mathematics

MDPI

Article

Research on a General State Formalization Method from the Perspective of Logic

Siyuan Qiu [1] and Jianfeng Xu [2,*]

[1] School of Computer Science, Shanghai Jiao Tong University, Shanghai 200030, China; qsyqsyqsy@sjtu.edu.cn
[2] Koguan School of Law, China Institute for Smart Justice, School of Computer Science, Shanghai Jiao Tong University, Shanghai 200030, China
* Correspondence: xujf@sjtu.edu.cn

Abstract

As information plays an ever more central role across disciplines, the lack of a precise and reusable definition of state impedes comparison, measurement, and verification. Building on Objective Information Theory (OIT), this paper proposes a logic-based framework that defines the state of an object or system at a time point (or interval) as the semantic valuation of a set of well-formed formulas over a given domain and interpretation. Within first-order and higher-order logic—extended to infinitary logic when needed—we show how finite and broad classes of infinite structures can be characterized, drawing on core results from model theory. We then instantiate the framework in economics, sociology, computer science, and natural language, demonstrating that logic provides a unifying language for representing, reasoning about, and relating states across domains. Finally, we refine OIT by supplying a universal state representation that supports cross-domain exchange, measurement, and verification.

Keywords: objective information theory; logical systems; states; formal methods

MSC: 03B10; 03B16

Academic Editor: Marjan Mernik

Received: 12 August 2025
Revised: 7 September 2025
Accepted: 12 September 2025
Published: 18 October 2025

Citation: Qiu, S.; Xu, J. Research on a General State Formalization Method from the Perspective of Logic. *Mathematics* **2025**, *13*, 3324. https://doi.org/10.3390/math13203324

1. Introduction

Information, matter, and energy are fundamental to nature, yet there is no cross-disciplinary consensus on what information is or how to define state precisely [1]. Classical information theory focuses on coding and communication efficiency and offers limited tools to characterize states at the levels of definition, analysis, and processing [2–4]. Objective Information Theory (OIT), grounded in a mapping between ontology and carrier, provides postulates, mathematical definitions, and measurement systems that unify diverse information principles [5–7]. However, OIT treats information as an enabling mapping between states without a rigorous and general definition of state itself, which hinders cross-domain comparability, logical inference about "the same state," and precise measurement and verification of information mappings.

This paper proposes a universal, verifiable, and reusable definition of state: the state of an object or system at a time point (or interval) is the semantic valuation of a set of well-formed formulas over a given domain and interpretation. Leveraging results in model theory, we analyze when first-order and higher-order logic (and, when appropriate, infinitary logic) can fully characterize structures, from finite to broad classes of infinite ones. We then present representative cases from economics, sociology, computer science,

and natural language to show that logic serves as a bridge for state representation across domains. Our contributions are:

1. a unified axiomatization of states as interpretations of formulas, with four axioms (parameter reference, property expressibility, logical closure, temporal causality) and an existence–uniqueness result up to logical equivalence that fills a gap in OIT;
2. systematic links to model-theoretic milestones (categoricity, Skolem/non-categoricity, second-order Peano, Scott's isomorphism theorem) and conditions for characterizing some uncountable structures;
3. a cross-disciplinary case library for specification, verification, and measurement;
4. an articulation of a deeper unity: many domain-specific notions of state admit uniform logical expression and transformation.

Figure 1 illustrates logic as a cross-domain bridge and a universal descriptive language.

