f(mx) <= f(x))}

```

```

max_x(a:T,b:{x:T|a<x}, f: fun_cont_on(a,b)):
  {mx: T | a <= mx AND mx <= b AND
    (FORALL (x: T): a <= x AND x <= b IMPLIES
      f(mx) >= f(x))}

```

```

fmin(a:T,b:{x:T|a<x},P: partition(a,b), f: fun_cont_on(a,b)):
  {ff: [T -> real] | LET xx = seq(P) IN
    FORALL (ii : below(length(P)-1)):
      FORALL (x: T): (xx(ii) < x AND x < xx(ii+1) IMPLIES
        ff(x) = f(min_x(xx(ii),xx(ii+1),f))) AND
        ((xx(ii) = x OR x = xx(ii+1)) IMPLIES
          ff(x) = f(x))}

```

```

fmax(a:T,b:{x:T|a<x},P: partition(a,b), f: fun_cont_on(a,b)):
  {ff: [T -> real] | LET xx = seq(P) IN
    FORALL (ii : below(length(P)-1)):
      FORALL (x: T): (xx(ii) < x AND x < xx(ii+1) IMPLIES
        ff(x) = f(max_x(xx(ii),xx(ii+1),f))) AND
        ((xx(ii) = x OR x = xx(ii+1)) IMPLIES
          ff(x) = f(x))}

```

These leaves us with the following goal:

```

{1}      step_function?(a!1, b!1, fmin(a!1, b!1, PP, f!1))
AND step_function?(a!1, b!1, fmax(a!1, b!1, PP, f!1))
AND FORALL (xx: closed_interval[T](a!1, b!1)):
      fmin(a!1, b!1, PP, f!1)(xx) <= f!1(xx) AND
      f!1(xx) <= fmax(a!1, b!1, PP, f!1)(xx)
AND integrable?(a!1, b!1,
      fmax(a!1, b!1, PP, f!1) - fmin(a!1, b!1, PP, f!1))
AND integral(a!1, b!1,
      fmax(a!1, b!1, PP, f!1) - fmin(a!1, b!1, PP, f!1))
      < eps!1

```

Next, we demonstrate that `fmin` and `fmax` are indeed step functions. And then seek to establish each of the remaining conjuncts. Lets look at just one of the obligations:

```
fmin(a!1, b!1, PP, f!1)(xx!1) <= f!1(xx!1)
```

To prove this we get the definition of `fmin` and bring in `part_induction`:

```

{-1}  FORALL (Prop: [T -> bool], a, b: T, P: partition[T](a, b),
      x: closed_interval[T](a, b)):
      LET xx: [below[P'length] -> closed_interval[T](a, b)] = seq(P),
          N = length(P)
      IN
      (FORALL (ii: below(N - 1)):
        xx(ii) <= x AND x <= xx(ii + 1) IMPLIES Prop(x))
      IMPLIES Prop(x)
[-2]  FORALL (ii: below(length(PP) - 1)):
      FORALL (x: T):
      (seq(PP)(ii) < x AND x < seq(PP)(1 + ii) IMPLIES
        fmin(a!1, b!1, PP, f!1)(x) =
          f!1(min_x[T](seq(PP)(ii), seq(PP)(1 + ii), f!1)))
      AND
      ((seq(PP)(ii) = x OR x = seq(PP)(1 + ii)) IMPLIES
        fmin(a!1, b!1, PP, f!1)(x) = f!1(x))
[-3]  a!1 < b!1
      |-----
[1]   fmin(a!1, b!1, PP, f!1)(xx!1) <= f!1(xx!1)

```

Provide the property for the induction

```

(inst -1 "(LAMBDA x: fmin(a!1, b!1, PP, f!1)(x) <= f!1(x))"
  "a!1" "b!1" "PP" "xx!1")

```

obtaining

