Notes on Lecture Notes: Logic, Category Theory, Topos Theory, Martin-Löf Type Theory, Homotopy Theory

Eric Schmid

Contents

IV Short Notes on Homotopy Theory 71

Part I

Mathematical Logic

0.1 Short Notes on Mathematical Logic

Mathematical logic is a subfield of mathematics exploring the applications of formal logic to mathematics. It bears close connections to metamathematics, the foundations of mathematics, and theoretical computer science.

0.1.1 First-Order Logic

First-order logic (FOL) is a powerful framework used to describe and reason about mathematical structures. It extends propositional logic by introducing quantifiers and the ability to talk about elements and their relationships within structures.

Structures and Signatures

A **structure** for a first-order language consists of a domain of discourse and an interpretation for each symbol in the language. The structure includes constants, functions, and relations.

Formal Definition A structure $\mathcal{A}$ for a signature σ is a tuple $(\mathcal{A}, \{c^{\mathcal{A}}\}_{c \in \mathrm{Const}(\sigma)}, \{f^{\mathcal{A}}\}_{f \in \mathrm{Func}(\sigma)}, \{R^{\mathcal{A}}\}_{R \in \mathrm{Rel}(\sigma)})$ where:

- $\mathcal{A}$ is a non-empty set (the domain).
- For each constant symbol c, $c^{\mathcal{A}} \in \mathcal{A}$.
- For each n-ary function symbol f, $f^{\mathcal{A}} : \mathcal{A}^n \to \mathcal{A}$.
- For each n-ary relation symbol R, $R^{\mathcal{A}} \subseteq \mathcal{A}^n$.

Example 1: Groups A group G is a structure $(G, \cdot, 1, ()^{-1})$ where:

- $(G, \cdot)$ forms a monoid: associativity and identity.
- Every element $g \in G$ has an inverse g^{-1}.

$$\forall a, b, c \in G, (a \cdot b) \cdot c = a \cdot (b \cdot c)$$
$$\forall a \in G, a \cdot 1 = 1 \cdot a = a$$
$$\forall a \in G, \exists b \in G, a \cdot b = b \cdot a = 1$$

Example 2: Ordered Fields An ordered field F is a structure $(F, 0, 1, +, \cdot, <)$ where $(F, 0, 1, +, \cdot)$ is a field and $<$ is a total order compatible with the field operations.

More Examples of Structures

Example 3: Vector Spaces A vector space V over a field F is a structure $(V, F, +, \cdot, 0, 1)$ where:

- $(V, +)$ is an abelian group.
- $\cdot$ is a scalar multiplication operation from $F \times V \to V$.
- 0 is the additive identity in V.
- 1 is the multiplicative identity in F.

Example 4: Graphs A graph G is a structure (V, E) where:

- V is a set of vertices.
- E is a set of edges, a binary relation on V (i.e., $E \subseteq V \times V$).

0.1.2 Syntax and Semantics

Syntax

The syntax of FOL involves terms and formulas:

- **Terms** represent elements of the domain and are constructed from variables, constants, and function symbols.
- **Formulas** express logical statements about terms and include atomic formulas, logical connectives $(\neg, \wedge, \vee, \rightarrow)$, and quantifiers $(\forall, \exists)$.

Example: Terms and Formulas Given a language with constants $0, 1$, a binary function $+$, and a binary relation $<$:

- Terms: $x, y, 0, 1, x + y$
- Atomic formulas: $x < y, x = y$
- Complex formulas: $x < y \wedge y < z$, $\forall x(x + 0 = x)$

Advanced Example: Arithmetic in $\mathbb{N}$ In the structure $\mathbb{N} = (\mathbb{N}, 0, 1, +, \cdot, <)$:

- Terms: $2 + 3, 5 \cdot 4, x^2$
- Atomic formulas: $2 + 2 = 4, 3 \cdot 3 = 9$
- Complex formulas: $\forall x(x^2 \geq 0), \exists y(y^2 = 2)$

Semantics

The semantics of FOL define how terms and formulas are interpreted in a structure:

- A term is interpreted as an element of the domain.
- An atomic formula $t_1 = t_2$ or $R(t_1, \ldots, t_n)$ is interpreted as true or false depending on the structure.
- Complex formulas are interpreted using the standard meanings of the logical connectives and quantifiers.

Example: Interpretation in Arithmetic In the structure $\mathbb{N} = (\mathbb{N}, 0, 1, +, \cdot, <)$:

- The term $x + 1$ represents the successor of x.
- The formula $\forall x(x + 0 = x)$ is true.
- The formula $\exists x(x \cdot x = 2)$ is false.

Advanced Example: Interpretation in Vector Spaces In the structure $\mathbb{R}^3 = (\mathbb{R}^3, \mathbb{R}, +, \cdot, 0, 1)$:

- The term $\mathbf{v} + \mathbf{w}$ represents the vector sum.
- The formula $\forall \mathbf{v}(\mathbf{v} + \mathbf{0} = \mathbf{v})$ is true.
- The formula $\exists \mathbf{v}(\mathbf{v} \cdot \mathbf{v} = -1)$ is false.

0.1.3 Definability

A set is definable in a structure if there exists a formula in the language that exactly characterizes the set.

Example: Definable Sets in $\mathbb{N}$

- The set of even numbers: $\{x \mid \exists y(x = 2 \cdot y)\}$
- The set of prime numbers: $\{x \mid \forall y \forall z(x = y \cdot z \to (y = 1 \lor z = 1))\}$

Advanced Example: Definable Sets in $\mathbb{R}$

- The set of positive numbers: $\{x \mid x > 0\}$
- The set of rational numbers: $\{x \mid \exists p \exists q(x = p/q \land q \neq 0)\}$

0.2 Theories and Models

A **theory** is a set of sentences in a particular language. A **model** of a theory is a structure in which all the sentences of the theory are true.

0.2.1 Axiomatization

The process of axiomatization involves identifying a set of axioms from which all other statements in the theory can be derived.

Example: Peano Arithmetic The axioms of Peano Arithmetic (PA) include:

- $\forall x(\neg S(x) = 0)$
- $\forall x \forall y(S(x) = S(y) \to x = y)$
- $\forall x(x + 0 = x)$
- $\forall x \forall y(x + S(y) = S(x + y))$
- Induction schema: $\varphi(0) \land \forall x(\varphi(x) \to \varphi(S(x))) \to \forall x \varphi(x)$

Detailed Example: Proving Properties in PA

To prove properties in Peano Arithmetic, we use induction. For instance, proving $\forall x(x + 0 = x)$:

1. Base case: $0 + 0 = 0$
2. Inductive step: Assume $x + 0 = x$ for some x. We need to show $S(x) + 0 = S(x)$.
3. By the definition of addition: $S(x) + 0 = S(x + 0) = S(x)$

Advanced Example: Induction on the Sum of Natural Numbers To prove $\forall x \forall y(x + y = y + x)$:

1. Base case: $y = 0 \implies x + 0 = x = 0 + x$
2. Inductive step: Assume $x + y = y + x$. We need to show $x + S(y) = S(y) + x$.
3. By the definition of addition: $x+S(y) = S(x+y) = S(y+x) = S(y) + x$

0.2.2 Completeness and Compactness

Gödel's Completeness Theorem

Gödel's Completeness Theorem states that if a formula is true in every model of a theory, it is provable from the theory.

Proof Sketch: Completeness Theorem

- Define a consistent set of sentences Σ such that every formula or its negation is in Σ.
- Extend Σ to a maximally consistent set Σ^*.
- Construct a model from Σ^* where every sentence in Σ^* is true.
- Show that if a formula is true in all models, it must be in Σ^*, hence provable.

Compactness Theorem

The Compactness Theorem states that if every finite subset of a theory has a model, then the whole theory has a model.

Example: Compactness in Graph Theory If every finite subgraph of an infinite graph is n-colorable, then the whole graph is n-colorable.

Proof Sketch: Compactness Theorem

- Assume T is a theory such that every finite subset of T has a model.
- Construct a model for T by considering the union of all finite models.
- Use the completeness theorem to show that the union is a model for T.

0.2.3 Model Theory

Model theory studies the relationships between formal languages and their interpretations or models. Key concepts include types, saturation, and elementary equivalence.

Types and Saturation

A type is a set of formulas with a single free variable that can be consistent with a theory.

Example: Types in Real Closed Fields Consider the type $p(x)$ in the theory of real closed fields T:

$$p(x) = \{x > 0, x < 1, x^2 < \frac{1}{2}, \ldots\}$$

A model M is κ-saturated if every type with fewer than κ parameters that is consistent with T is realized in M.

Saturation A model M is κ-saturated if for every set of formulas Σ with fewer than κ parameters, if Σ is consistent with M, then there is an element in M that satisfies all formulas in Σ.

Elementary Equivalence and Extensions

Two structures M and N are elementarily equivalent, denoted $M \equiv N$, if they satisfy the same first-order sentences. An elementary extension of M is a structure N such that $M \subseteq N$ and $M \equiv N$.

Example: Nonstandard Models of Arithmetic Nonstandard models of arithmetic contain "infinite" elements not present in the standard model, yet they satisfy all the axioms of Peano Arithmetic.

Advanced Example: Saturation in Model Theory Consider the type $p(x) = \{x \in \mathbb{N} \mid x \text{ is even}\}$. A model M of Peano Arithmetic is ω-saturated if every type that can be finitely realized in M is realized in M.

0.2.4 Recursion Theory

Recursion theory (computability theory) explores the capabilities and limits of algorithmic processes. It defines computable functions and examines problems that are unsolvable by any algorithm.

Example: Halting Problem The Halting Problem is undecidable: there is no algorithm that can determine whether an arbitrary program halts on a given input.

Primitive Recursive Functions

Primitive recursive functions are constructed from basic functions using composition and primitive recursion.

Example: Factorial Function The factorial function is primitive recursive:

$$f(0) = 1$$
$$f(n+1) = (n+1) \cdot f(n)$$

Advanced Example: Sum of Natural Numbers The function to sum the first n natural numbers is primitive recursive:

$$\text{sum}(0) = 0$$
$$\text{sum}(n+1) = (n+1) + \text{sum}(n)$$

General Recursive Functions

General recursive functions extend primitive recursive functions by allowing minimization (search for the least value).

Example: Ackermann Function The Ackermann function is an example of a general recursive function that is not primitive recursive:

$$A(0,n) = n+1$$
$$A(m+1,0) = A(m,1)$$
$$A(m+1,n+1) = A(m, A(m+1,n))$$

0.2.5 Proof Theory

Proof theory analyzes the structure of mathematical proofs. It involves formal systems, derivations, and consistency proofs.

Sequent Calculus

Sequent calculus is a formal system for proving logical statements, using sequents to represent implications.

Example: Sequent Calculus Rules The inference rules in sequent calculus include:

- **Axiom**: $\frac{}{A \vdash A}$
- **Cut Rule**: $\frac{\Gamma \vdash \Delta, A \quad \Gamma, A \vdash \Delta}{\Gamma \vdash \Delta}$
- **Conjunction**: $\frac{\Gamma \vdash \Delta, A \quad \Gamma \vdash \Delta, B}{\Gamma \vdash \Delta, A \wedge B}$

Advanced Example: Proof of a Theorem in Sequent Calculus To prove $\vdash A \vee \neg A$ (law of excluded middle):

1. Start with the sequent $\Gamma \vdash A, \neg A$ (axiom rule).
2. Apply the cut rule to obtain $\Gamma, \neg A \vdash A$ and $\Gamma, A \vdash \neg A$.
3. Conclude $\Gamma \vdash A \vee \neg A$.

0.2.6 Set Theory

Set theory provides a foundation for much of mathematics, dealing with the properties of sets, relations, and functions. Key results include Zermelo-Fraenkel set theory (ZF) and the Axiom of Choice (AC).

Example: Zermelo-Fraenkel Axioms ZF axioms include:

- **Extensionality**: $\forall x \forall y (\forall z (z \in x \leftrightarrow z \in y) \rightarrow x = y)$
- **Foundation**: $\forall x (\exists y (y \in x) \rightarrow \exists y (y \in x \wedge \forall z (z \in x \rightarrow \neg (z \in y))))$
- **Replacement**: $\forall A \forall B (\forall x (x \in A \rightarrow \exists ! y (B(x, y))) \rightarrow \exists C \forall y (y \in C \leftrightarrow \exists x (x \in A \wedge B(x, y))))$

Axiom of Choice

The Axiom of Choice states that for any set of nonempty sets, there exists a choice function selecting an element from each set.

Example: Applications of Axiom of Choice

- **Tychonoff's Theorem**: The product of any collection of compact topological spaces is compact.
- **Zorn's Lemma**: Every partially ordered set in which every chain has an upper bound contains at least one maximal element.

Advanced Example: Well-Ordering Theorem The Well-Ordering Theorem, equivalent to the Axiom of Choice, states that every set can be well-ordered.

0.2.7 Categorical Logic

Categorical logic connects logic and category theory, studying logical systems within the framework of category theory.

Topos Theory

A topos is a category that behaves like the category of sets and supports a form of logic known as intuitionistic logic.

Example: Properties of Topoi

- **Subobject Classifier**: A topos has a subobject classifier, which generalizes the concept of characteristic functions.
- **Exponentiation**: A topos allows for the construction of function spaces within the category.