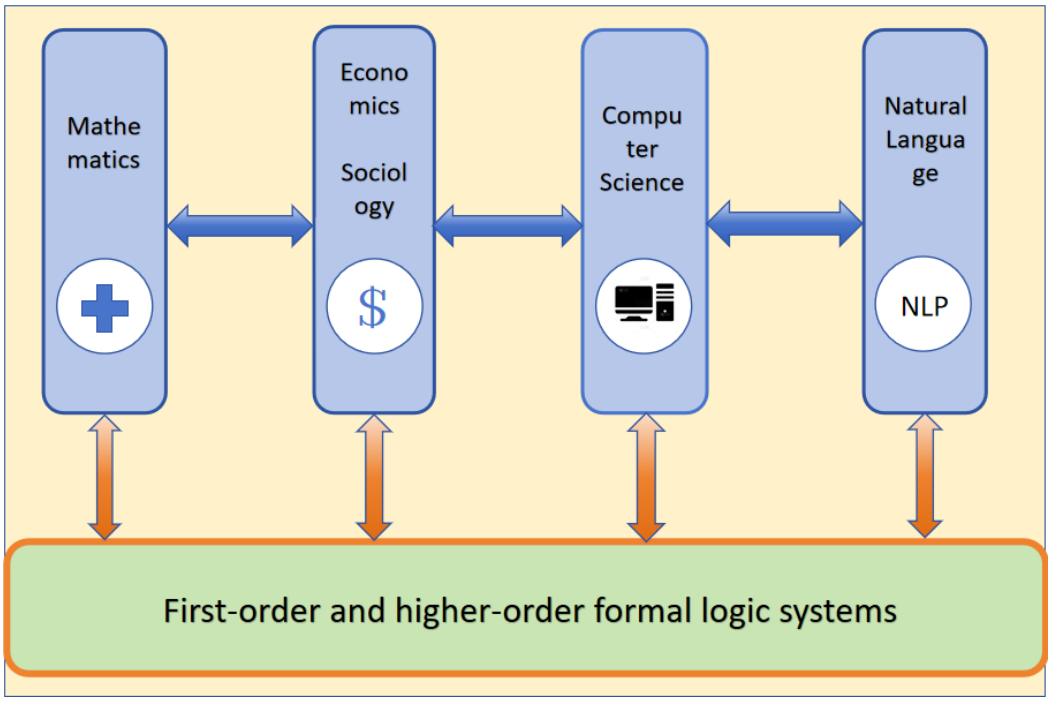

Figure 1. The logical system has become a bridge for communication between various fields and is the most universal language.

2. Formal Expression of State

The information postulate emphasizes that the state of an ontology can be mapped to the state of a carrier. Establishing a formal representation for both is therefore essential. Mathematical logic, grounded in axiomatic systems and symbolic language, enables rigorous and unambiguous characterization of concepts, propositions, and reasoning, thereby avoiding the ambiguity of natural language and providing a precise tool for scientific inquiry. We first restrict attention to a first-order object language.

2.1. First-Order Formal System Definition

First-order predicate logic language has become a core tool for formal modeling and automatic reasoning in fields such as mathematics, information science, and artificial intelligence due to its strong expressiveness, clear structure, rigorous reasoning, good computability, and strong versatility [8]. First, we define the most basic symbols in the first-order formal system:

Definition 1 (Symbols in $\mathcal{L}^{(1)}$)**.** *$\mathcal{L}^{(1)}$ contains the following symbols:*

- *First-order variables:* $x_1^{(1)}, x_2^{(1)}, \ldots$;
- *First-order constants:* $a_1^{(1)}, a_2^{(1)}, \ldots$;
- *First-order function symbols:* $f_1^{(1)1}, f_2^{(1)1}, \ldots, f_1^{(1)2}, f_2^{(1)2}, \ldots$;
- *brackets:* $(,)$;
- *First-order predicate symbols:* $A_1^{(1)1}, A_2^{(1)1}, \ldots, A_1^{(1)2}, A_2^{(1)2}, \ldots$;
- *Logical connectives:* $\sim$ *or* $\neg$ *(negation),*$\rightarrow$ *(implication);*
- *Quantifiers:* $\forall$ *(universal quantifier).*

As usual, we define $\vee, \wedge, \leftrightarrow$, and $\exists$ from these primitives. Symbols such as $=$ and $\in$ are treated as predicate symbols.

Terms in a language are similar to nouns or noun phrases in a natural language, but terms and nouns (phrases) are not exactly the same. The main difference is that terms contain variables and are "compound" items constructed using variables.

Definition 2 (Terms in $\mathcal{L}^{(1)}$)**.** *The terms in $\mathcal{L}^{(1)}$ are generated as follows:*

(1) *Variables and constants are terms.*

(2) *If $f_i^{(1)n}$ $(n > 0, i > 0)$ is a function symbol in $\mathcal{L}^{(1)}$ and $u_1, \ldots, u_n$ is a term in $\mathcal{L}^{(1)}$, then $f_i^{(1)n}(u_1, \ldots, u_n)$ is also a term in $\mathcal{L}^{(1)}$.*

Next, we define atomic formulas. Atomic formulas are the most basic formulas in the language.

Definition 3 (Atomic formulas in $\mathcal{L}^{(1)}$)**.** *If $A_i^{(1)n}$ $(n > 0, i > 0)$ is a predicate symbol in $\mathcal{L}^{(1)}$ and $u_1, \ldots, u_n$ is a term in $\mathcal{L}^{(1)}$, then $A_i^{(1)n}(u_1, \ldots, u_n)$ is an atomic formula in $\mathcal{L}^{(1)}$.*

Definition 4 (Well-formed formulas in $\mathcal{L}^{(1)}$)**.** *The well-formed formula in $\mathcal{L}^{(1)}$ is defined as follows:*

(1) *Each atomic formula is a well-formed formula in $\mathcal{L}^{(1)}$;*

(2) *If $\mathcal{A}$ and $\mathcal{B}$ are well-formed formulas in $\mathcal{L}^{(1)}$, then $\sim \mathcal{A}$ and $\mathcal{A} \rightarrow \mathcal{B}$ are both well-formed formulas in $\mathcal{L}^{(1)}$;*

