TFG

TRABAJO FIN DE GRADO
Curso 2024/2025

UNIVERSIDAD COMPLUTENSE
MADRID

FACULTAD DE CIENCIAS MATEMÁTICAS

GRADO EN MATEMÁTICAS

DEPARTAMENTO DE SISTEMAS INFORMÁTICOS Y COMPUTACIÓN

Nombre del estudiante: Longjian Jiang
Nombre del tutor: Jorge Carmona Ruber

Madrid, 9 de julio de 2025

Abstract

This thesis is about formalizing mathematical proofs using LEAN4. The proof of the Inverse Function Theorem in $\mathbb{R}$ will be the main example to illustrate the formalization.

Índice general

INTRODUCCIÓN

En matemáticas, para verificar una demostración puede ser muy costosa. Con lo que apareció el problema de demostrar los teoremas de forma automática utilizando ordenadores. LEAN4 es un asistente de pruebas interactivo y un lenguaje de programación funcional diseñado para dicho propósito, es decir, una vez que pruebemos el resultado en LEAN4, no tenemos que preocupar por su veracidad, ya que está garantizada. Además, con el rápido avance de la Inteligencia Artificial, surgió la ambición de que la demostración pueda ser automática.

El presente trabajo trata de introducir el marco teórico en el que se basa LEAN4, y unos ejemplos prácticos de cómo formalizar resultados matemáticos con la ayuda de LEAN4. El principal resultado será el Teorema de la función inversa en $\mathbb{R}$. Para ello, es conveniente familizarse con el lenguaje LEAN4 y con la librería Mathlib. Asimismo, en muchas ocasiones, hay dos formas de hacer pruebas, una de ellas consiste usar las tácticas (teoremas probados) existentes en Mathlib, y otra posibilidad sería demostrar el resultado elaborarando nuestras propias definiciones. Cabe destacar una tercera posibilidad mezclando las dos formas anteriores.

En cuanto a la estructura, el documento comenzará con un capítulo dedicado a explicar los fundamentos del lenguaje LEAN4

Capítulo 1

Los objetivos y el plan de trabajo

El presente trabajo trata de comprender cómo usar LEAN4 para probar resultados matemáticos. Para ello, vamos a ver en el siguiente capítulo la base teórica de LEAN4. Tras haber adquirido cierta base teórica, se presenta unos ejercicios simples realizados con LEAN4 para ilustrar la teoría expuesta. Más adelante, se muestra numerosas tacticas importantes para las demostraciones realizadas.

Posteriormente, se encuentra un capítulo en el que trata de analizar un ejemplo realizado por el autor durante el curso, que es más complejo. En concreto, consiste en demostrar el Teorema de la Función Inversa para $\mathbb{R}$. Además, la demostración está hecha de dos maneras, una utilizando las tácticas, y otra bansandose en nuestras propias definiciones que requiere probar la equivlancia con las de Mathlib.

Asimismo, se va a explicar las diferencias entre los dos métodos, y las conexiones que hay entre ellos.

Por último, se reflexiona sobre todo el contenido de la presente memoria y lo trabajado durante el curso. De esta manera, se concluye el proyecto.

Capítulo 2

Marco teórico de LEAN4

Para poder enlazar con las tareas prácticas, este capítulo se centra en explicar la base teórica. Además, se ilustrará con algunos ejemplos simples para facilitar la comprensión.

2.1. Fundamento teórico

LEAN4 se basa en una versión de teoría de tipos dependientes llamado calculo de construcciones, con una jerarquía contable de universos no cumulativos y tipos inductivos. Para comprender mejor su significado, veamos las siguientes nociones.

2.1.1. Teoría de tipos simple

La teoría de tipos clasifica los objetos según su tipo. Por ejemplo, en el siguiente contexto, **X** representa un número real y **F** una función de $\mathbb{R}$ en $\mathbb{R}$.

```
def X : Real := 2.3
def f : Real → Real :=  x => 2 * x
```

Ejemplo 2.1: Declaración de variables en Lean 4.

Cabe destacar que en LEAN, los propios tipos son objetos, y cada uno de ellos tiene un tipo.

```
def a : Type := Nat
def b : Type := Bool
#check a    -- Type
#check b    -- Type
```

Ejemplo 2.2: Declaración de tipos como objetos en Lean 4.

El comando #check pregunta a LEAN de qué tipo es el objeto, y el comando #eval hace la evaluación de la expresión dada.

Además, podemos construir nuevos tipos a partir de otros, por ejemplo, tenemos dos tipos a y b, podemos construir el nuevo tipo de función a $\rightarrow$ b. De esta manera, si consideramos ***Type 0*** como el universo de los tipos "pequeños", ***Type 1*** como un universo más grande que contiene a ***Type 0*** como un elemento, y ***Type 2*** el universo más grande que contenga a ***Type 1*** como un elemento. Reiterando el proceso, obtenemos una lista infinita de universos de tipos en la que para cada $n \in \mathbb{N}$, existe un ***Type n***, y se sigue la jerarquía de que ***Type n+1*** es un universo más grande que contiene a ***Type n*** como un elemento.

2.1.2. λ-cálculo

En cuanto a la abstracción de funciones y su evaluación, LEAN se apoya en λ-cálculo. Si tenemos una variable ***x :*** α, y construimos una expresión ***t :*** β, tendríamos la función fun (x : α) $\Rightarrow$ t, o de forma equivalente, $\lambda\ (x : \alpha) \Rightarrow t$, que también es un objeto de tipo $\alpha \rightarrow \beta$.

```
def f ( n : Nat ) : Nat := 2*n
def g ( m : Nat ) : Bool := m % 2 = 0

#check fun x : Nat => g (f x)     -- Nat → Bool
#check fun x => g (f x)           -- Nat → Bool
```

Ejemplo 2.3: Abstracción de funciones en Lean 4.

Observemos que para definir la composición de f y g, no es necesario precisar el tipo de x, puesto que LEAN lo infiere automáticamente a partir de las definiciones de f y g. Si intentamos hacer la composición de f (g x), se producirá un error, y LEAN nos proporciona información sobre el fallo como indica la siguiente figura.

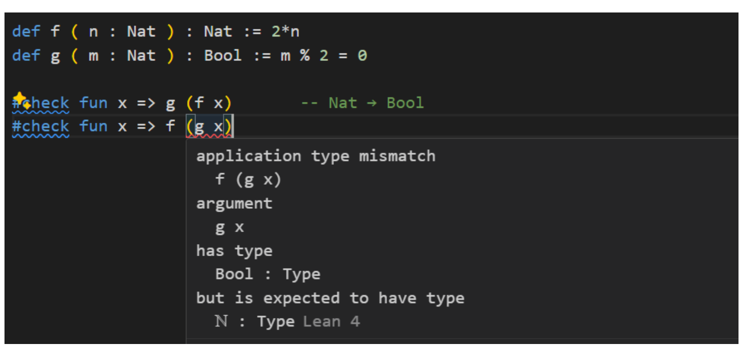

Figura 2.1: Error por tipos inadecuados en Lean 4.

2.1.3. La dependencia de la teoría de tipos dependientes

Los tipos pueden depender de los argumentos, por ejemplo $\boldsymbol{List}\ \alpha$ depende del parámetro α, por ello $\boldsymbol{List}\ \boldsymbol{Nat}$ se distingue de $\boldsymbol{List}\ \boldsymbol{Bool}$.