Advanced Example: Heyting Algebras Heyting algebras, which interpret intuitionistic logic, are examples of structures in a topos.

0.3 Applications and Implications

The concepts of first-order logic and mathematical logic have profound implications across mathematics, computer science, and philosophy. They provide the foundations for formal verification, database theory, and the philosophy of mathematics.

0.3.1 Formal Verification

Formal verification uses logic to prove the correctness of software and hardware systems. It ensures that systems behave as intended under all possible conditions.

Example: Model Checking Model checking is a technique used in formal verification to automatically check whether a model of a system satisfies a given specification.

Advanced Example: Temporal Logic in Model Checking Temporal logic extends classical logic with operators that describe the temporal ordering of events, useful in verifying concurrent systems.

0.3.2 Database Theory

First-order logic underpins database query languages like SQL, allowing for the formal specification and manipulation of data.

Example: Relational Algebra Relational algebra is a formal system for manipulating relations in a database. It includes operations like selection, projection, union, and join.

Advanced Example: Query Optimization Query optimization involves transforming a database query into an equivalent one that can be executed more efficiently, often using logical equivalences.

0.3.3 Philosophy of Mathematics

Logical frameworks address foundational questions about the nature of mathematical truth, provability, and the limits of formal systems.

Example: Formalism vs. Platonism

- **Formalism**: Mathematics is a creation of formal systems and symbols.
- **Platonism**: Mathematical entities exist independently of human thought and language.

Advanced Example: Intuitionism Intuitionism, a philosophy of mathematics that denies the law of excluded middle, has significant implications for the interpretation of mathematical logic.

Part II

A Very Short Introduction to Topos Theory (adapted from Prof. Pettigrew's notes)

Chapter 1

Motivating Category Theory

1.1 The Idea Behind Category Theory

Category theory encourages a shift from focusing on the intrinsic properties of mathematical objects to emphasizing their roles and interactions. This approach can be summarized by the phrase: "Ask not what a thing is; ask what it does" [1]. This shift in perspective helps unify various areas of mathematics by providing a common framework to describe different mathematical constructs through their relationships.

1.2 Traditional vs. Categorical Perspectives

Traditional mathematics often describes objects by their intrinsic properties:

- **Groups:** Defined by a set with an operation following group axioms.
- **Sets:** Collections of distinct elements.
- **Topological Spaces:** Sets endowed with a topology specifying open sets.

In contrast, category theory interprets these objects through their morphisms:

- **Groups:** Understood through group homomorphisms.
- **Sets:** Understood through functions.
- **Topological Spaces:** Understood through continuous maps.

Category theory provides a more flexible and general approach by focusing on the relationships between objects rather than their internal structure.

Chapter 2

The Definition of a Category

A category $\mathcal{C}$ consists of:

- A collection of objects $\mathrm{Ob}(\mathcal{C})$.
- A collection of morphisms (arrows) $\mathrm{Ar}(\mathcal{C})$.
- Two functions: domain and codomain, assigning to each morphism $f : A \to B$ its source A and target B.
- A composition function $\circ$ that assigns to each pair of composable morphisms $f : A \to B$ and $g : B \to C$ an arrow $g \circ f : A \to C$.

$$A \xrightarrow{f} B \xrightarrow{g} C$$

These components must satisfy the following axioms:

1. **Associativity:** For all composable arrows f, g, h, the equation $(h \circ g) \circ f = h \circ (g \circ f)$ holds.
2. **Identity:** For each object A, there exists an identity morphism $\mathrm{id}_A : A \to A$ such that for any morphisms $f : A \to B$ and $g : B \to A$, we have $\mathrm{id}_B \circ f = f$ and $g \circ \mathrm{id}_A = g$.

2.1 Examples of Categories

- **Set:** Objects are sets, and morphisms are functions between sets.
- **Grp:** Objects are groups, and morphisms are group homomorphisms.
- **Top:** Objects are topological spaces, and morphisms are continuous maps.
- **Vect:** Objects are vector spaces, and morphisms are linear transformations.

Chapter 3

Slice Categories [3]

3.1 Sets/I and Sets($\to$) taken from Jacobs

Consider a family of sets as a function $\varphi : X \to I$. We often describe a family of sets as a function $\varphi : X \to I$ and say that X is a family over I and that φ displays the family (X_i). In order to emphasize that we think of such a map φ as a family, we often write it vertically as $\begin{pmatrix} X \\ \downarrow \\ I \end{pmatrix}$. A constant family is one of the form $\begin{pmatrix} I\times X \\ \downarrow \\ I \end{pmatrix}$, where π is the Cartesian product projection; often it is written simply as $I \times X$. Notice that all fibers of this constant family are (isomorphic to) X.

Such families $\begin{pmatrix} X \\ \downarrow \\ I \end{pmatrix}$ of sets give rise to two categories: the slice category **Sets**$/I$ and the arrow category **Sets**$(\to)$. The objects of **Sets**$/I$ are the I-indexed families, for a fixed set I; the objects of **Sets**$(\to)$ are all the I-indexed families, for all possible I.

3.1.1 Sets/I

Objects: families $\begin{pmatrix} X \\ \downarrow \\ I \end{pmatrix}$.

Morphisms: $\begin{pmatrix} X \\ \downarrow \\ I \end{pmatrix} \xrightarrow{f} \begin{pmatrix} Y \\ \downarrow \\ I \end{pmatrix}$ are functions $f : X \to Y$ making the following diagram commute.

$$\begin{tikzcd} X \arrow[rr, "f"] \arrow[dr, "\varphi"'] & & Y \arrow[dl, "\psi"] \\ & I & \end{tikzcd}$$

3.1.2 $\mathbf{Sets}(\to)$

Objects: families $\left(\begin{smallmatrix} X \\ \downarrow \\ I \end{smallmatrix}\right)$, for arbitrary sets I.

Morphisms: $\left(\begin{smallmatrix} X \\ \downarrow \\ I \end{smallmatrix}\right) \xrightarrow{(u,f)} \left(\begin{smallmatrix} Y \\ \downarrow \\ J \end{smallmatrix}\right)$ are pairs of functions $u : I \to J$ and $f : X \to Y$ for which the following diagram commutes.

$$\begin{tikzcd} X \arrow[r, "f"] \arrow[d, "\varphi"] & Y \arrow[d, "\psi"] \\ I \arrow[r, "u"] & J \end{tikzcd}$$

Hence, objects in the arrow category $\mathbf{Sets}(\to)$ involve an extra function u between the index sets. Notice that one can now view f as a collection of functions $f_i : X_i \to Y_{u(i)}$, since for $x \in \varphi^{-1}(i)$, $f(x)$ lands in $\psi^{-1}(u(i))$. Composition and identities in $\mathbf{Sets}(\to)$ are component-wise inherited from $\mathbf{Sets}$.

We further remark that there is a codomain functor $\mathrm{cod} : \mathbf{Sets}(\to) \to \mathbf{Sets}$; it maps $\left(\begin{smallmatrix} X \\ \downarrow \\ I \end{smallmatrix}\right) \mapsto I$ and $(\,(u,f)\,) \mapsto u$.

Also, for each I, there is a (non-full) inclusion functor $\mathbf{Sets}/I \to \mathbf{Sets}(\to)$.

Chapter 4

Monics, Epics, and Isomorphisms

An arrow $f : A \to B$ in a category $\mathcal{C}$ is:

- *Monic* (monomorphism) if for all arrows $g, h : C \to A$, $f \circ g = f \circ h$ implies $g = h$.
- *Epic* (epimorphism) if for all arrows $g, h : B \to C$, $g \circ f = h \circ f$ implies $g = h$.
- *Isomorphic* (isomorphism) if there exists an arrow $g : B \to A$ such that $g \circ f = \mathrm{id}_A$ and $f \circ g = \mathrm{id}_B$.

4.1 Examples in Set

- **Monic:** Injective functions.
- **Epic:** Surjective functions.
- **Isomorphisms:** Bijective functions.

4.2 Examples in Grp

- **Monic:** Injective group homomorphisms.
- **Epic:** Surjective group homomorphisms.
- **Isomorphisms:** Bijective group homomorphisms (isomorphisms of groups).

Chapter 5

Diagrams, Cones, Cocones, Limits, Colimits

In category theory, diagrams provide a structured way to visualize and understand the relationships between objects and morphisms. Cones and cocones are structures that relate diagrams to limits and colimits, respectively.

5.1 Diagrams

A *diagram* in a category $\mathcal{C}$ is a functor $D : J \to \mathcal{C}$ where J is an indexing category. Objects of J are mapped to objects of $\mathcal{C}$, and morphisms of J are mapped to morphisms in $\mathcal{C}$.

5.1.1 Examples of Diagrams

Diagrams can range from simple single-object diagrams to complex networks of objects and morphisms, such as sequences, commutative squares, or more intricate structures. They serve as a framework for discussing the relationships between objects and morphisms.

$$\begin{array}{ccc} A & \xrightarrow{f} & B \\ {\scriptstyle g}\downarrow & & \downarrow{\scriptstyle h} \\ C & \xrightarrow{k} & D \end{array}$$

5.2 Cones

A *cone* over a diagram $D : J \to \mathcal{C}$ consists of an object L and a family of morphisms $\lambda_X : L \to D(X)$ for each object X in J such that for every morphism $f : X \to Y$ in J, the following diagram commutes:

$$\begin{tikzcd} & L \arrow[dl, "\lambda_X"'] \arrow[dr, "\lambda_Y"] & \\ D(X) \arrow[rr, "D(f)"'] & & D(Y) \end{tikzcd}$$

5.3 Limits

"Let $F : J \to \mathcal{C}$ be a diagram of shape J in a category $\mathcal{C}$. A *cone* to F is an object N of $\mathcal{C}$ together with a family $\psi_X : N \to F(X)$ of morphisms indexed by the objects X of J, such that for every morphism $f : X \to Y$ in J, we have $F(f) \circ \psi_X = \psi_Y$.

A *limit* of the diagram $F : J \to \mathcal{C}$ is a cone (L, ϕ) to F such that for every cone (N, ψ) to F, there exists a unique morphism $u : N \to L$ such that $\phi_X \circ u = \psi_X$ for all X in J." [4]

$$\begin{tikzcd} & N \arrow[dl, "\psi_X"'] \arrow[d, dashed, "u"] \arrow[dr, "\psi_Y"] & \\ F(X) \arrow[rr, bend right, "F(f)"'] & L \arrow[l, "\phi_X"'] \arrow[r, "\phi_Y"] & F(Y) \end{tikzcd}$$

"One says that the cone (N, ψ) factors through the cone (L, ϕ) with the unique factorization u. The morphism u is sometimes called the *mediating morphism*." [4]

5.3.1 Examples of Limits

- **Products:** The product of two objects A and B in a category $\mathcal{C}$ is a limit of the diagram consisting of A and B with no morphisms between them.

- **Equalizers:** An equalizer of two parallel morphisms $f, g : A \to B$ is a limit of the diagram formed by A and B with two parallel arrows.

5.4 Cocones

A *cocone* under a diagram $D : J \to \mathcal{C}$ consists of an object L and a family of morphisms $\lambda_X : D(X) \to L$ for each object X in J such that for every morphism $f : X \to Y$ in J, the following diagram commutes:

$$
\begin{tikzcd}
D(X) \arrow[rr, "D(f)"] \arrow[dr, "\lambda_X"'] & & D(Y) \arrow[dl, "\lambda_Y"] \\
& L &
\end{tikzcd}
$$

5.5 Colimits

A *colimit* of a diagram $D : J \to \mathcal{C}$ is a cocone (L, λ) under D that is universal among all such cocones. This means that for any other cocone (L', λ') under D, there exists a unique morphism $u : L \to L'$ such that $\lambda'_X \circ u = \lambda_X$ for all X in J.

$$
\begin{tikzcd}
D(X) \arrow[rr, "D(f)"] \arrow[dr, "\lambda'_X"'] \arrow[ddr, "\lambda_X"'] & & D(Y) \arrow[dl, "\lambda'_Y"] \arrow[ddl, "\lambda_Y"] \\
& L \arrow[d, dashed, "u"] & \\
& L' &
\end{tikzcd}
$$

5.5.1 Examples of Colimits

- **Coproducts:** The coproduct of two objects A and B in a category $\mathcal{C}$ is a colimit of the diagram consisting of A and B with no morphisms between them.

- **Coequalizers:** A coequalizer of two parallel morphisms $f, g : A \to B$ is a colimit of the diagram formed by A and B with two parallel arrows.

5.6 Equalizers and Coequalizers

5.6.1 Equalizers

An *equalizer* of two parallel arrows $f, g : A \to B$ is an object E together with a morphism $e : E \to A$ such that $f \circ e = g \circ e$ and for any object Z with a morphism $z : Z \to A$ such that $f \circ z = g \circ z$, there exists a unique morphism $u : Z \to E$ such that $e \circ u = z$. This is depicted as follows:

$$\begin{tikzcd}
E \arrow[r, "e"] & A \arrow[r, shift left, "f"] \arrow[r, shift right, "g"'] & B \\
Z \arrow[u, dashed, "u"] \arrow[ur, "z"'] & &
\end{tikzcd}$$

5.6.2 Coequalizers

A *coequalizer* of two parallel arrows $f, g : A \to B$ in a category $\mathcal{C}$ is an object Q together with a morphism $q : B \to Q$, universal with the property $q \circ f = q \circ g$, as in the following diagram: [2]

$$\begin{tikzcd}
A \arrow[r, shift left, "f"] \arrow[r, shift right, "g"'] & B \arrow[r, "q"] \arrow[dr, "z"'] & Q \arrow[d, dotted, "u"] \\
& & Z
\end{tikzcd}$$

Given any Z and $z : B \to Z$, if $z \circ f = z \circ g$, then there exists a unique $u : Q \to Z$ such that $u \circ q = z$ [2].