(3) *If $\mathcal{A}$ is a well-formed formula in $\mathcal{L}^{(1)}$ and u is a variable or function symbol in $\mathcal{L}^{(1)}$, then $(\forall u)\mathcal{A}$ is a well-formed formula in $\mathcal{L}^{(1)}$.*

2.2. Recursive Definition of Higher-Order Formal Systems

Considering that first-order logic still has many shortcomings in terms of quantified objects, recursive induction, and other issues, such as its inability to fully express the concept of a set and its lack of direct characterization of higher-order properties, we need to expand the characterization capabilities of logical systems. Higher-order predicate logic (HOL) is an extension of first-order predicate logic. It allows quantification over predicates, functions, and even predicates about predicates. It offers greater expressive power, can formalize complex semantics in natural language and mathematics, and supports a richer set of logical tools and theoretical frameworks.

We obtain higher-order logic by iterating the construction above. Symbols, terms, atomic formulas, and wffs of order k are defined inductively from order $k-1$ (see Appendix A.1 for full details).

2.3. Interpretation of Formal Systems

Next, we can define the interpretation of the formal system.

Definition 5 (Interpretation of formal systems)**.** *An interpretation E of the formal system $\mathcal{L}$ is a two-tuple $E = \langle D_E, J \rangle$. Where:*

- *The domain D_E is a non-empty set that contains the value range of all elements in $\mathcal{L}$, including individuals, properties, relations, and functions.*
- *The interpretation function J is a mapping that maps symbols in $\mathcal{L}$ to concrete semantics in the domain D_E and is defined as follows:*
  - *Interpretation of constants and variables: Each constant $a_i^{(k)}$ is interpreted as an element in D_E, i.e., $J(a_i^{(k)}) \in D_E$; each variable $x_i^{(k)}$ is interpreted as an element in D_E, i.e., $J(x_i^{(k)}) \in D_E$.*
  - *Interpretation of function symbols: Each function symbol $f_i^{(k)n}$ is interpreted as a mapping from D_E^n to D_E, that is, $J(f_i^{(k)n}) : D_E^n \to D_E$.*
  - *Interpretation of predicate symbols: Each predicate symbol $A_i^{(k)n}$ is interpreted as a mapping from D_E^n to $\{True, False\}$, i.e., $J(A_i^{(k)n}) : D_E^n \to \{True, False\}$.*
  - *Interpretation of the terms:*

$$J(u) = \begin{cases} J(a_i^{(k)}), & \text{if } u \text{ is constant } a_i^{(k)} \\ J(x_i^{(k)}), & \text{if } u \text{ is variable } x_i^{(k)} \\ J(f_i^{(k)n})(J(u_1), \ldots, J(u_n)), & \text{if } u = f_i^{(k)n}(u_1, \ldots, u_n) \end{cases}$$

  - *Interpretation of atomic formula:*

$$if \mathcal{A} = A_i^{(k)n}(u_1, \ldots, u_n), then J(\mathcal{A}) = J(A_i^{(k)n})(J(u_1), \ldots, J(u_n)).$$

  - *Interpretation of logical connectives:*

$$J(\sim \mathcal{A}) = True \iff J(\mathcal{A}) = False$$

$$J(\mathcal{A} \to \mathcal{B}) = True \iff (J(\mathcal{A}) \to J(\mathcal{B})) = True$$

  - *Interpretation of quantifiers: If $\mathcal{A} = (\forall u)\mathcal{B}$, where u is a variable or function symbol, then*

$$J(\mathcal{A}) = True \iff \forall d \in D_E, when\ u\ is\ interpreted\ as\ d, J(\mathcal{B}) = True$$

The interpretation of formal systems transforms the abstract syntax of formal logic into concrete semantics, serving as a crucial link between the "symbolic world" and the "real world." It not only renders logical language meaningful but also provides a theoretical foundation for correct reasoning, modeling, verification, and automated applications. It is an essential concept in mathematical logic and information science.

2.4. Axiom System for Logical Expression of Ontology Components Under State Decomposition

In everyday usage, "state" is typically understood as something that pertains to a particular object, whose properties, meanings, and characteristics can be articulated and understood in natural language, that can be composed with other states to form new ones, and whose expression is inseparable from time and must respect temporal causality.

Let X denote a set of objects, T denote a set of time points or intervals, and L denote a higher-order formal system. In view of these properties of states, we propose the following four axioms [9]:

1. Parameter Reference Axiom: Every object $x \in X$, every moment or period $t \in T$, and every function $f \in F$ is represented by a unique constant or term c_x, c_t, c_f in L.