```

[-1]  seq(PP)(ii!1) <= xx!1
[-2]  xx!1 <= seq(PP)(1 + ii!1)

```

```

{-3} (seq(PP)(ii!1) < xx!1 AND xx!1 < seq(PP)(1 + ii!1) IMPLIES
      fmin(a!1, b!1, PP, f!1)(xx!1) =
      f!1(min_x[T](seq(PP)(ii!1), seq(PP)(1 + ii!1), f!1)))
      AND
      ((seq(PP)(ii!1) = xx!1 OR xx!1 = seq(PP)(1 + ii!1)) IMPLIES
      fmin(a!1, b!1, PP, f!1)(xx!1) = f!1(xx!1))
[-4] a!1 < b!1
      |-----
[1]  fmin(a!1, b!1, PP, f!1)(xx!1) <= f!1(xx!1)

```

If $\text{seq}(PP)(ii!1) < xx!1$ then the result follows quickly from the definition of min_x . But we have to also deal with the cases where $\text{seq}(PP)(ii!1) = xx!1$ or $\text{seq}(PP)(1+ii!1) = xx!1$ which I will pass over here.

Once we have established the integrability of

$$f_{\max}(a!1, b!1, PP, f!1) - f_{\min}(a!1, b!1, PP, f!1)$$

we need to establish:

$$\text{integral}(a!1, b!1, f_{\max}(a!1, b!1, PP, f!1) - f_{\min}(a!1, b!1, PP, f!1)) < \text{eps!1}.$$

In the prover, we have:

```

[-1] integrable?(a!1, b!1,
      fmax(a!1, b!1, PP, f!1) - fmin(a!1, b!1, PP, f!1))
[-2] step_function?(a!1, b!1, fmax(a!1, b!1, PP, f!1))
[-3] step_function?(a!1, b!1, fmin(a!1, b!1, PP, f!1))
[-4] eq_partition(a!1, b!1, 2 + floor((b!1 - a!1) / delta!1)) = PP
[-5] FORALL (x,
      y:
      (LAMBDA (t: real):
      IF T_pred(t) AND a!1 <= t AND t <= b!1 THEN TRUE
      ELSE FALSE
      ENDIF)):
      abs(x - y) < delta!1 IMPLIES
      abs(f!1(x) - f!1(y)) < (eps!1 / 2) / (b!1 - a!1)
[-6] a!1 < b!1
[-7] FORALL (x: closed_interval(a!1, b!1)): continuous(f!1, x)
      |-----
{1}  integral(a!1, b!1, fmax(a!1, b!1, PP, f!1) - fmin(a!1, b!1, PP, f!1))
      < eps!1

```

Using the a lemma about partitions with equal sub-intervals, we have

```

{-1} width(a!1, b!1,
      eq_partition(a!1, b!1, 2 + floor((b!1 - a!1) / delta!1)))
      = (b!1 - a!1) / (1 + floor((b!1 - a!1) / delta!1))

```

with some algebraic manipulations we obtain:

```

[-2] width(a!1, b!1, PP) < delta!1

```

Using lemma `integral_bound_abs` we get

```

{-1} (a!1 < b!1 AND
      integrable?(a!1, b!1,
                  fmax(a!1, b!1, PP, f!1) - fmin(a!1, b!1, PP, f!1))
      AND
      (FORALL (x: closed_interval[T](a!1, b!1)):
        abs((fmax(a!1, b!1, PP, f!1) - fmin(a!1, b!1, PP, f!1))(x)) <=
          eps!1 / (2 * (b!1 - a!1))))
      IMPLIES
      abs(integral(a!1, b!1,
                  fmax(a!1, b!1, PP, f!1) - fmin(a!1, b!1, PP, f!1)))
        <= eps!1 / (2 * (b!1 - a!1)) * (b!1 - a!1)
[-2] width(a!1, b!1, PP) < delta!1
[-3] integrable?(a!1, b!1,
                  fmax(a!1, b!1, PP, f!1) - fmin(a!1, b!1, PP, f!1))
[-4] step_function?(a!1, b!1, fmax(a!1, b!1, PP, f!1))
[-5] step_function?(a!1, b!1, fmin(a!1, b!1, PP, f!1))
[-6] FORALL (x,
            y:
              (LAMBDA (t: real):
                IF T_pred(t) AND a!1 <= t AND t <= b!1 THEN TRUE
                ELSE FALSE
                ENDIF)):
      abs(x - y) < delta!1 IMPLIES
      abs(f!1(x) - f!1(y)) < (eps!1 / 2) / (b!1 - a!1)
[-7] a!1 < b!1
[-8] FORALL (x: closed_interval(a!1, b!1)): continuous(f!1, x)
|-----
[1]  integral(a!1, b!1, fmax(a!1, b!1, PP, f!1) - fmin(a!1, b!1, PP, f!1))
    < eps!1
[2]  integrable?(a!1, b!1, f!1)

```

The uniform continuity result [-6] is used to establish