5.7 Products and coproducts

5.7.1 Products

A *product* of two objects A and B in a category $\mathcal{C}$ is an object P together with two morphisms $\pi_1 : P \to A$ and $\pi_2 : P \to B$ such that for any object X with morphisms $f : X \to A$ and $g : X \to B$, there exists a unique morphism $u : X \to P$ such that $\pi_1 \circ u = f$ and $\pi_2 \circ u = g$. This can be depicted as:

$$\begin{tikzcd}
& X \arrow[dl, "f"'] \arrow[d, dashed, "u"] \arrow[dr, "g"] & \\
A & P \arrow[l, "\pi_1"] \arrow[r, "\pi_2"'] & B
\end{tikzcd}$$

5.7.2 Coproducts

A *coproduct* of two objects A and B in a category $\mathcal{C}$ is an object C together with two morphisms $\iota_1 : A \to C$ and $\iota_2 : B \to C$ such that for any object X with morphisms $f : A \to X$ and $g : B \to X$, there exists a unique morphism $u : C \to X$ such that $u \circ \iota_1 = f$ and $u \circ \iota_2 = g$. This can be depicted as:

$$\begin{tikzcd}
A \arrow[r, "\iota_1"] \arrow[dr, "f"'] & C \arrow[d, dashed, "u"] & B \arrow[l, "\iota_2"'] \arrow[dl, "g"] \\
& X &
\end{tikzcd}$$

5.8 Pushouts and Pullbacks

5.8.1 Pullbacks

A *pullback* (also known as a fiber product) of two morphisms $f : X \to Z$ and $g : Y \to Z$ in a category $\mathcal{C}$ is an object P together with two morphisms $p_1 : P \to X$ and $p_2 : P \to Y$ such that the following diagram commutes:

$$\begin{tikzcd}
P \arrow[r, "p_2"] \arrow[d, "p_1"'] & Y \arrow[d, "g"] \\
X \arrow[r, "f"'] & Z
\end{tikzcd}$$

Moreover, P must be universal with respect to this property, meaning that for any other object Q with morphisms $q_1 : Q \to X$ and $q_2 : Q \to Y$ making the diagram commute, there exists a unique morphism $u : Q \to P$ such that $p_1 \circ u = q_1$ and $p_2 \circ u = q_2$. This situation is illustrated in the following commutative diagram:

$$\begin{tikzcd}
Q \arrow[dr, dashed, "u"] \arrow[drr, bend left, "q_2"] \arrow[ddr, bend right, "q_1"'] & & \\
& P \arrow[r, "p_2"] \arrow[d, "p_1"'] & Y \arrow[d, "g"] \\
& X \arrow[r, "f"'] & Z
\end{tikzcd}$$

Explicitly, a pullback of the morphisms f and g consists of an object P and two morphisms $p_1 : P \to X$ and $p_2 : P \to Y$ for which the diagram

$$\begin{tikzcd}
P \arrow[r, "p_2"] \arrow[d, "p_1"'] & Y \arrow[d, "g"] \\
X \arrow[r, "f"'] & Z
\end{tikzcd}$$

commutes. Moreover, the pullback (P, p_1, p_2) must be universal with respect to this diagram. That is, for any other such triple (Q, q_1, q_2) where $q_1 : Q \to X$ and $q_2 : Q \to Y$ are morphisms with $f \circ q_1 = g \circ q_2$, there must exist a unique $u : Q \to P$ such that

$$p_1 \circ u = q_1, \quad p_2 \circ u = q_2.$$

This situation is illustrated in the following commutative diagram:

$$\begin{tikzcd}
Q \arrow[drr, bend left, "q_2"] \arrow[ddr, bend right, "q_1"'] \arrow[dr, dashed, "u"] & & \\
& P \arrow[r, "p_2"] \arrow[d, "p_1"'] & Y \arrow[d, "g"] \\
& X \arrow[r, "f"'] & Z
\end{tikzcd}$$

5.8.2 Pushouts

A *pushout* (also known as a cofiber product) of two morphisms $f : Z \to X$ and $g : Z \to Y$ in a category $\mathcal{C}$ is an object P together with two morphisms $\iota_1 : X \to P$ and $\iota_2 : Y \to P$ such that the following diagram commutes:

$$\begin{tikzcd}
Z \arrow[r, "g"] \arrow[d, "f"'] & Y \arrow[d, "\iota_2"] \\
X \arrow[r, "\iota_1"'] & P
\end{tikzcd}$$

Moreover, P must be universal with respect to this property, meaning that for any other object Q with morphisms $q_1 : X \to Q$ and $q_2 : Y \to Q$ making the diagram commute, there exists a unique morphism $u : P \to Q$ such that $u \circ \iota_1 = q_1$ and $u \circ \iota_2 = q_2$.

$$\begin{tikzcd}
Z \arrow[r, "g"] \arrow[d, "f"'] & Y \arrow[d, "\iota_2"] \arrow[ddr, "q_2"] & \\
X \arrow[r, "\iota_1"'] \arrow[drr, "q_1"'] & P \arrow[dr, dashed, "u"] & \\
& & Q
\end{tikzcd}$$

Chapter 6

Initial and Terminal Objects [1]

Consider the empty diagram in the category $\mathcal{C}$. The cones and cocones over this diagram are simply the objects of $\mathcal{C}$. Therefore:

- If the empty diagram has a limit, it is an object 1 such that, for every object A in $\mathcal{C}$, there is a unique morphism $1_A : A \to 1$.

- If the empty diagram has a colimit, it is an object 0 such that, for every object A in $\mathcal{C}$, there is a unique morphism $0_A : 0 \to A$.

6.1 Definitions

- **Initial Object:** A limit of the empty diagram (if it exists) is called a terminal object of $\mathcal{C}$.

- **Terminal Object:** A colimit of the empty diagram (if it exists) is called an initial object of $\mathcal{C}$.

By Propositions 5.1.3 and 5.2.3, a terminal or initial object is unique up to isomorphism. By Axiom 2, every topos has initial and terminal objects.

6.1.1 Examples

- In **Set**, the empty set $\emptyset$ is the only initial object, and any singleton set $\{a\}$ is a terminal object.
- In **Grp**, the trivial group $\{e_G\}$ is both initial and terminal. Such objects are called zero objects.
- In a category based on a poset, any minimum element is an initial object (if it exists), and any maximum element is a terminal object (if it exists).
- If $\mathcal{C}$ is a category, then $Id_X : X \to X$ is a terminal object of $\mathcal{C}/X$. If $\mathcal{C}$ has an initial object 0, then $0_X : 0 \to X$ is an initial object of $\mathcal{C}/X$.

6.2 Properties

- If 1 is a terminal object and $f : 1 \to A$, then the arrow is monic.
- If 1 is a terminal object, then $1 \times A \cong A \cong A \times 1$.

Chapter 7

Members of Objects [1]

In traditional set theory, the fundamental notion is the membership relation. In category theory, the analogous concept involves arrows rather than elements. In a category $\mathcal{C}$ with a terminal object 1, the members of an object A are the morphisms from 1 to A.

7.1 Membership

In the context of category theory, the action of picking out a member of a set A can be understood as a function from a singleton set into A. This is because any singleton is a terminal object and any terminal object is a singleton in **Set**.

Member of: If $\mathcal{C}$ is a category with a terminal object 1 and A is an object of $\mathcal{C}$, then a member of A is an arrow $x : 1 \to A$.

This definition implies that there cannot be 'membership chains' as members are arrows, not objects, and arrows cannot have members.

- 1 has exactly one element.
- If 0 has an element, then $0 \cong 1$.

7.2 Injective and Surjective

We can also define injective and surjective arrows in this context.

Injective and Surjective: Given $f : A \to B$, we say that

- The arrow is **injective** if, for all $x, y : 1 \to A$, if $fx = fy$ then $x = y$.

- The arrow is **surjective** if, for all $y : 1 \to B$, there is $x : 1 \to A$ such that $fx = y$.

In **Set**, the injective arrows are the monics, and the surjective arrows are the epics. However, this is not necessarily true in all categories with terminal objects.

Chapter 8

Exponential Objects [1]

Axiom 2 ensures that all toposes have analogues of the addition and multiplication operations on sets—they are coproducts and products, respectively. However, it does not guarantee analogues of power sets $P(A)$ or function spaces $B^A = \{f : A \to B\}$. To address this, we introduce an axiom guaranteeing these objects.

Consider a (set-theoretical) function $f : A \times C \to B$. For every element $c \in C$, the function $f_c : a \mapsto f(a, c)$ is a function from A to B. The function $\hat{f} : c \mapsto f_c$ maps C into B^A. There is also a function $\text{ev} : A \times B^A \to B$ that evaluates a function from A to B at a value in A.

Therefore:

$$ev(a, f_c) = f_c(a) = f(a, c)$$

Exponential: Suppose $\mathcal{C}$ is a category with products. For any objects A and B, an exponential of A and B consists of

- An object B^A of category $\mathcal{C}$
- An arrow $ev : A \times B^A \to B$ of category $\mathcal{C}$

such that for any arrow $f : A \times C \to B$, there is an arrow $\hat{f} : C \to B^A$ making the following diagram commute:

$$\begin{tikzcd} A \times B^A \arrow[r, "\text{ev}"] & B \\ A \times C \arrow[u, "\text{Id}_A \times \hat{f}"] \arrow[ur, "f"'] & \end{tikzcd}$$

Exponentials of A and B are unique up to isomorphism, and anything isomorphic to an exponential of A and B is itself an exponential of A and B.

We can check the validity of the exponential B^A by considering whether it satisfies certain basic conditions stated in terms of the membership relation. In set theory, an exponential object should have the following property:

$$f \text{ is a member of } B^A \text{ iff } f : A \to B$$

Although this is not exactly the result we get in category theory, we do obtain a close analogue: there is a one-to-one correspondence between arrows $f : A \to B$ and arrows $g : 1 \to B^A$.

Name of an arrow: If $A \to B$ is an arrow, the name of f (written $\ulcorner f \urcorner$) is the arrow $1 \to B^A$ such that the following diagram commutes:

$$\begin{tikzcd} A \times B^A \arrow[r, "\mathrm{ev}"] & B \\ A \times 1 \arrow[u, "\mathrm{Id}_A \times \ulcorner f \urcorner"] \arrow[ur, "f"'] & \end{tikzcd}$$

There is a bijection $f \mapsto \ulcorner f \urcorner$ between the set of arrows $f : A \to B$ and the set of arrows $\ulcorner f \urcorner : 1 \to B^A$.

With the definition of an exponential in hand, we can introduce the third axiom of a topos.

A topos has exponentials for every pair of objects.

Cartesian Closed Category: A category with limits for all finite diagrams and exponentials for all pairs of objects is called a Cartesian closed category.

[1].

Chapter 9

Subobjects and Their Classifiers [1]

9.1 Subobjects

In category theory, the analogue of a subset of a set A is called a subobject of A. A subobject is not another object but an arrow, specifically a monic arrow from an object S into A.

Part or Subobject: Suppose A is an object. Then a subobject of A is a monic arrow $S \to A$.

Inclusion: Suppose $S \to A$ and $T \to A$ are subobjects of A. We say that S is included in T (written $S \subseteq T$) if there is a morphism $S \to T$ such that the following diagram commutes:

$$\begin{tikzcd} S \arrow[r] \arrow[dr] & T \arrow[d] \\ & A \end{tikzcd}$$

9.2 Subobject Classifiers

Next, we introduce an object Ω in a topos whose members act as truth values and associate with each subobject $S \to A$ a characteristic function $A \to \Omega$.

9.2.1 Motivation

Consider sets where $\Omega = \{\text{true}, \text{false}\}$. The characteristic function of a subset $S \subseteq A$ can be seen as the function χ_S mapping A to $\{\text{true}, \text{false}\}$ such that the inverse image of $\{\text{true}\}$ under χ_S is S.

9.2.2 Inverse Images of Subobjects

In **Set**, a pullback for the diagram

$$\begin{tikzcd} B \arrow[dr, "f"] & & S \arrow[dl, "i"'] \\ & A & \end{tikzcd}$$

is the set

$$\{(b, s) \mid f(b) = i(s)\}$$

If $S \to A$ and i is the inclusion map $i : s \mapsto s$ for all $s \in S$, then a pullback is

$$\{(b, s) \in B \times S \mid f(b) = i(s) = s\} \cong \{b \in B \mid f(b) \in S\} = f^{-1}(S)$$

Inverse Image: If $S \to A$ and $B \to A$, then the pullback of the diagram

$$\begin{tikzcd} B \arrow[dr, "f"] & & S \arrow[dl, "i"'] \\ & A & \end{tikzcd}$$

is called an inverse image of $S \to A$ under f (written $f^{-1}(S)$).