2. Property Expressibility Axiom: The properties, form, value, relationship, and other attributes of a set of objects in the entire domain can be expressed through functions and predicates in the formal system.
3. Logical Combination and the Closure Axiom:
   The generation rules of the state space $\mathcal{S}$ are limited to the following logical operations:
   - Implication:If $S_1, S_2 \in \mathcal{S}$, then it implies that $S_1 \to S_2$ is also a state of $\mathcal{S}$;
   - Negation:If $S \in \mathcal{S}$, then $\neg S \in \mathcal{S}$;
   - Quantification: if $S(x)$ is a state predicate, then $\forall x\, S(x)$ and $\exists x\, S(x)$ belong to S.

   Only finitely many applications of these operations may be used to generate new states.
4. Temporal Causality Axiom: When any attribute, relationship, or state is established at a certain moment, its change or evolution at subsequent moments can be described by the formula in L.

Theorem 1. *If for any $x \in X$, $t \in T$, there exists at least one attribute, relation, or property that can be expressed by L, and satisfies the axioms of parameter reference, attribute expressibility, logical combination closure, and temporal causality, then all states of any object x at any time t can be uniquely characterized by a set of well-formed formulas $\varphi_{S(x,t)}$ in L.*

Proof of Theorem 1. According to the parameter reference axiom, the object x, time t, and the function f involved in the set can all be represented by unique terms c_x, c_t, c_f in L.

Next, according to the property expressibility axiom, the various properties of the state set $S(x,t)$ can be expressed using functions and predicates in the formal system.

By definition, all expressions in the above state sets are atomic formulas in L. By logical combination and the closure axiom, any complex ontological state G can be recursively constructed from the base state by applying a finite number of generation rules expressible in L. Each generation rule is uniquely described by a well-formed formula and inference rule in L. Therefore, for any object x and time t, all its ontological states $S(x,t)$ can be uniquely mapped and characterized in L by the corresponding set of formulas $\varphi_{S(x,t)}$, where $\varphi_{S(x,t)}$ is recursively generated from the atomic formulas using logical rules.

In terms of uniqueness, the construction of $S(x,t)$ depends solely on x, t, and the set of ontological components. The representation of all predicates, functions, and parameters in L is uniquely determined by the axiomatic system. Therefore, $\varphi_{S(x,t)}$ uniquely corresponds to $S(x,t)$ within L. If $\varphi', \varphi'' \in L$ both characterize $S(x,t)$, then by the logical equivalence relation in L, $\varphi' \equiv \varphi_{S(x,t)} \equiv \varphi''$, guaranteeing uniqueness.

Furthermore, the temporal causality axiom states that any time t can be expressed by a recursive or evolutionary formula in L. Specifically, for any $x \in X$, there exists a formula $\psi(x,t',x,t)$ in L, such that $S(x,t')$ is uniquely determined by $S(x,t)$ and related laws. Thus, any dynamic evolution of a system can be recursively expressed by a chain of well-formed formulas in L, and the history and future of its state can be reduced to the logical deduction of a set of formulas.

In summary, $S(x,t)$ can always be rigorously characterized by a unique set of well-formulated formulas in L under interpretation, and this expression holds true for any dynamic evolution. The theorem is proved. □

2.5. The State of an Object at a Specific Time

Then, according to the theorem, we give the definition of state:

Definition 6 (state)**.** *The state $S(x,t)$ of a set of objects x at a particular time set t is an interpretation of a set of well-formed formulas in the formal system $\mathcal{L}$ on the universe $x \times t$. The specific*

properties of x and t, as well as the choice of formula set and the definition of the interpretation, are determined by the specific application scenario.

OIT holds that information is an enabling mapping from state to state, but it does not itself answer what a "state" is. By Definition 6, we clarify the concept of state and decompose it, making the expressions of "state" and "information" more fine-grained, precise, and concrete, thereby achieving a logical closure within OIT.

Theorem 1 and Definition 6 show that, under the axioms of parameter reference, property expressibility, logical combination and closure, and temporal causality, for every object x at time t, there exists a set of formulas $\varphi_S(x,t)$ that characterizes its state $S(x,t)$; moreover, this representation is unique up to logical equivalence. The formal representation of state is not only a technical tool but also a fundamental way for humans to understand and transform the world. It transforms intuitive concepts into precise mathematical objects, enabling rigorous reasoning and systematic analysis. As science and technology become increasingly complex, this formalization capability will continue to be a vital force driving the development of informatics and the progress of human civilization.

2.6. Relationship to and Distinctions from Existing Frameworks

We position our framework relative to established semantics as follows. Kripke semantics treats states as possible worlds with accessibility relations; in our setting, Kripke models arise as specific interpretations, but our focus is a uniform object–property–time semantics in first-/higher-order logic. LTL/CTL specifications embed as temporal fragments; dynamic logic modalities can be represented via interpretable predicates/functions; TLA+ is captured via state variables and a next relation; ASMs translate to predicate–function interpretations with update rules [10–14]. Our added value lies in a common semantic substrate and model-theoretic tools spanning finite, countable, and selected uncountable structures.