```

abs(MAX_x - MIN_x) < delta!1 IMPLIES
abs(f!1(MAX_x) - f!1(MIN_x)) < (eps!1 / 2) / BMA

```

(Note: this occurs as the following subcase:

```

continuous_integrable.1.1.1.1.1.1.1.1.1.2.1.1.1.2.1.1.2.1.1.1.1.1 )

```

Further manipulation enables us to simply (-1) to:

```

{-1} abs(integral(a!1, b!1,
                  fmax(a!1, b!1, PP, f!1) - fmin(a!1, b!1, PP, f!1)))
      <= eps!1 / (2 * BMA) * BMA

```

where `BMA = "b!1-a!1`, from which the subgoal follows from properties about `abs`. The complete proof trace is 8000 lines long.

C Example of Proof Deficiency in Rosenlicht

The proofs in Rosenlicht are remarkably complete and well documented. Nevertheless, it is not unusual to encounter special cases that are not covered in the book. Here is an example of such a deficiency taken from page 114 and 115.

Theorem. Let $\alpha, \beta \in [a, b]$ with $\alpha < \beta$. Let $f : [a, b] \rightarrow \mathfrak{R}$ be defined by

$$f(x) = \begin{cases} 1 & \text{if } x \in (\alpha, \beta) \\ 0 & \text{if } x \in [a, b], x \notin (\alpha, \beta) \end{cases}$$

Then

$$\int_a^b f(x) dx = \beta - \alpha$$

Proof. Let x_0, x_1, \dots, x_N be a partition of $[a, b]$ of width less than δ and consider a Riemann sum for f corresponding to this partition, say

$$S = \sum_{i=1}^N f(x'_i)(x_i - x_{i-1})$$

where $x_{i-1} \leq x'_i \leq x_i$ for each $i = 1, 2, \dots, N$. Since $f(x'_i)$ is 1 or 0 according as the point x'_i is in the open interval (α, β) or not, we have

$$S = \sum^* (x_i - x_{i-1})$$

the asterisk indicating that we include in the sum only those i for which $x'_i \in (\alpha, \beta)$. Now choose p, q from among the $i = 1, 2, \dots, N$ such that

$$x_{p-1} \leq \alpha < x_p, \quad x_{q-1} < \beta \leq x_q$$

But this step overlooks the possibility that all of the x_i may fall outside of (α, β) . The proof is repairable, but this case must be dealt with explicitly. The PVS theorem prover required that all of the details of this case be supplied. However, these details were not included in the Rosenlicht text. Admittedly this would clutter up the text book, but a complete formal proof must cover it. Another special case is when N is 3 or less for which the above construction again fails. Later in the proof the following fact is used $p + 1 \leq q - 1$. But this is not possible if N is very small. \square

D User Guide

It is expected that most users of this formalization of the integral, will only need the theorems in two PVS theories: `integral` and `fundamental_theorem`. Here is a quick reference guide to these theorems:

Theorem	Lemma Name
$\int_a^a f(x) dx = 0$	Integral_a_to_a
$\int_a^b c dx = c * (b - a)$	Integral_const_fun
$\int_b^a f(x) dx = - \int_a^b f(x) dx$	Integral_rev
$\int_a^b cf(x) dx = c \int_a^b f(x) dx$	Integral_scal
$\int_a^b f(x) + g(x) dx = \int_a^b f(x) dx + \int_a^b g(x) dx$	Integral_sum
$\int_a^b f(x) - g(x) dx = \int_a^b f(x) dx - \int_a^b g(x) dx$	Integral_diff
$\int_a^b f$ WITH $[(y) := yv] dx = \int_a^b f dx$	Integral_chg_one_pt
$f(x) \geq 0 \supset \int_a^b f dx \geq 0$	Integral_ge_0
$ f(x) < M \supset \int_a^b f(x) dx \leq M * (b - a)$	Integral_bounded
f integrable $\supset f(x) < B$	Integrable_bounded
f continuous $\supset f$ integrable	continuous_Integrable?
$\int_a^b f(x) dx + \int_b^c f(x) dx = \int_a^c f(x) dx$	Integral_split
For step function $f : \int_a^b f(x) dx = \sum_{i=1}^n c_i(x_i - x_{i-1})$	step_function_on_integral
$F = \int_a^x dt \supset F' = f$	fundamental1
$\int_a^b f(t) dt = F(b) - F(a)$	fundamental3