9.2.3 Definition

We define the characteristic function χ_i of a subobject $S \to A$ to be the unique function such that S is an inverse image of $1 \to \Omega$ under χ_i.

Subobject Classifier: Suppose $\mathcal{C}$ is a category with a terminal object 1. A subobject classifier in $\mathcal{C}$ consists of:

- An object Ω of category $\mathcal{C}$.
- An arrow $1 \to \Omega$

such that for any object A and subobject $S \to A$, there is a unique arrow $\chi_S : A \to \Omega$ such that:

$$\begin{tikzcd} S \arrow[r] \arrow[d] & 1 \arrow[d, "1"] \\ A \arrow[r, "\chi_i"] & \Omega \end{tikzcd}$$

is a pullback square.

Subobject classifiers are unique up to isomorphism, and anything isomorphic to a subobject classifier is itself a subobject classifier.

The final axiom of toposes can now be stated:

A topos has a subobject classifier.

False: The arrow $1 \to \Omega$ is the characteristic function of the subobject $0 \to 1$.

[1].

Chapter 10

The Definition of a Topos

10.1 The Definition

Having outlined the necessary axioms, we define a topos:

Topos: A topos is a category with:

- Limits and colimits for all finite diagrams.
- Exponentials for every pair of objects.
- A subobject classifier.

10.2 Examples

- **Set** is a topos.
- **Grp** is not a topos.
- **FinSet** is a topos.

10.3 Fundamental Theorem of Toposes

"If $\mathcal{E}$ is a topos and X is an object in $\mathcal{E}$, then $\mathcal{E}/X$ is a topos." [1].

Chapter 11

Algebra of Subobjects

In category theory, understanding subobjects and their algebra is crucial for grasping the structure and properties of categories. Here, we explain subobject lattices and their relationship to Boolean and Heyting algebras.

11.0.1 Subobject Lattices

A **subobject** of an object A in a category $\mathcal{C}$ is defined as an equivalence class of monomorphisms $m : S \hookrightarrow A$. Two monomorphisms $m : S \hookrightarrow A$ and $m' : S' \hookrightarrow A$ are considered equivalent if there exists an isomorphism $f : S \to S'$ such that $m = m' \circ f$.

The collection of all subobjects of A, denoted as $\mathrm{Sub}(A)$, forms a **partially ordered set** (poset) under inclusion. Specifically, for subobjects $[m : S \hookrightarrow A]$ and $[m' : S' \hookrightarrow A$, we have $[m] \leq [m']$ if there exists a morphism $f : S \to S'$ making the following diagram commute:

$$\begin{tikzcd}
S \arrow[rr, "f"] \arrow[dr, "m"'] & & S' \arrow[dl, "m'"] \\
& A &
\end{tikzcd}$$

This poset of subobjects, $\mathrm{Sub}(A)$, possesses additional structure, making it a **lattice**.

11.0.2 Lattice Structure of Subobjects

A **lattice** is a poset in which any two elements have a greatest lower bound (glb) or meet, and a least upper bound (lub) or join.

For subobjects S and T of A:

- The **meet** $S \wedge T$ is given by the intersection of S and T in A.
- The **join** $S \vee T$ is represented by the subobject generated by the union of S and T.

Formally, these operations can be defined through pullbacks and pushouts in the category $\mathcal{C}$.

11.0.3 Heyting Algebra of Subobjects

In a **Heyting category**, the poset $\mathrm{Sub}(A)$ is not just a lattice but a **Heyting algebra**. This means it supports an additional operation called **implication** $\to$, which is characterized by the property that for subobjects S and T of A, there is a largest subobject U such that $S \wedge U \leq T$.

Heyting algebras generalize Boolean algebras and are essential in the internal logic of a topos, particularly in intuitionistic logic.

11.0.4 Power Objects in Topos Theory

A **topos** can be seen as a categorical generalization of set theory, incorporating both logical and geometrical aspects. In a topos, every subobject poset $\mathrm{Sub}(A)$ is a Heyting algebra, and there exists a **power object** $\mathcal{P}(A)$, which internalizes the notion of the power set in set theory.

The power object $\mathcal{P}(A)$ of an object A is equipped with a monomorphism $\in_A : A \times \mathcal{P}(A) \to \Omega$, where Ω is the subobject classifier of the topos. This morphism satisfies a universal property analogous to the characteristic function of a subset in set theory.

11.0.5 Boolean Topos

In a **Boolean topos**, the internal Heyting algebra $\mathrm{Sub}(A)$ for any object A is a Boolean algebra. This implies that every subobject has a complement, and the internal logic of the topos aligns with classical Boolean logic.

11.0.6 Internal Logic and Lawvere-Tierney Topology

The internal logic of a topos can be intuitionistic or classical, depending on whether it forms a Heyting algebra or a Boolean algebra. The **Lawvere-Tierney topology** provides a framework to study these logical structures within a topos, defining modalities that extend the logical operations within the category.

References

- Pettigrew, R. *An Introduction to Toposes.* [Unpublished Notes].
- Awodey, S. (2010). *Category Theory.* Oxford Logic Guides 52, Oxford University Press.
- nLab: Heyting Algebra

Chapter 12

Kinds of Topos

12.1 Non-degenerate Toposes

A topos is non-degenerate if it has more than one object and more than one morphism.

12.2 Well-pointed Toposes

A topos is well-pointed if the only arrow $1 \to 1$ is the identity arrow.

12.3 Bivalent Toposes

A topos is bivalent if it has exactly two objects and two morphisms.

12.4 Boolean Toposes

A topos is Boolean if its subobject classifier is a Boolean algebra.

12.5 Natural Number Objects

A topos has a natural number object if there is an object N and arrows $0 : 1 \to N$ and $s : N \to N$ such that for any object A and arrows $f : 1 \to A$ and $g : A \to A$, there is a unique arrow $h : N \to A$ such that $h \circ 0 = f$ and $h \circ s = g \circ h$.

[1].

Chapter 13

Functors

13.1 The Definition of a Functor

A functor $F : \mathcal{C} \to \mathcal{D}$ consists of:

- A function $F : \mathrm{Ob}(\mathcal{C}) \to \mathrm{Ob}(\mathcal{D})$
- A function $F : \mathrm{Ar}(\mathcal{C}) \to \mathrm{Ar}(\mathcal{D})$

such that the following properties hold:

- $F(\mathrm{id}_A) = \mathrm{id}_{F(A)}$ for all objects A in $\mathcal{C}$.
- $F(g \circ f) = F(g) \circ F(f)$ for all arrows f, g in $\mathcal{C}$.

13.2 Examples of Functors

- The identity functor $\mathrm{Id} : \mathcal{C} \to \mathcal{C}$.
- The constant functor $C : \mathcal{C} \to \mathcal{D}$ that sends every object to a fixed object D and every arrow to id_D.
- The hom-functor $\mathrm{Hom}(A, -) : \mathcal{C} \to \mathbf{Set}$ that sends every object B to the set of arrows $A \to B$ and every arrow $f : B \to C$ to the function $\mathrm{Hom}(A, f) : \mathrm{Hom}(A, B) \to \mathrm{Hom}(A, C)$ given by composition with f.

13.3 Properties of Functors

Functors preserve the structure of categories in various ways:

- They preserve isomorphisms: If $f : A \to B$ is an isomorphism in $\mathcal{C}$, then $F(f)$ is an isomorphism in $\mathcal{D}$.
- They preserve commutative diagrams: If a diagram commutes in $\mathcal{C}$, its image under F commutes in $\mathcal{D}$.
- They preserve limits and colimits: If $\mathcal{C}$ has a limit (or colimit) of a diagram D, then $\mathcal{D}$ has a limit (or colimit) of $F(D)$.

Chapter 14

Natural Transformations

14.1 The Definition of a Natural Transformation

A natural transformation $\eta : F \Rightarrow G$ between two functors $F, G : \mathcal{C} \to \mathcal{D}$ consists of:

- For each object A in $\mathcal{C}$, an arrow $\eta_A : F(A) \to G(A)$ in $\mathcal{D}$

such that for every arrow $f : A \to B$ in $\mathcal{C}$, the following diagram commutes:

$$\begin{array}{ccc} F(A) & \xrightarrow{F(f)} & F(B) \\ {\scriptstyle \eta_A}\big\downarrow & & \big\downarrow{\scriptstyle \eta_B} \\ G(A) & \xrightarrow{G(f)} & G(B) \end{array}$$

This commutative diagram ensures that natural transformations respect the structure of the categories involved, providing a way to compare functors in a coherent manner.

14.2 Examples of Natural Transformations

- The identity natural transformation $\mathrm{Id} : F \Rightarrow F$ assigns the identity morphism to each object, ensuring that $\eta_A = \mathrm{id}_{F(A)}$

for all A in $\mathcal{C}$.

- The composition of two natural transformations $\eta : F \Rightarrow G$ and $\mu : G \Rightarrow H$ is a natural transformation $\mu \circ \eta : F \Rightarrow H$ defined by $(\mu \circ \eta)_A = \mu_A \circ \eta_A$ for each object A in $\mathcal{C}$.

- For the hom-functor $\mathrm{Hom}(A, -) : \mathcal{C} \to \mathbf{Set}$, a natural transformation between $\mathrm{Hom}(A, -)$ and another functor $\mathrm{Hom}(B, -)$ would be given by a function $\eta : \mathrm{Hom}(A, X) \to \mathrm{Hom}(B, X)$ for each X in $\mathcal{C}$ that respects composition and identity.

14.3 Properties of Natural Transformations

Natural transformations have several key properties:

- **Naturality:** The naturality condition, expressed by the commutative diagram above, ensures that the transformation is consistent with the action of the functors on morphisms.

- **Vertical and Horizontal Composition:** Natural transformations can be composed both vertically (as in the composition of η and μ) and horizontally, which involves transforming functors applied in sequence.

[1].

Chapter 15

Adjoint Functors

15.1 Definition

Two functors $F : \mathcal{C} \to \mathcal{D}$ and $G : \mathcal{D} \to \mathcal{C}$ are adjoint if there is a natural isomorphism

$$\mathrm{Hom}_{\mathcal{D}}(F(A), B) \cong \mathrm{Hom}_{\mathcal{C}}(A, G(B))$$

for all objects A in $\mathcal{C}$ and B in $\mathcal{D}$.

15.1.1 Definition via Counit-Unit Adjunction [4]

"A counit-unit adjunction between two categories $\mathcal{C}$ and $\mathcal{D}$ consists of two functors $F : \mathcal{D} \to \mathcal{C}$ and $G : \mathcal{C} \to \mathcal{D}$ and two natural transformations $\varepsilon : FG \to 1_{\mathcal{C}}$ and $\eta : 1_{\mathcal{D}} \to GF$, respectively called the counit and the unit of the adjunction, such that the compositions

$$F \xrightarrow{F\eta} FGF \xrightarrow{\varepsilon F} F \quad \text{and} \quad G \xrightarrow{\eta G} GFG \xrightarrow{G\varepsilon} G$$

are the identity transformations 1_F and 1_G on F and G respectively. In this situation, we say that F is left adjoint to G and G is right adjoint to F, and may indicate this relationship by writing $(\varepsilon, \eta) : F \dashv G$, or simply $F \dashv G$.

In equation form, the above conditions on (ε, η) are the counit-unit equations:

$$1_F = \varepsilon F \circ F\eta \quad \text{and} \quad 1_G = G\varepsilon \circ \eta G$$

which mean that for each X in $\mathcal{C}$ and each Y in $\mathcal{D}$,

$$1_{FY} = \varepsilon_{FY} \circ F(\eta_Y) \quad \text{and} \quad 1_{GX} = G(\varepsilon_X) \circ \eta_{GX}$$

Note that $1_{\mathcal{C}}$ denotes the identity functor on the category $\mathcal{C}$, 1_F denotes the identity natural transformation from the functor F to itself, and 1_{FY} denotes the identity morphism of the object FY."

15.1.2 Definition via Universal Morphisms [4]

"By definition, a functor $F : \mathcal{D} \to \mathcal{C}$ is a left adjoint functor if for each object X in $\mathcal{C}$ there exists a universal morphism from F to X. Spelled out, this means that for each object X in $\mathcal{C}$ there exists an object $G(X)$ in $\mathcal{D}$ and a morphism $\varepsilon_X : F(G(X)) \to X$ such that for every object Y in $\mathcal{D}$ and every morphism $f : F(Y) \to X$ there exists a unique morphism $g : Y \to G(X)$ with $\varepsilon_X \circ F(g) = f$.

The latter equation is expressed by the following commutative diagram:

$$\begin{tikzcd} F(Y) \arrow[rr, dashed, "F(g)"] \arrow[dr, "f"'] & & F(G(X)) \arrow[dl, "\varepsilon_X"] \\ & X & \end{tikzcd}$$

In this situation, one can show that G can be turned into a functor $G : \mathcal{C} \to \mathcal{D}$ in a unique way such that $\varepsilon_X \circ F(G(f)) = f \circ \varepsilon_{X'}$ for all morphisms $f : X' \to X$ in $\mathcal{C}$; F is then called a left adjoint to G.