Next, we specify in detail the relationships and distinctions between this work and existing frameworks.

(1) Modal logic (Kripke structures): Kripke semantics treats states as possible worlds and transitions as accessibility relations, focusing on "reachability/necessity." In this paper, a state is defined as "the semantic object of formulas under an interpretation," and Kripke structures can, when needed, be embedded as a specific interpretation (worlds = elements of the domain; R as relations induced by predicates/functions). Our main thrust, however, is to use logical expressive power to unify the semantic construction of "object–property–time," rather than confining ourselves to the realm of accessibility. In other words, Kripke semantics is a specialized interpretation within our framework, while our framework natively supports higher-order properties, functions, and cross-domain mappings [15].

(2) Linear/branching-time logics (LTL/CTL): LTL/CTL excel at temporal specifications and model checking, targeting safety/liveness over path- or tree-shaped time structures [16]. This paper incorporates the temporal dimension but does not fix time solely as a linear or branching transition system; instead, it incorporates "time" into the domain and interpretation and allows first-order/higher-order predicates to describe intrinsic mathematical properties and cross-domain relations of structures. For engineering use, LTL/CTL specifications can be regarded as a temporal subset of our state language, while our framework provides broader object-level semantics and model-theoretic tools (e.g., types, Scott sentences, and isomorphism metrics).

(3) Dynamic logic (PDL, dynamic first-/higher-order logic): Dynamic logic takes program actions as modalities and is well-suited to characterize executable transformations. Our focus is the unified semantic definition and cross-domain representation

of "state," emphasizing that actions/processes are also treated as interpretable predicates/functions, thereby expressing object properties and evolution laws within a single language. By comparison, dynamic logic is strong in the calculational encapsulation of programmatic transformations, whereas this paper is strong in the semantic unification of cross-disciplinary objects and higher-order structures [17]. The two are complementary: embedding action semantics into our interpretive layer yields greater expressive power for complex object structures.

(4) TLA+: TLA+ centers on state variables and the next-step relation and is well-suited for proving safety and liveness in concurrent/distributed systems [18]. In our approach, TLA+ states and the Next relation can be viewed as instances of interpretations of specific predicates/functions, thereby bringing TLA+ specifications under a unified logical semantics that can interoperate with state models in mathematics, economics, or natural language. Our added value lies in providing model-theoretic tools spanning finite/countable to certain uncountable structures (e.g., Scott sentences and approximation limits), as well as a dual-sided ontology–carrier expression and measurement of "information mappings."

(5) Abstract State Machines (ASM): ASM describes system behavior using refined states and transition rules [19]. We can translate ASM state families and update rules into a unified predicate–function interpretation with corresponding inferential commitments. The difference is that ASM targets execution-level abstractions for engineering modeling, whereas this paper provides a cross-disciplinary repository of semantic isomorphism and expressibility theorems, enabling states from mathematics/social sciences/language to be aligned and compared with engineering specifications such as ASM/TLA+ on a common semantic foundation.

In sum, this paper is not competing with these frameworks but provides a more general semantic substrate: it treats first-order/higher-order (and, when necessary, infinitary) logic as a "universal language of state," viewing Kripke semantics, LTL/CTL, dynamic logic, TLA+, and ASM as subtheories or instances under specific signatures, semantics, and accessibility relations. When cross-domain problems require simultaneous treatment of higher-order properties, structural isomorphism, information mappings, and temporal evolution, our framework can integrate these methods within a single logical interpretation and model-theoretic toolbox, thereby enabling unified expression, comparison, and verification.

3. Mathematical Field State Expression

Mathematics, as a fundamental discipline, encompasses a wide range of branches and a vast system. Fundamental fields such as number theory focus on the properties of integers; algebra encompasses linear algebra (vector spaces and matrices), abstract algebra (groups, rings, and fields), and polynomial theory; and geometry includes Euclidean geometry and differential geometry (manifolds and curvature). Applied mathematics encompasses topology, probability theory and statistics (such as stochastic processes, Bayesian statistics, and the foundations of machine learning), and computational mathematics (numerical analysis, algorithm design, and scientific computing). Furthermore, new problems continue to emerge in discrete mathematics, mathematical physics, logic, and set theory.

From elementary arithmetic to cutting-edge research, mathematics demonstrates a progression in depth and abstraction, with numerous fields intersecting and integrating. Theorems, propositions, and formulas within each branch can be viewed as characterizing the "state" of certain mathematical objects. Broadly speaking, the state of mathematical objects is a core concept for understanding the dynamics, contextual dependence, and inherent connections of mathematical structures. Studying mathematical states not only

helps focus on key properties and ignore minor details, but also helps grasp the essence of a problem, forming a crucial foundation for the development of mathematical theory.