The theory was first developed for integrals where $a < b$. These results are distributed over a number of PVS theories:

Theorem	Lemma Name	Theory
$\int_a^b c dx = c * (b - a)$	integral_const_fun	integral_prep
$\int_a^b cf(x) dx = c \int_a^b f(x) dx$	integral_scal	integral_prep
$\int_a^b f(x) + g(x) dx = \int_a^b f(x) dx + \int_a^b g(x) dx$	integral_sum	integral_prep
$\int_a^b f(x) - g(x) dx = \int_a^b f(x) dx - \int_a^b g(x) dx$	integral_diff	integral_prep
$\int_a^b f dx = \int_a^b f$ WITH $[(y) := yv] dx$	integral_chg_one_pt	integral_prep
$f(x) \geq 0 \supset \int_a^b f dx \geq 0$	integral_ge_0	integral_prep
f integrable $\supset f(x) < B$	integrable_bounded	integral_bounded
f continuous $\supset f$ integrable	continuous_integrable	integral_cont
$\int_a^b f(x) dx + \int_b^c f(x) dx = \int_a^c f(x) dx$	integral_split	integral_split
For step function $f : \int_a^b f(x) dx = \sum_{i=1}^n c_i(x_i - x_{i-1})$	step_function_on_integral	integral_step

All of the theories and proofs are available at

<http://shemesh.larc.nasa.gov/fm/ftp/larc/PVS-library/pvslib.html>

References

- [1] <ftp://info.mcs.anl.gov/pub/qed/manifesto>, May 2004.
- [2] <http://shemesh.larc.nasa.gov/fm/>, May 2004.
- [3] Carreño, V.; and Muñoz, C.: Aircraft Trajectory Modeling and Alerting Algorithm Verification. In *Theorem Proving in Higher Order Logics: 13th International Conference, TPHOLs 2000*, vol. 1869 of *Lecture Notes in Computer Science*, 2000, pp. 90–105. An earlier version appears as report NASA/CR-2000-210097 ICASE No. 2000-16.
- [4] Carreno, Victor A.; Gottliebsen, Hanne; Butler, Ricky; and Kalvala, Sara: *Formal Modeling and Analysis of a Preliminary Small Aircraft Transportation System (SATS) Concept*. NASA, NASA Technical Memorandum NASA/TM-2004-212999,, Mar. 2004.
- [5] Cruz-Filipe, L.: *Constructive Real Analysis: a Type-Theoretical Formalization and Applications*. PhD thesis, University of Nijmegen, Apr. 2004.
- [6] Dowek, Gilles; Munoz, Cesar; and Geser, Alfons: Tactical Conflict Detection and Resolution In A 3-D Airspace. In *4th USA/Europe Air Traffic Management R&D Seminar, Santa Fe*, Dec. 2001, pp. 3–7.
- [7] Dutertre, Bruno: Elements of mathematical analysis in PVS. In *Theorem Proving in Higher Order Logics: 9th International Conference, TPHOLs '96*, vol. 1125, Turku, Finland, 1996, Springer-Verlag, pp. 141–156.
- [8] Fleuriot, Jacques D.: On the Mechanization of Real Analysis in Isabelle/HOL. In *Theorem Proving in Higher Order Logics: 13th International Conference, TPHOLs 2000, Lecture Notes in Computer Science, volume 1869*, 2000. Available at <http://homepages.inf.ed.ac.uk/bundy/projects/phd/nsa.html>.
- [9] Harrison, John: *Theorem Proving with the Real Numbers*. Springer-Verlag, 1998.
- [10] Owre, Sam; Rushby, John M.; and Shankar, Natarajan: PVS: A Prototype Verification System. In Kapur, Deepak, editor 1992:, *11th International Conference on Automated Deduction (CADE)*, vol. 607 of *Lecture Notes in Artificial Intelligence*, Saratoga, NY, June 1992, Springer Verlag, pp. 748–752.
- [11] Richter, Stefan: <http://www-lti.informatik.rwth-aachen.de/~richter/papers/index.html>, Formalizing Integration Theory, with an Application to Probabilistic Algorithms, May 2003. Diploma Thesis, Technische Universitat Munchen, Department of Informatics.
- [12] Rosenlicht, Maxwell: *Introduction to Analysis*. Scott, Foresman and Company, 1968.
- [13] Shankar, N.; Owre, S.; and Rushby, J. M.: *The PVS Proof Checker: A Reference Manual (Beta Release)*. Computer Science Laboratory, SRI International, Menlo Park, CA, Feb. 1993.