Similarly, we may define right-adjoint functors. A functor $G : \mathcal{C} \to \mathcal{D}$ is a right adjoint functor if for each object Y in $\mathcal{D}$ there exists a universal morphism from Y to G. Spelled out, this means that for each object Y in $\mathcal{D}$ there exists an object $F(Y)$ in $\mathcal{C}$ and a morphism $\eta_Y : Y \to G(F(Y))$ such that for every object X in $\mathcal{C}$ and every morphism $g : Y \to G(X)$ there exists a unique morphism $f : F(Y) \to X$ with $G(f) \circ \eta_Y = g$

The latter equation is expressed by the following commutative diagram:"

$$\begin{tikzcd} Y \arrow[rr, "\eta_Y"] \arrow[dr, "g"'] & & G(F(Y)) \arrow[dl, dashed, "G(f)"] \\ & G(X) & \end{tikzcd}$$

15.2 Examples Concerning Set

- The forgetful functor $U : \mathbf{Grp} \to \mathbf{Set}$ has a left adjoint $F : \mathbf{Set} \to \mathbf{Grp}$ that sends each set S to the free group on S. This means there is a natural isomorphism $\mathrm{Hom}_{\mathbf{Grp}}(F(S), G) \cong \mathrm{Hom}_{\mathbf{Set}}(S, U(G))$ for any set S and group G.

- The forgetful functor $U : \mathbf{Top} \to \mathbf{Set}$ has a left adjoint $F : \mathbf{Set} \to \mathbf{Top}$ that sends each set S to the discrete topology on S. Here, $\mathrm{Hom}_{\mathbf{Top}}(F(S), T) \cong \mathrm{Hom}_{\mathbf{Set}}(S, U(T))$ for any set S and topological space T.

15.3 Examples Concerning Forgetful Functors

Forgetful functors between various categories often have adjoints. For example:

- The forgetful functor $U : \mathbf{Mon} \to \mathbf{Set}$ has a left adjoint $F : \mathbf{Set} \to \mathbf{Mon}$ that sends each set S to the free monoid on S.

- The forgetful functor $U : \mathbf{Vect} \to \mathbf{Set}$ has a left adjoint $F : \mathbf{Set} \to \mathbf{Vect}$ that sends each set S to the free vector space on S.

[1].

Bibliography

[1] Pettigrew, R., *An Introduction to Toposes*, Unpublished Notes.

[2] Awodey, S., *Category Theory*, Oxford Logic Guides 52, Oxford University Press, 2010.

[3] Jacobs, B., *Categorical Logic and Type Theory*, Elsevier, 1999.

[4] Wikipedia contributors, "Adjoint functors," Wikipedia, The Free Encyclopedia, `https://en.wikipedia.org/wiki/Adjoint_functors` (accessed June 12, 2024).

[5] nLab contributors, *Heyting algebra*, `https://ncatlab.org/nlab/show/Heyting+algebra#to_toposes`, nLab. (accessed June 12, 2024).

Part III

Short Notes on Dependent Type Theory

15.4 Basics of Dependent Type Theory

15.4.1 Types and Terms

In dependent type theory, every term belongs to a type, and these types can depend on terms. This dependency differentiates dependent types from simple types. For example, the type of vectors of a given length is a dependent type: $\mathrm{Vec}(n, A)$, where n is a natural number and A is the type of the vector elements.

15.4.2 Contexts and Judgements

A context Γ is a sequence of variable declarations. Judgements in dependent type theory include:

- **Type judgements:** $\Gamma \vdash T$ type
- **Term judgements:** $\Gamma \vdash t : T$
- **Equality judgements:** $\Gamma \vdash t = s : T$

15.4.3 Syntax and Typing Rules

Dependent type theory extends the simply typed lambda calculus with dependent types and terms. The syntax includes:

- **Types:** $T ::= \mathrm{Type} \mid T \to T \mid \Pi(x : T).T$
- **Terms:** $t ::= x \mid \lambda x : T.t \mid t\,t \mid (t, t) \mid \pi_1(t) \mid \pi_2(t)$

15.4.4 Typing Rules

The typing rules for dependent types are as follows:

- **Variable:** If $x : T \in \Gamma$, then $\Gamma \vdash x : T$.
- **Abstraction:** If $\Gamma, x : T_1 \vdash t : T_2$, then $\Gamma \vdash \lambda x : T_1.t : T_1 \to T_2$.
- **Application:** If $\Gamma \vdash t_1 : T_1 \to T_2$ and $\Gamma \vdash t_2 : T_1$, then $\Gamma \vdash t_1\, t_2 : T_2$.
- **Dependent Product:** If $\Gamma \vdash T : \mathrm{Type}$ and $\Gamma, x : T \vdash U : \mathrm{Type}$, then $\Gamma \vdash \Pi(x : T).U : \mathrm{Type}$.
- **Pair:** If $\Gamma \vdash t_1 : T_1$ and $\Gamma \vdash t_2 : T_2[t_1/x]$, then $\Gamma \vdash (t_1, t_2) : \Sigma(x : T_1).T_2$.

- **Projection:** If $\Gamma \vdash t : \Sigma(x : T_1).T_2$, then $\Gamma \vdash \pi_1(t) : T_1$ and $\Gamma \vdash \pi_2(t) : T_2[\pi_1(t)/x]$.

15.5 Dependent Sum Types (Σ-types)

15.5.1 Formation and Introduction

Dependent sum types, or Σ-types, generalize the concept of pairs. A type of the form $\Sigma(x : A).B(x)$ represents pairs (a, b) where a is of type A and b is of type $B(a)$. The formation rule is:

$$\frac{\Gamma \vdash A : \text{Type} \quad \Gamma, x : A \vdash B(x) : \text{Type}}{\Gamma \vdash \Sigma(x : A).B(x) : \text{Type}}$$

The introduction rule for pairs is:

$$\frac{\Gamma \vdash a : A \quad \Gamma \vdash b : B(a)}{\Gamma \vdash (a, b) : \Sigma(x : A).B(x)}$$

15.5.2 Elimination and Computation

To extract components from a pair, we use the projection functions:

$$\frac{\Gamma \vdash p : \Sigma(x : A).B(x)}{\Gamma \vdash \pi_1(p) : A}$$

$$\frac{\Gamma \vdash p : \Sigma(x : A).B(x)}{\Gamma \vdash \pi_2(p) : B(\pi_1(p))}$$

The computation rules for Σ-types are:

$$\pi_1(a, b) = a$$

$$\pi_2(a, b) = b$$

15.6 Dependent Product Types (Π-types)

15.6.1 Formation and Introduction

Dependent product types, or Π-types, generalize function types. A type of the form $\Pi(x : A).B(x)$ represents functions that take an input a of type A and return a value of type $B(a)$. The formation rule is:

$$\frac{\Gamma \vdash A : \text{Type} \quad \Gamma, x : A \vdash B(x) : \text{Type}}{\Gamma \vdash \Pi(x : A).B(x) : \text{Type}}$$

The introduction rule for functions is:

$$\frac{\Gamma, x : A \vdash b : B(x)}{\Gamma \vdash \lambda x : A.b : \Pi(x : A).B(x)}$$

15.6.2 Elimination and Computation

To apply a dependent function to an argument, we use the application rule:

$$\frac{\Gamma \vdash f : \Pi(x : A).B(x) \quad \Gamma \vdash a : A}{\Gamma \vdash f\, a : B(a)}$$

The computation rule for Π-types is:

$$(\lambda x : A.b)\, a = b[a/x]$$

15.7 Inductive Types and Pattern Matching

15.7.1 Inductive Types

Inductive types allow the definition of complex data structures and recursive functions. An example is the natural numbers:

$$\text{Inductive}\, \mathbb{N} := \text{zero} : \mathbb{N} \mid \text{succ} : \mathbb{N} \to \mathbb{N}$$

15.7.2 Pattern Matching

Pattern matching provides a way to define functions by cases. For example, the addition function on natural numbers can be defined as:

$$\text{fix}\, \text{add}(nm : \mathbb{N}) : \mathbb{N} := \text{match}\, n\, \text{with}\, \text{zero} \Rightarrow m \mid \text{succ}(n') \Rightarrow \text{succ}(\text{add}\, n'\, m)$$

15.8 Applications of Dependent Type Theory

15.8.1 Proof Assistants

Proof assistants based on dependent type theory provide expressive languages for specifying and reasoning about mathematical theories and programs. They ensure that:

- Theorems are applied with the correct hypotheses.
- Functions are applied to the correct arguments.
- No cases are missing in proofs or function definitions.
- No invalid logical steps are taken (all reasoning is reduced to elementary steps).

15.8.2 Programming Languages

Programming languages that incorporate dependent types, such as Agda and Idris, enable the development of programs with strong guarantees about their behavior. Dependent types can express complex properties and invariants, allowing the compiler to check these properties at compile-time.

15.9 Semantics and Formal Rules

15.9.1 Types

The basic notion in Martin-Löf's type theory is the notion of type. A type is explained by saying what an object of the type is and what it means for two objects of the type to be identical. This means that we can make the judgement:

$$A \text{ is a type}$$

which we formally write as:

$$A \text{ type}$$

when we know the conditions for asserting that something is an object of type A and when we know the conditions for asserting that two objects of type A are identical. We require that the conditions for identifying two objects must define an equivalence relation64†source63†source.

15.9.2 Hypothetical Judgements

Hypothetical judgements depend on a context of variable declarations:

$$x_1 \in A_1, x_2 \in A_2, \ldots, x_n \in A_n$$

where we already know that A_1 is a type, A_2 is a type in the context $x_1 \in A_1$, and so on. The explanations of hypothetical judgements are made by induction on the length of a context. For instance, to know that:

$$A \text{ type } [x \in C]$$

means that, for an arbitrary object c of type C, $A[x \leftarrow c]$ is a type, that is, A is a type when c is substituted for x. Furthermore, we must know that if c and d are identical objects of type C, then $A[x \leftarrow c]$ and $A[x \leftarrow d]$ are the same types64†source.

15.9.3 Function Types

To form a function type, if we have a type A and a family B of types over A, we can form the dependent function type $(x \in A)B$. An object c of type $(x \in A)B$ means that when applied to an object a of type A, we get an object $c(a)$ in $B[x \leftarrow a]$, and $c(a)$ and $c(b)$ are identical objects of type $B[x \leftarrow a]$ when a and b are identical objects of type A64†source.

15.9.4 The Type `Set`

The objects in the type `Set` consist of inductively defined sets. To know that `Set` is a type, we must explain what a set is and what it means for two sets to be the same. A canonical element of a set is an element in constructor form, e.g., 0 and the successor function for natural numbers.

Two sets are the same if an element of one set is also an element of the other and if two equal elements of one set are also equal elements of the other. This explanation justifies the rule:

$$\texttt{Set} \text{ type}$$

15.10 Propositional Logic

Type theory can be used as a logical framework to represent different theories. A theory is presented by a list of typings and definitions:

- **Typings:** $c_1 \in A_1, \ldots, c_n \in A_n$
- **Definitions:** $d_1 = e_1 \in B_1, \ldots, d_m = e_m \in A_m$

The basic types of Martin-Löf's type theory are `Set` and the types of elements of particular sets. For example, the primitive constant `&` for conjunction is introduced by the declaration:

$$\& \in (\mathtt{Set}; \mathtt{Set})\mathtt{Set}$$

From this declaration, we obtain the clause for conjunction in the inductive definition of formulas in the propositional calculus:

$$A \in \mathtt{Set} \quad B \in \mathtt{Set} \quad A\&B \in \mathtt{Set}$$

15.11 Set Theory

This section introduces a theory of sets with natural numbers, lists, functions, etc., useful for specifying and implementing computer programs. We introduce the following sets and their respective constructors, selectors, and rules.

15.11.1 The Set of Boolean Values

The set of Boolean values is introduced by the declaration:

$$\mathtt{Bool} \in \mathtt{Set}$$

with the constructors:

$$\mathtt{true} \in \mathtt{Bool} \quad \mathtt{false} \in \mathtt{Bool}$$

The principal selector `if` for Boolean values is defined by:

$$\mathtt{if} \in (\mathtt{C} \in (\mathtt{Bool})\mathtt{Set}; b \in \mathtt{Bool}; \mathtt{C(true)}; \mathtt{C(false)})\mathtt{C(b)}$$

with the defining equations:

```
if(C, true, a, b) = a
if(C, false, a, b) = b
```

15.11.2 The Empty Set

The empty set $\{\}$ is defined with no constructors:

$$\{\} \in \mathtt{Set}$$

The selector `case` for the empty set is defined by:

$$\mathtt{case} \in (\mathtt{C} \in (\{\})\mathtt{Set}; a \in \{\})\mathtt{C(a)}$$

This corresponds to the absurd proposition.