3.1. Formalization of Finite Mathematical Structures

Finite structures are the foundation of discrete mathematics. Problems such as finite sets and their subsets, and the connectivity, coloring, and matching of graphs in finite graph theory are all inseparable from the study of finite structures. In addition, many "infinite" mathematical concepts originate from the generalization of finite structures [20–22].

Finite mathematical structures are not only an essential component of mathematics but also fundamental tools for understanding the complex world, solving practical problems, and advancing science and technology. Here, we provide a rigorous proof that finite mathematical structures can be formalized in a first-order manner.

Definition 7 (Finite structure). *Let* $\mathfrak{A} = \langle A, R_1^{\mathfrak{A}}, \ldots, R_m^{\mathfrak{A}}, f_1^{\mathfrak{A}}, \ldots, f_n^{\mathfrak{A}}, c_1^{\mathfrak{A}}, \ldots, c_k^{\mathfrak{A}} \rangle$ *be a finite structure, where:* A *is a finite set,* $|A| = N$, $R_i^{\mathfrak{A}} \subseteq A^{a_i}$ *is an* a_i *ary-relation,* $f_j^{\mathfrak{A}} : A^{b_j} \to A$ *is a* b_j *ary-function, and* $c_l^{\mathfrak{A}} \in A$ *is a constant.*

Theorem 2 (First-order complete characterization of finite structures). *If* $\mathfrak{A}$ *is a finite structure, then there exists a first-order language* L *and a set of* L*-sentences* Γ *such that for any* L*-structure* $\mathfrak{B}$*:*

$$\mathfrak{B} \models \Gamma \text{ if and only if } \mathfrak{B} \cong \mathfrak{A} \tag{1}$$

Proof of Theorem 2. The definition L includes:

- Relation symbols: $R_1, \ldots, R_m$ (with arity $a_1, \ldots, a_m$ respectively)
- Function symbols: $f_1, \ldots, f_n$ (with arity $b_1, \ldots, b_n$ respectively)
- Constant symbols: $c_1, \ldots, c_k$
- Individual constants: $d_1, \ldots, d_N$ (corresponding to each element in A)

Let $A = \{\alpha_1, \alpha_2, \ldots, \alpha_N\}$; construct the following statement:

(Γ_1) domain restriction statement:

$$\forall x(x = d_1 \vee x = d_2 \vee \cdots \vee x = d_N) \tag{2}$$

(Γ_2) element-wise distinction statement:

$$d_i \neq d_j \quad (\text{for all } 1 \leq i < j \leq N) \tag{3}$$

(Γ_3) relation characterization statement:
For each relation symbol R_i and each tuple $(\alpha_{j_1}, \ldots, \alpha_{j_{a_i}}) \in A^{a_i}$:

$$\begin{cases} R_i(d_{j_1}, \ldots, d_{j_{a_i}}) & \text{If } (\alpha_{j_1}, \ldots, \alpha_{j_{a_i}}) \in R_i^{\mathfrak{A}} \\ \neg R_i(d_{j_1}, \ldots, d_{j_{a_i}}) & \text{If } (\alpha_{j_1}, \ldots, \alpha_{j_{a_i}}) \notin R_i^{\mathfrak{A}} \end{cases} \tag{4}$$

(Γ_4) function characterization statement:
For each function symbol f_j and each tuple $(\alpha_{k_1}, \ldots, \alpha_{k_{b_j}}) \in A^{b_j}$:

$$f_j(d_{k_1}, \ldots, d_{k_{b_j}}) = d_l \tag{5}$$

where l satisfies $f_j^{\mathfrak{A}}(\alpha_{k_1}, \ldots, \alpha_{k_{b_j}}) = \alpha_l$

(Γ_5) Constant Characterization Statement:

$$c_i = d_j \quad \text{where } j \text{ satisfies } c_i^{\mathfrak{A}} = \alpha_j \tag{6}$$

We define $\Gamma = \{\Gamma_1, \Gamma_2, \Gamma_3, \Gamma_4, \Gamma_5\}$. Next, we prove two lemmas.

Lemma 1. *If* $\mathfrak{B} \models \Gamma$, *then* $|B| = N$.

Proof of Lemma 1. By (Γ_1), $\forall x \in B, \exists i \in \{1, \ldots, N\}, x = d_i^{\mathfrak{B}}$, so $|B| \leq N$. By (Γ_2), $d_i^{\mathfrak{B}} \neq d_j^{\mathfrak{B}}$ for all $i \neq j$, so $|B| \geq N$. Therefore, $|B| = N$. □

Lemma 2. *If* $\mathfrak{B} \models \Gamma$, *then the map* $h : A \to B$ *defined as* $h(\alpha_i) = d_i^{\mathfrak{B}}$ *is a bijection.*

Proof of Lemma 2. This follows directly from Lemma 1 and (Γ_2). □

Next, we can prove the consequence of Theorem 2:
(⇒) If $\mathfrak{B} \cong \mathfrak{A}$, then $\mathfrak{B} \models \Gamma$:
Let $g : A \to B$ be an isomorphic mapping. Definition of $\mathfrak{B}$:

- $d_i^{\mathfrak{B}} = g(\alpha_i)$
- $R_i^{\mathfrak{B}}, f_j^{\mathfrak{B}}, c_l^{\mathfrak{B}}$ are defined by isomorphic correspondences.

By the definition of isomorphism, $\mathfrak{B}$ satisfies all statements in Γ.
(⇐) If $\mathfrak{B} \models \Gamma$, then $\mathfrak{B} \cong \mathfrak{A}$:
By Lemma 2, $h : A \to B$ is defined as $h(\alpha_i) = d_i^{\mathfrak{B}}$, which is a bijection.
Verify that h maintains the relationship:
For any $(\alpha_{j_1}, \ldots, \alpha_{j_{a_i}}) \in A^{a_i}$:

$$\begin{aligned} & (\alpha_{j_1}, \ldots, \alpha_{j_{a_i}}) \in R_i^{\mathfrak{A}} \\ \iff & \mathfrak{B} \models R_i(d_{j_1}, \ldots, d_{j_{a_i}}) \quad \text{(by } (\Gamma_3)\text{)} \\ \iff & (d_{j_1}^{\mathfrak{B}}, \ldots, d_{j_{a_i}}^{\mathfrak{B}}) \in R_i^{\mathfrak{B}} \\ \iff & (h(\alpha_{j_1}), \ldots, h(\alpha_{j_{a_i}})) \in R_i^{\mathfrak{B}} \end{aligned} \tag{7}$$

Verify that h holds:
For any $(\alpha_{k_1}, \ldots, \alpha_{k_{b_j}}) \in A^{b_j}$, let $f_j^{\mathfrak{A}}(\alpha_{k_1}, \ldots, \alpha_{k_{b_j}}) = \alpha_l$
From (Γ_4):

$$\begin{aligned} & \mathfrak{B} \models f_j(d_{k_1}, \ldots, d_{k_{b_j}}) = d_l \\ \iff & f_j^{\mathfrak{B}}(d_{k_1}^{\mathfrak{B}}, \ldots, d_{k_{b_j}}^{\mathfrak{B}}) = d_l^{\mathfrak{B}} \\ \iff & f_j^{\mathfrak{B}}(h(\alpha_{k_1}), \ldots, h(\alpha_{k_{b_j}})) = h(\alpha_l) \\ \iff & f_j^{\mathfrak{B}}(h(\alpha_{k_1}), \ldots, h(\alpha_{k_{b_j}})) = h(f_j^{\mathfrak{A}}(\alpha_{k_1}, \ldots, \alpha_{k_{b_j}})) \end{aligned} \tag{8}$$

Verify that h remains constant:this is directly derived from (Γ_5).
Thus, h is an isomorphism, $\mathfrak{B} \cong \mathfrak{A}$. □

We now derive the following corollaries.

Corollary 1 (Uniqueness)**.** *The set of axioms* Γ *uniquely determines* $\mathfrak{A}$ *in the sense of logical equivalence.*

Corollary 2 (Completeness)**.** *For any first-order property* φ *on* $\mathfrak{A}$, *either* $\Gamma \models \varphi$ *or* $\Gamma \models \neg\varphi$.

Proof of Corollary 2. Let φ be any first-order sentence. Since φ is a sentence, either $\mathfrak{A} \models \varphi$ or $\mathfrak{A} \models \neg\varphi$ must hold.

If $\mathfrak{A} \models \varphi$, then, by Theorem 1, any structure $\mathfrak{B}$ satisfying Γ is isomorphic to $\mathfrak{A}$, so $\mathfrak{B} \models \varphi$. Therefore, $\Gamma \models \varphi$.

If $\mathfrak{A} \models \neg\varphi$, then similarly, $\Gamma \models \neg\varphi$. □

Corollary 3 (Decidability)**.** *The set* $\{\varphi : \Gamma \models \varphi\}$ *is decidable.*

Proof of Corollary 3. By Corollary 2, for any sentence φ, we can directly verify that $\mathfrak{A} \models \varphi$ on the finite structure $\mathfrak{A}$. If true, then $\Gamma \models \varphi$; otherwise, $\Gamma \models \neg\varphi$. □

3.2. Previous Research on the Formalization of Infinite Structures

Naturally, we will wonder whether or not all infinite structures, except finite ones, can be completely characterized by a set of first-order logic formulas.

Generally speaking, the answer is no. In fact, according to the research results of Skolem et al., even countable structures cannot be guaranteed to be fully described by first-order logic [23].