15.11.3 The Set of Natural Numbers

The set of natural numbers $\mathbb{N}$ is introduced with the constructors:

$$0 \in \mathbb{N}$$

$$\mathtt{succ} \in (\mathtt{n} \in \mathbb{N})\mathbb{N}$$

The selector `natrec` for primitive recursion is defined by:

$\mathtt{natrec} \in (\mathtt{C} \in (\mathbb{N})\mathtt{Set}; d \in \mathtt{C(0)}; e \in (\mathtt{x} \in \mathbb{N}; \mathtt{y} \in \mathtt{C(x)})\mathtt{C(succ(x))}; n \in \mathbb{N})\mathtt{C(n)}$

with the defining equations:

$$\mathtt{natrec(C,\ d,\ e,\ 0)\ =\ d}$$

$$\mathtt{natrec(C,\ d,\ e,\ succ(m))\ =\ e(m,\ natrec(C,\ d,\ e,\ m))}$$

15.11.4 The Set of Lists

The set of lists `List(A)` is introduced with the constructors:

$$\mathtt{nil} \in (\mathtt{A} \in \mathtt{Set})\mathtt{List(A)}$$

$$\mathtt{cons} \in (\mathtt{A} \in \mathtt{Set}; a \in \mathtt{A}; l \in \mathtt{List(A)})\mathtt{List(A)}$$

The selector `listrec` for lists is defined by:

$\mathtt{listrec} \in (\mathtt{A} \in \mathtt{Set}; \mathtt{C} \in (\mathtt{List(A)})\mathtt{Set}; c \in \mathtt{C(nil(A))}; e \in (\mathtt{x} \in \mathtt{A}; \mathtt{y} \in \mathtt{List(A)}; \mathtt{z} \in \mathtt{C(y)})\mathtt{C(cons(A,\ x,\ y))}; l \in \mathtt{List(A)})\mathtt{C(l)}$

with the defining equations:

$$\mathtt{listrec(A,\ C,\ c,\ e,\ nil(A))\ =\ c}$$

$$\mathtt{listrec(A,\ C,\ c,\ e,\ cons(A,\ a,\ l))\ =\ e(l,\ a,\ listrec(A,\ C,\ c,\ e,\ l))}$$

15.11.5 Disjoint Union of Two Sets

For two sets A and B, the disjoint union $A + B$ is introduced with the constants:

$$+ \in (\mathtt{A,\ B} \in \mathtt{Set})\mathtt{Set}$$

$$\mathtt{inl} \in (\mathtt{A,\ B} \in \mathtt{Set}; \mathtt{A})\mathtt{A+B}$$

$$\mathtt{inr} \in (\mathtt{A,\ B} \in \mathtt{Set}; \mathtt{B})\mathtt{A+B}$$

The selector `when` is defined by:

$\texttt{when} \in (\texttt{A, B} \in \texttt{Set}; \texttt{C} \in (\texttt{A+B})\texttt{Set}; e \in (\texttt{x} \in \texttt{A})\texttt{C(inl(A, B, x))}; f \in (\texttt{y} \in \texttt{B})\texttt{C(inr(A, B, y))}; p \in \texttt{A+B})\texttt{C(p)}$

with the defining equations:

```
when(A, B, C, e, f, inl(A, B, a)) = e(a)

when(A, B, C, e, f, inr(A, B, b)) = f(b)
```

15.11.6 Propositional Equality

Propositional equality is introduced with the type:

$$\texttt{Id} \in (\texttt{X} \in \texttt{Set}; \texttt{a} \in \texttt{X}; \texttt{b} \in \texttt{X})\texttt{Set}$$

The constructor for propositional equality is:

$$\texttt{id} \in (\texttt{X} \in \texttt{Set}; \texttt{x} \in \texttt{X})\texttt{Id(X, x, x)}$$

The selector `idpeel` is introduced by:

$\texttt{idpeel} \in (\texttt{A} \in \texttt{Set}; \texttt{C} \in (\texttt{x, y} \in \texttt{A}; \texttt{e} \in \texttt{Id(A, x, y)})\texttt{Set}; \texttt{a, b} \in \texttt{A}; \texttt{e} \in \texttt{Id(A, a, b)}; \texttt{d} \in (\texttt{x} \in \texttt{A})\texttt{C(x, x, id(A, x))})\texttt{C(a, b, e)}$

with the defining equation:

```
idpeel(A, C, a, b, id(A, a), d) = d(a)
```

15.12 Extended Concepts and Applications

15.12.1 Higher-Order Logic and Type Theory

Gilles Barthe's notes on Dependent Type Theory and Higher-Order Logic introduce a foundational language for defining mathematical objects, performing computations, and reasoning about these objects. The language underlies several proof assistants like Coq, Epigram, and Agda, and a few programming languages like Cayenne and DML63†source64†source.

15.12.2 Proof Assistants

Proof assistants implement type theories and higher-order logics to specify and reason about mathematics. They feature expressive specification languages that allow encoding of complex data structures and mathematical theories. Interactive proofs ensure:

- Theorems are applied with the correct hypotheses.
- Functions are applied to the correct arguments.
- No missing cases in proofs or function definitions.
- No illicit logical step (all reasoning is reduced to elementary steps).

Completed proofs are represented by proof objects that can be easily checked by a small, trusted proof-checker, providing the highest correctness guarantees.

15.12.3 Type Theory and the Curry-Howard Isomorphism

Type theory is a programming language for writing algorithms where all functions are total and terminating, making convertibility decidable. Type theory is also a language for proofs via the Curry-Howard isomorphism:

$$\text{Propositions} = \text{Types}$$

$$\text{Proofs} = \text{Terms}$$

$$\text{Proof-Checking} = \text{Type-Checking}$$

The underlying logic is constructive, though classical logic can be recovered with an axiom or a control operator63†source.

15.12.4 Pure Type Systems and the Lambda Cube

Pure Type Systems (PTS) provide a uniform framework for typed lambda-calculi, accommodating various binding choices. The lambda cube extends the simply-typed lambda calculus ($\lambda \to$) with:

- System F: extended with a new binder Λ for type variables.
- System Fω: extended with a new binder λ for type variables (of arbitrary kind) at the type level.

The Calculus of Constructions (λC) combines all three features, formalized as a particular Pure Type System63†source.

15.12.5 Dependent Types and Conversion

In dependent type systems, one can quantify on terms in types, allowing for the formation of dependent types like:

$$\Pi x : \mathrm{nat}.\mathrm{eq}\, x\, (0 + x) : \star$$

where eq is an equality predicate and eqrefl is a proof of reflexivity64†source.

15.12.6 Propositional Logic in Type Theory

Martin-Löf's type theory can be used to represent different theories, presenting them with typings and definitions. The primitive constant for conjunction (&) in the propositional calculus is defined as:

$$\& \in (\mathtt{Set}; \mathtt{Set})\mathtt{Set}$$

yielding the clause:

$$A \in \mathtt{Set} \quad B \in \mathtt{Set} \quad A\&B \in \mathtt{Set}$$

This reflects how type theory incorporates logical constructs into its framework64†source.

15.13 Bibliography

Bibliography

[1] Per Martin-Löf. *Intuitionistic Type Theory*. Bibliopolis, 1984.

[2] Gilles Barthe. *Dependent Type Theory and Higher-Order Logic*. INRIA Sophia-Antipolis, France.

[3] Jason Koenig. *Week 4 Lecture Notes on Dependent Type Theory*. 2013.

Part IV

Short Notes on Homotopy Theory

15.14 Basic Concepts in Homotopy Theory

15.14.1 Homotopy and Homotopy Equivalence

Two continuous maps $f, g : X \to Y$ are said to be homotopic if there exists a continuous map $H : X \times [0, 1] \to Y$ such that $H(x, 0) = f(x)$ and $H(x, 1) = g(x)$ for all $x \in X$. This map H is called a homotopy between f and g. If such a homotopy exists, we write $f \sim g$.

A homotopy equivalence between spaces X and Y is a pair of maps $f : X \to Y$ and $g : Y \to X$ such that $g \circ f \sim \mathrm{id}_X$ and $f \circ g \sim \mathrm{id}_Y$. In this case, X and Y are said to be homotopy equivalent, or of the same homotopy type.

15.14.2 Fundamental Group

The fundamental group $\pi_1(X, x_0)$ of a topological space X with basepoint x_0 is the group of homotopy classes of loops based at x_0. A loop is a continuous map $\gamma : [0, 1] \to X$ with $\gamma(0) = \gamma(1) = x_0$. The group operation is given by concatenation of loops.

The concatenation of two loops $\alpha, \beta : [0, 1] \to X$ based at x_0 is defined by

$$(\alpha \cdot \beta)(t) = \begin{cases} \alpha(2t) & 0 \le t \le 1/2, \\ \beta(2t - 1) & 1/2 \le t \le 1. \end{cases}$$

The fundamental group $\pi_1(X, x_0)$ is a group under the operation of concatenation of loops. The identity element is the constant loop at x_0, and the inverse of a loop γ is the loop traversed in the opposite direction.

To show associativity, let α, β, γ be three loops based at x_0. We need to show that $(\alpha \cdot \beta) \cdot \gamma \sim \alpha \cdot (\beta \cdot \gamma)$. Define a homotopy $H : [0, 1]^2 \to X$ by

$$H(s, t) = \begin{cases} \alpha(4st) & 0 \le t \le 1/2, \\ \beta(4st - 2s) & 1/2 \le t \le 1, \end{cases}$$

which shows $(\alpha \cdot \beta) \cdot \gamma \sim \alpha \cdot (\beta \cdot \gamma)$. The identity loop is given by $\epsilon(t) = x_0$ for all $t \in [0, 1]$, and the inverse loop $\gamma^{-1}(t) = \gamma(1 - t)$ satisfies $\gamma \cdot \gamma^{-1} \sim \epsilon$.

15.14.3 Higher Homotopy Groups

For $n \geq 2$, the n-th homotopy group $\pi_n(X, x_0)$ is defined as the set of homotopy classes of maps $f : (S^n, s_0) \to (X, x_0)$, where S^n is the n-dimensional sphere and s_0 is a basepoint. These groups are abelian for $n \geq 2$.

The n-th homotopy group $\pi_n(X, x_0)$ is defined as

$$\pi_n(X, x_0) = \{[f] \mid f : (S^n, s_0) \to (X, x_0)\}.$$

For $n \geq 2$, the homotopy group $\pi_n(X, x_0)$ is abelian.

Consider two maps $f, g : S^n \to X$ representing elements of $\pi_n(X, x_0)$. We define the sum $f + g$ using the standard decomposition of S^n into two n-dimensional hemispheres glued along S^{n-1}. The map $f + g$ is then given by

$$(f+g)(x) = \begin{cases} f(x) & x \in \text{upper hemisphere}, \\ g(x) & x \in \text{lower hemisphere}. \end{cases}$$

Since S^{n-1} is contractible within S^n, we can construct a homotopy that shows $f + g = g + f$, proving that $\pi_n(X, x_0)$ is abelian.

15.14.4 Homotopy Groups of Spheres

The homotopy groups of spheres, $\pi_n(S^m)$, are fundamental objects in homotopy theory. The computation of these groups is complex and reveals deep properties of topological spaces. For example, $\pi_3(S^2)$ is isomorphic to $\mathbb{Z}$, indicating a non-trivial structure.

$\pi_3(S^2) \cong \mathbb{Z}$

We use the Hopf fibration $S^3 \to S^2$ with fiber S^1. Consider the long exact sequence of homotopy groups associated with this fibration:

$$\cdots \to \pi_3(S^1) \to \pi_3(S^3) \to \pi_3(S^2) \to \pi_2(S^1) \to \cdots$$

Since $\pi_3(S^1) = 0$ and $\pi_2(S^1) = 0$, we get an isomorphism $\pi_3(S^3) \to \pi_3(S^2)$. Given that $\pi_3(S^3) \cong \mathbb{Z}$, we conclude that $\pi_3(S^2) \cong \mathbb{Z}$.

15.14.5 Cohomology and Homotopy

Cohomology theories, such as singular cohomology, can be used to study homotopy types of spaces. The cohomology ring of a space provides algebraic invariants that help in distinguishing between different homotopy types.

The cohomology groups $H^n(X; A)$ of a space X with coefficients in an abelian group A are defined using the cochain complex $C^*(X; A)$, where $C^n(X; A)$ is the group of singular n-cochains and the coboundary operator $\delta : C^n(X; A) \to C^{n+1}(X; A)$ satisfies $\delta^2 = 0$.

If two spaces X and Y are homotopy equivalent, then their cohomology rings are isomorphic.

Suppose $f : X \to Y$ and $g : Y \to X$ are homotopy equivalences such that $g \circ f \sim \mathrm{id}_X$ and $f \circ g \sim \mathrm{id}_Y$. The induced maps on cohomology $f^* : H^n(Y; A) \to H^n(X; A)$ and $g^* : H^n(X; A) \to H^n(Y; A)$ satisfy $g^* \circ f^* = \mathrm{id}_{H^n(Y;A)}$ and $f^* \circ g^* = \mathrm{id}_{H^n(X;A)}$, making f^* and g^* isomorphisms.

15.15 Fiber Bundles and Fibrations

15.15.1 Fiber Bundles

A fiber bundle is a map $p : E \to B$ such that for every point $b \in B$, there exists an open neighborhood U of b and a homeomorphism $\phi : p^{-1}(U) \to U \times F$ that commutes with the projection to U. Here, E is the total space, B is the base space, and F is the fiber.

A fiber bundle (E, B, p) consists of a continuous surjection $p : E \to B$, called the projection, such that for every $b \in B$, there exists an open neighborhood $U \subset B$ and a homeomorphism $\phi : p^{-1}(U) \to U \times F$ where F is the fiber.

If $p : E \to B$ is a fiber bundle with fiber F and base B, then the homotopy type of F does not depend on the choice of the point $b \in B$.

Let $b_0, b_1 \in B$. Choose a path $\gamma : [0, 1] \to B$ with $\gamma(0) = b_0$ and $\gamma(1) = b_1$. The homotopy lifting property of the fiber bundle provides a map $H : F \times [0, 1] \to E$ such that $H(f, 0) = f$ and $H(f, t) \in p^{-1}(\gamma(t))$ for all $f \in F$ and $t \in [0, 1]$. This gives a homotopy equivalence between $p^{-1}(b_0)$ and $p^{-1}(b_1)$, showing that the fibers are homotopy equivalent.

15.15.2 Principal Bundles

A principal bundle is a fiber bundle $p : P \to B$ with a structure group G acting on the fiber F such that $P \times_G F \cong E$. Principal bundles play a key role in various areas of mathematics, including gauge theory and the study of characteristic classes.

A principal G-bundle is a fiber bundle $p : P \to B$ together with a continuous right action of G on P such that $p(pg) = p(p)$ for all $p \in P$ and $g \in G$, and (P, B, p) is locally trivial.

For any principal G-bundle $p : P \to B$, the quotient space P/G is homeomorphic to the base space B.

Define a map $\phi : P/G \to B$ by $\phi([p]) = p(p)$. This map is well-defined because the action of G preserves the fibers. It is continuous because the quotient topology on P/G is the finest topology that makes the projection $P \to P/G$ continuous. The map ϕ is bijective because for any $b \in B$, there is a fiber $p^{-1}(b)$ that maps to b. Since B is locally compact and Hausdorff, and P/G is locally compact, the map ϕ is a homeomorphism.

15.15.3 Vector Bundles

Vector bundles are fiber bundles where the fiber is a vector space. An important example is the tangent bundle of a manifold, which associates a tangent space to each point on the manifold. Vector bundles are central in differential geometry and topology.

A vector bundle (E, B, p) is a fiber bundle where the fiber F is a vector space, and the transition functions are linear isomorphisms.

Every vector bundle E over a paracompact base space B admits a Riemannian metric, i.e., a smoothly varying inner product on each fiber.

Cover B by open sets U_α over which E is trivial. On each U_α, choose a local trivialization $\phi_\alpha : p^{-1}(U_\alpha) \to U_\alpha \times \mathbb{R}^n$. Define a local inner product on $U_\alpha \times \mathbb{R}^n$ by the standard inner product on $\mathbb{R}^n$. Use a partition of unity subordinate to the cover $\{U_\alpha\}$ to patch together these local inner products into a global inner product on E.

15.15.4 Fibrations

A fibration is a map $p : E \to B$ that satisfies the homotopy lifting property: given any map $f : W \to E$ and any homotopy $H : W \times I \to B$ with $H(w, 0) = p(f(w))$, there exists a homotopy $\tilde{H} : W \times I \to E$ such that $p(\tilde{H}(w, t)) = H(w, t)$ and $\tilde{H}(w, 0) = f(w)$.

A fibration $p : E \to B$ satisfies the homotopy lifting property (HLP) if for any space W, any map $f : W \to E$, and any homotopy $H : W \times I \to B$ with $H(w, 0) = p(f(w))$, there exists a homotopy $\tilde{H} : W \times I \to E$ such that $p(\tilde{H}(w, t)) = H(w, t)$ and $\tilde{H}(w, 0) = f(w)$.

If $p : E \to B$ is a fibration with fiber F, then there is an associated long exact sequence of homotopy groups:

$$\cdots \to \pi_{n+1}(B) \to \pi_n(F) \to \pi_n(E) \to \pi_n(B) \to \pi_{n-1}(F) \to \cdots$$

Consider the path space $P(B, b_0)$ of paths in B starting at b_0. The fibration $p : E \to B$ induces a fibration $P(E, e_0) \to P(B, b_0)$ with fiber ΩF. By analyzing the homotopy types of the fibers and base spaces in the associated fibration sequences, we obtain the long exact sequence of homotopy groups.

15.15.5 Examples of Fiber Bundles

- **Trivial Bundle**: A simple example of a fiber bundle is the trivial bundle $E = B \times F$ with projection map $p : B \times F \to B$.
- **Hopf Fibration**: The Hopf fibration is a non-trivial fiber bundle $S^3 \to S^2$ with fiber S^1.
- **Tangent Bundle**: The tangent bundle TM of a manifold M is a vector bundle whose fiber at each point is the tangent space to the manifold at that point.

15.16 Exact Sequences in Homotopy

15.16.1 Long Exact Sequence of a Fibration

Given a fibration $p : E \to B$ with fiber F, there is an associated long exact sequence of homotopy groups:

$$\cdots \to \pi_{n+1}(B) \to \pi_n(F) \to \pi_n(E) \to \pi_n(B) \to \pi_{n-1}(F) \to \cdots$$

This sequence provides important information about the relationships between the homotopy groups of the total space, the base space, and the fiber.

15.16.2 Mayer-Vietoris Sequence

The Mayer-Vietoris sequence is a tool in algebraic topology that provides a long exact sequence of homology groups for a space that is the union of two subspaces. It is analogous to the long exact sequence in homotopy theory and is useful for computing homology groups.

Let $X = U \cup V$ be a space that is the union of two open sets U and V. The Mayer-Vietoris sequence is a long exact sequence:

$$\cdots \to H_n(U \cap V) \to H_n(U) \oplus H_n(V) \to H_n(X) \to H_{n-1}(U \cap V) \to \cdots$$

Consider the short exact sequence of chain complexes:

$$0 \to C_*(U \cap V) \to C_*(U) \oplus C_*(V) \to C_*(X) \to 0.$$

Applying the snake lemma, we obtain the long exact sequence in homology.

15.16.3 Gysin Sequence

The Gysin sequence is a long exact sequence associated with a sphere bundle. It relates the cohomology of the total space, the base space, and the fiber, and is particularly useful in the study of characteristic classes and their applications.

For a sphere bundle $S^{n-1} \to E \to B$, the Gysin sequence is:

$$\cdots \to H^k(B) \xrightarrow{\cup e} H^{k+n}(B) \to H^{k+n}(E) \to H^{k+1}(B) \xrightarrow{\cup e} \cdots$$

where e is the Euler class of the bundle.

The proof uses the Leray-Serre spectral sequence of the fibration and the Thom isomorphism theorem, relating the cohomology of the total space to the base and the Euler class.

15.16.4 Exact Sequence of a Cofibration

Given a cofibration $i : A \to X$, there is an associated long exact sequence of homotopy groups:

$$\cdots \to \pi_n(A) \to \pi_n(X) \to \pi_n(X/A) \to \pi_{n-1}(A) \to \cdots$$

This sequence helps understand the homotopy type of the quotient space X/A in terms of the homotopy types of A and X.

If $i : A \to X$ is a cofibration, then the sequence

$$\cdots \to \pi_n(A) \xrightarrow{i_*} \pi_n(X) \xrightarrow{j_*} \pi_n(X/A) \xrightarrow{\partial} \pi_{n-1}(A) \to \cdots$$

is exact.

Consider the mapping cone $C_i = X \cup_i CA$. The long exact sequence of the pair (C_i, X) and the homotopy equivalence $C_i \simeq X/A$ yield the desired long exact sequence in homotopy.

15.17 Cofibrations and Homotopy Extensions

15.17.1 Cofibrations

A map $i : A \to X$ is a cofibration if it satisfies the homotopy extension property: given any map $f : X \to Y$ and a homotopy $H : A \times I \to Y$ such that $H(a, 0) = f(i(a))$, there exists an extension $\tilde{H} : X \times I \to Y$ such that $\tilde{H}(x, 0) = f(x)$ and $\tilde{H}(i(a), t) = H(a, t)$.

A map $i : A \to X$ is a cofibration if for any map $f : X \to Y$ and any homotopy $H : A \times I \to Y$ with $H(a, 0) = f(i(a))$, there exists an extension $\tilde{H} : X \times I \to Y$ such that $\tilde{H}(x, 0) = f(x)$ and $\tilde{H}(i(a), t) = H(a, t)$.

If $i : A \to X$ is a cofibration, then the mapping cone $C(i) = (A \times I) \cup X/(a, 1) \sim i(a)$ has the homotopy type of X/A.

The map $\phi : C(i) \to X/A$ defined by $\phi([x]) = \overline{x}$ is a homotopy equivalence, where $\overline{x}$ is the image of x in the quotient space X/A. This can be shown using the homotopy extension property to construct a homotopy inverse.

15.17.2 Mapping Cylinders and Cones

The mapping cylinder $M(i)$ of a map $i : A \to X$ is defined as

$$M(i) = (A \times I) \cup X/ \sim$$

where $(a, 1) \sim i(a)$ for $a \in A$. The mapping cone $C(i)$ is the quotient space

$$C(i) = M(i)/(A \times \{0\}).$$

If $i : A \to X$ is a cofibration, then the inclusion $X \hookrightarrow C(i)$ is a homotopy equivalence.

Define a homotopy inverse $H : C(i) \to X$ by $H([a, t]) = i(a)$ for $t \in [0, 1)$ and $H([x]) = x$ for $x \in X$. The homotopy $G : C(i) \times I \to C(i)$ defined by $G([a, t], s) = [a, (1 - s)t]$ and $G([x], s) = [x]$ shows that $H \circ \mathrm{id} \sim \mathrm{id}$ and $\mathrm{id} \circ H \sim \mathrm{id}$.

15.17.3 Cylinder and Path Spaces

The cylinder $X \times I$ and path space X^I play crucial roles in homotopy theory. These constructions allow the definition of homotopy, the study of path spaces, and the formulation of homotopy lifting properties.

The path space X^I is the space of continuous maps $f : I \to X$ with the compact-open topology.

The map $\pi : X^I \to X$ defined by $\pi(f) = f(1)$ is a fibration with fiber the based loop space ΩX.

Given a map $H : W \times I \to X$ and a map $f : W \to X^I$ such that $H(w, 0) = \pi(f(w))$, define $\tilde{H} : W \times I \to X^I$ by $\tilde{H}(w, t)(s) = H(w, st)$. This homotopy $\tilde{H}$ lifts the homotopy H from X to X^I.

15.17.4 Homotopy Coherence

Homotopy coherence is the concept that extends the notion of a homotopy to higher dimensions. It involves a sequence of homotopies that coherently deform one map to another. This concept is essential in the study of higher categorical structures.

A homotopy coherent diagram is a diagram of spaces and maps where the compositions and higher homotopies are coherently defined by a sequence of higher homotopies.

If X and Y are homotopy coherent diagrams, then any homotopy between X and Y induces a homotopy equivalence between the homotopy colimits of X and Y.

The homotopy colimit $\text{hocolim} X$ is constructed using the bar construction, and any homotopy between X and Y induces a homotopy between their respective bar constructions. This homotopy lifts to a homotopy equivalence of the homotopy colimits.

15.18 Homotopy Theory of CW Complexes

15.18.1 CW Complexes

A CW complex is a topological space constructed by gluing cells of increasing dimension. Formally, a CW complex is a space X with a filtration

$$X^0 \subset X^1 \subset X^2 \subset \cdots \subset X$$

where X^n is obtained from X^{n-1} by attaching n-cells via characteristic maps.

A CW complex X is a space constructed inductively by attaching n-cells e_α^n to X^{n-1} via maps $\phi_\alpha : S^{n-1} \to X^{n-1}$.

If X is a CW complex, then X has the homotopy type of a simplicial complex.

Given a CW complex X, construct a simplicial complex K by replacing each cell e^n_α with a simplex of dimension n. The characteristic maps ϕ_α induce the attaching maps of the simplicial complex, preserving the homotopy type.

15.18.2 Cell Attachments and Homotopy

Cell attachments in CW complexes allow for the construction of spaces with controlled homotopy types. Attaching an n-cell to a CW complex corresponds to extending the complex by adding higher-dimensional features.

If X is a CW complex and e^n is an n-cell attached to X, then the inclusion $X \hookrightarrow X \cup e^n$ is a cofibration.

The map $i : X \to X \cup e^n$ satisfies the homotopy extension property because the attachment of e^n can be viewed as a pushout in the category of topological spaces. The homotopy lifting property follows from the definition of a CW complex.

15.18.3 Homotopy Groups of CW Complexes

CW complexes have nice properties regarding homotopy groups. The homotopy groups of a CW complex X can be computed using its cellular structure. If X is a CW complex with $\{e^n_\alpha\}$ as its n-cells, then the n-th homotopy group $\pi_n(X)$ can be analyzed in terms of these cells.

If X is a CW complex, then the n-th homotopy group $\pi_n(X)$ is isomorphic to the group of cellular n-cycles modulo the boundaries of $(n+1)$-cells.

Consider the cellular chain complex $C_*(X)$ of X. The cellular boundary map $\partial_n : C_n(X) \to C_{n-1}(X)$ defines the homology groups $H_n(C_*(X))$. By the Hurewicz theorem, $\pi_n(X) \cong H_n(X)$ for $n \geq 2$, where $H_n(X)$ is the homology of the cellular chain complex.

15.18.4 CW Approximation

The CW approximation theorem states that for any space X, there exists a CW complex Y and a homotopy equivalence $Y \to X$. This theorem allows the replacement of arbitrary spaces by CW complexes without changing their homotopy type.

For any topological space X, there exists a CW complex Y and a map $f : Y \to X$ such that f is a homotopy equivalence.

Construct Y by attaching cells to X in increasing dimensions. Start with the 0-skeleton X^0, and inductively attach n-cells to kill the homotopy groups $\pi_n(X^n)$ until X has the homotopy type of a CW complex. The map f is defined by the inclusion of the skeletons.

15.19 Spectral Sequences in Homotopy Theory

15.19.1 Introduction to Spectral Sequences

A spectral sequence is a computational tool used in algebraic topology to compute homology and cohomology groups. It provides a sequence of pages, each containing groups and differentials, converging to the desired homotopy or homology groups.

A spectral sequence $\{E_r^{p,q}, d_r\}$ is a sequence of bigraded groups $E_r^{p,q}$ and differentials $d_r : E_r^{p,q} \to E_r^{p+r,q-r+1}$ such that $d_r^2 = 0$ and $E_{r+1}^{p,q} = \ker(d_r)/\mathrm{im}(d_r)$.

The spectral sequence $\{E_r^{p,q}, d_r\}$ converges to the graded group associated with a filtered complex.

Given a filtered complex $\{F_pC_*, d\}$, the associated graded complex $\mathrm{gr}_p C_* = F_pC_*/F_{p+1}C_*$ yields the E_0-page of the spectral sequence. Inductively, the differentials d_r are defined by the boundary maps of the filtered complex, and the convergence follows from the stability of the filtration.

15.19.2 Serre Spectral Sequence

The Serre spectral sequence is associated with a fibration $F \to E \to B$. It provides a method to compute the homology of E from the homology of F and B.

For a fibration $F \to E \to B$ with $\pi_1(B)$ acting trivially on $H_*(F)$, there is a spectral sequence $E_2^{p,q} = H_p(B; H_q(F))$ converging to $H_*(E)$.

Consider the Leray-Serre spectral sequence with E_2-page $E_2^{p,q} = H_p(B; H_q(F))$. The differentials $d_r : E_r^{p,q} \to E_r^{p+r,q-r+1}$ are induced by the boundary maps of the fiber bundle. The spectral sequence converges to $H_*(E)$ by the Eilenberg-Moore comparison theorem.

15.19.3 Adams Spectral Sequence

The Adams spectral sequence is a powerful tool in stable homotopy theory. It is used to compute stable homotopy groups of spheres and other spectra. This spectral sequence converges to the stable homotopy groups and is based on Ext groups in the category of comodules over a Hopf algebra.

The Adams spectral sequence $\{E_r^{s,t}, d_r\}$ for computing stable homotopy groups of spheres π_*^s has E_2-page given by $\mathrm{E}_2^{s,t}$
$= \mathrm{Ext}_{A_*}^{s,t}(H_*(S^0), \mathbb{Z}/2)$.

Consider the cobar complex $C^{s,t}(A_*, H_*(S^0), \mathbb{Z}/2)$ with differential d induced by the coaction of the Steenrod algebra A_* on $H_*(S^0)$. The Ext groups $\mathrm{Ext}_{A_*}^{s,t}(H_*(S^0), \mathbb{Z}/2)$ form the E_2-page, and the differentials are induced by the cobar complex differential. The convergence follows from the Adams spectral sequence comparison theorem.

15.19.4 Eilenberg-Moore Spectral Sequence

The Eilenberg-Moore spectral sequence is used to compute the cohomology of the pullback of a fibration. It provides a way to understand the cohomology of complex spaces by breaking down their structure into more manageable pieces.

For a pullback square

$$\begin{array}{ccc} P & \to & E \\ \downarrow & & \downarrow \\ B & \to & B' \end{array}$$

there is an Eilenberg-Moore spectral sequence $E_2^{p,q} =$
$\mathrm{Tor}_{p,q}^{H^*(B')}(H^*(B), H^*(E))$ converging to $H^*(P)$.

Consider the cobar resolution of the differential graded algebra $H^*(B')$ with coefficients in $H^*(B)$ and $H^*(E)$. The E_2-page is given by the homology of the cobar complex, yielding $\mathrm{Tor}_{p,q}^{H^*(B')}$
$(\mathrm{H}^*(B), H^*(E))$. The convergence follows from the spectral sequence comparison theorem.

15.19.5 Leray Spectral Sequence

The Leray spectral sequence is associated with a continuous map $f : X \to Y$ and relates the cohomology of X to the cohomology of Y and the fibers of f. It is a valuable tool for computing the cohomology of fiber bundles and other structured spaces.

For a continuous map $f : X \to Y$ and a sheaf $\mathcal{F}$ on X, the Leray spectral sequence is

$$E_2^{p,q} = H^p(Y, R^q f_* \mathcal{F}) \Rightarrow H^{p+q}(X, \mathcal{F}).$$

Consider the derived functors $R^q f_* \mathcal{F}$ of the sheaf $f_* \mathcal{F}$. The Grothendieck spectral sequence associated with the composition of functors f_* and R^q provides the desired spectral sequence. The convergence follows from the fact that the derived functors stabilize.

15.20 Applications of Homotopy Theory

15.20.1 Homotopy Groups of Spheres

One of the major applications of homotopy theory is the computation of homotopy groups of spheres. These computations reveal deep properties of topological spaces and their mappings. For instance, the homotopy groups $\pi_n(S^m)$ for various n and m provide insights into the structure and classification of spheres.

The homotopy groups of spheres $\pi_n(S^m)$ are non-trivial for certain values of n and m, exhibiting intricate algebraic structures.

The non-triviality of $\pi_n(S^m)$ is shown using various tools, including the Freudenthal suspension theorem, the Hurewicz theorem, and the stable homotopy groups. For example, the non-triviality of $\pi_3(S^2) \cong \mathbb{Z}$ follows from the Hopf fibration $S^3 \to S^2$.

15.20.2 Homotopy and Fiber Bundles

Homotopy theory plays a crucial role in the study of fiber bundles. The homotopy lifting property and exact sequences associated with fibrations are essential tools in the analysis of fiber bundles. Applications include the classification of bundles and the study of characteristic classes.

The classification of fiber bundles up to homotopy equivalence is determined by the homotopy classes of maps into a classifying space.

Consider the classifying space BG for a principal G-bundle. Any principal G-bundle $P \to B$ is classified by a homotopy class of maps $B \to BG$. The homotopy lifting property and the universal property of the classifying space ensure the classification up to homotopy equivalence.

15.20.3 Homotopy Theory in Algebraic Topology

Homotopy theory is foundational in algebraic topology, influencing the study of homological and cohomological properties of spaces. It provides techniques for classifying spaces, understanding continuous maps, and exploring the algebraic structures underlying topological spaces.

The homotopy category of topological spaces $\mathcal{H}\wr(Top)$ has a well-defined structure with morphisms given by homotopy classes of continuous maps.

Consider the category Top of topological spaces with continuous maps. Define the homotopy category $\mathcal{H}\wr(Top)$ with the same objects and morphisms given by homotopy classes of maps. The composition of morphisms is well-defined up to homotopy, and the identity morphism is given by the homotopy class of the identity map.

15.20.4 Homotopy and Category Theory

Homotopy theory intersects with category theory in the study of model categories, derived categories, and higher categories. These frameworks allow for the abstract treatment of homotopy-theoretic concepts and provide a deeper understanding of the relationships between different topological structures.

The category of simplicial sets **SSet** with the Kan model structure forms a model category with fibrations, cofibrations, and weak equivalences.

Consider the category **SSet** of simplicial sets. Define a map $f : X \to Y$ to be a fibration if it has the right lifting property with respect to all horn inclusions, a cofibration if it has the left lifting property with respect to trivial fibrations, and a weak equivalence if it induces isomorphisms on all homotopy groups. The axioms of a model category (MC1-MC5) are satisfied, making **SSet** a model category.

15.20.5 Homotopy Theory in Physics

In theoretical physics, homotopy theory is used in the study of topological defects, gauge theory, and string theory. Concepts like fiber bundles and characteristic classes are essential in the formulation of physical theories and the analysis of topological properties of physical systems.

In gauge theory, the classification of principal G-bundles over a base space X is determined by the homotopy classes of maps $[X, BG]$, where BG is the classifying space for G-bundles.

Consider a principal G-bundle $P \to X$. The classification theorem states that P is classified by a map $f : X \to BG$, where BG is the classifying space for G-bundles. This map f induces a homotopy equivalence $[X, BG]$, providing the classification.

15.21 Advanced Topics in Homotopy Theory

15.21.1 Model Categories

Model categories provide an abstract framework for homotopy theory. They generalize the notion of a topological space and continuous maps to more general settings, allowing the study of homotopy in a broader context. In a model category, one can define notions of weak equivalences, fibrations, and cofibrations, which facilitate the study of homotopy theory.

A model category is a category $\mathcal{C}$ equipped with three distinguished classes of morphisms: weak equivalences, fibrations, and cofibrations, satisfying five axioms (MC1-MC5) that generalize the properties of homotopy theory.

In a model category $\mathcal{C}$, any morphism can be factored into a cofibration followed by a trivial fibration and into a trivial cofibration followed by a fibration.

Consider a morphism $f : A \to B$ in $\mathcal{C}$. Use the small object argument to construct a factorization $A \xrightarrow{i} X \xrightarrow{p} B$ where i is a cofibration and p is a trivial fibration. Similarly, factor f as $A \xrightarrow{j} Y \xrightarrow{q} B$ where j is a trivial cofibration and q is a fibration. The axioms of a model category ensure the existence of these factorizations.

15.21.2 Homotopy Limits and Colimits

Homotopy limits and colimits extend the concepts of limits and colimits in category theory to homotopy theory. They provide a way to understand how homotopy interacts with these categorical constructions. Homotopy limits and colimits are used to study the behavior of spaces and maps under homotopy, providing insights into their global structure.

The homotopy limit of a diagram $D : \mathcal{I} \to \mathcal{C}$ in a model category $\mathcal{C}$ is the derived limit holim$D$ constructed using the total right derived functor of the limit.

In a model category $\mathcal{C}$, the homotopy limit holim$D$ of a diagram D is weakly equivalent to the limit of a fibrant replacement of D.

Consider a diagram $D : \mathcal{I} \to \mathcal{C}$. Replace D by a fibrant diagram D^f such that each object and morphism in D^f is fibrant. The homotopy limit holim$D$ is defined as the limit of D^f, and the weak equivalence follows from the properties of fibrant replacement and the definition of homotopy limits.

15.21.3 Stable Homotopy Theory

Stable homotopy theory studies the stable properties of homotopy groups by stabilizing suspension. It introduces the notion of spectra, which generalize spaces and allow for a more flexible and comprehensive study of homotopy theory. The stable homotopy category provides a setting for understanding phenomena that are invariant under suspension.

A spectrum $\mathcal{E}$ is a sequence of pointed spaces $\{E_n\}$ together with structure maps $\sigma_n : \Sigma E_n \to E_{n+1}$, where Σ denotes the suspension functor.

The stable homotopy category Ho(Spectra) is a triangulated category with distinguished triangles corresponding to cofiber sequences of spectra.

Consider the category of spectra with stable weak equivalences, stable fibrations, and stable cofibrations. Define the homotopy category Ho(Spectra) by inverting the stable weak equivalences. The distinguished triangles are defined by cofiber sequences, and the axioms of a triangulated category are satisfied.

15.21.4 Equivariant Homotopy Theory

Equivariant homotopy theory extends homotopy theory to spaces with group actions. It studies the homotopy properties of G-spaces, where G is a group acting on a space. Equivariant homotopy theory provides tools for understanding how symmetries affect the homotopy types of spaces and maps.

A G-space is a topological space X together with a continuous action of a group G on X, denoted $G \times X \to X$.

The equivariant homotopy category $\mathrm{Ho}_G(Top)$ is a model category with weak equivalences, fibrations, and cofibrations defined by equivariant maps.

Consider the category of G-spaces with G-equivariant maps. Define a weak equivalence as a G-map that induces weak equivalences on fixed point sets X^H for all subgroups $H \subseteq G$. Define fibrations and cofibrations similarly. The axioms of a model category are satisfied, making $\mathrm{Ho}_G(Top)$ a model category.

15.21.5 Axiomatic Homotopy Theory

Axiomatic homotopy theory aims to develop homotopy theory based on a set of axioms, similar to how homology theory is developed. This approach seeks to identify the fundamental properties and structures that define homotopy theory and provide a unified framework for its study.

An axiomatic homotopy theory is a category $\mathcal{C}$ equipped with a distinguished class of morphisms called weak equivalences, satisfying the axioms of a homotopy category.

Any axiomatic homotopy theory satisfying the axioms of a model category can be localized to form a homotopy category $\mathcal{H}\wr(\mathcal{C})$.

Consider the category $\mathcal{C}$ with weak equivalences. Localize $\mathcal{C}$ by formally inverting the weak equivalences, resulting in the homotopy category $\mathcal{H}\wr(\mathcal{C})$. The axioms of a model category ensure that the localization is well-defined and retains the homotopy-theoretic properties.

Bibliography

[1] A. Hatcher, *Algebraic Topology*, Cambridge University Press, 2002.

[2] S. Mac Lane, *Categories for the Working Mathematician*, Springer, 1998.

[3] J. P. May, *A Concise Course in Algebraic Topology*, University of Chicago Press, 1999.

[4] E. H. Spanier, *Algebraic Topology*, McGraw-Hill, 1966.

[5] R. Bott and L. W. Tu, *Differential Forms in Algebraic Topology*, Springer, 1982.