Theorem 3 (Skolem)**.** *The standard natural numbers are countable structures that cannot be characterized by first-order categoricity.*

We omit the proof. The main idea of the proof is to introduce infinite elements through extension theory. Then, we use the compactness theorem to derive a non-standard model and conclude.

Of course, if we expand the tools from first-order logic to higher-order logic, we can expand the characterization capabilities of the logical language [24,25]:

Theorem 4 (Peano)**.** *Under standard second-order semantics, the second-order Peano axioms categorically characterize the structure of natural numbers. That is, if quantification over set variables is allowed, then the sequence of natural numbers can be uniquely characterized by second-order logic.*

This result highlights the greater expressive power of second-order logic under standard semantics. It not only solves the problem of characterizing natural numbers but also reveals a fundamental property of the expressive power of logical systems—higher-order logic possesses greater expressiveness than first-order logic.

However, despite its greater expressiveness, higher-order logic still cannot represent all countable structures. The boundaries of the logical structures that higher-order logic can represent remain unresolved. This result reflects the fundamental tension between computability and logical expressiveness. Even the most powerful logical systems cannot fully "tame" the complexity of infinite structures. This perhaps reveals a certain irreducible complexity of mathematical reality.

At present, the problem of expressing mathematical structures still depends on the work done by Scott in 1965 [26].

Scott first introduced the concept of infinite logic:

Definition 8 (Infinitary logic $L_{\omega_1\omega}$)**.** *The language* $L_{\omega_1\omega}$ *is defined by the following rules:*

1. *Contains all atomic formulas of first-order logic.*
2. *If* $\{\phi_i : i \in I\}$ *is a set of formulas and* $|I| \leq \aleph_0$*, then* $\bigwedge_{i\in I} \phi_i$ *and* $\bigvee_{i\in I} \phi_i$ *are also formulas.*
3. *If* ϕ *is a formula and* x *is a variable, then* $\exists x\phi$ *and* $\forall x\phi$ *are formulas.*
4. *Every formula contains only a finite number of free variables.*

Furthermore, he proposed the crucial isomorphism theorem in the article.

Theorem 5 (Scott's isomorphism theorem, 1965)**.** *Let $\mathfrak{A}$ be a countable structure and $\mathcal{L}$ be a countable language. Then, there exists a $L_{\omega_1\omega}$ sentence $\phi_{\mathfrak{A}}$ (called a* ***Scott sentence*** *of $\mathfrak{A}$), such that:*

For any structure $\mathfrak{B}$,

$$\mathfrak{B} \models \phi_{\mathfrak{A}} \iff \mathfrak{B} \cong \mathfrak{A} \tag{9}$$

That is, $\phi_{\mathfrak{A}}$ completely characterizes the structure $\mathfrak{A}$ in an isomorphic sense.

Scott's isomorphism theorem is more than just a technical result; it reveals that infinitely long formulas are a natural tool for dealing with infinite structures, and that abstract existence can be transformed into concrete constructions.

Scott's isomorphism theorem not only solves a specific mathematical problem but also, more importantly, opens up a whole new research paradigm, influencing multiple branches of mathematics and still guiding development in related fields today. This makes it one of the most important achievements in mathematical logic of the 20th century.

3.3. Formalization of Conditional Infinite Structures

To address the problem that infinite structures are difficult to characterize using logic, we present and prove a slightly weaker but still highly universal theorem. First, we provide several definitions.

Definition 9 (Relationship maintenance)**.** *Let $\mathfrak{M} = (M, R_1, R_2, \ldots, R_k)$ be a structure where R_i is a n_i-ary relation. Let $\{\mathfrak{M}_j = (M_j, R_1^j, R_2^j, \ldots, R_k^j)\}_{j\in\mathbb{N}}$ be an approximate sequence.*

Relationship maintenance means:

$$\forall i \in \{1, 2, \ldots, k\}, \forall j \in \mathbb{N}: \quad R_i^j = R_i \upharpoonright M_j^{n_i} \tag{10}$$

then

$$R_i^j = \{(a_1, \ldots, a_{n_i}) \in M_j^{n_i} : (a_1, \ldots, a_{n_i}) \in R_i\} \tag{11}$$

Definition 10 (A precise definition of recursive approximation)**.** *The structure $\mathfrak{M} = (M, R_1, \ldots, R_k)$ satisfies recursive approximation if and only if:*

There exists a sequence $\{\mathfrak{M}_n\}_{n\in\mathbb{N}}$ where every $\mathfrak{M}_n = (M_n, R_1^n, \ldots, R_k^n)$ satisfies:

1. *Monotonicity: $\mathfrak{M}_n \subseteq \mathfrak{M}_{n+1} \subseteq \mathfrak{M}$*
2. *Countability: $|M_n| = \aleph_0$ for all n*
3. *Recursion: There exists a recursive function that computes the Scott sentence for each $\mathfrak{M}_n$*
4. *Density: $\overline{\bigcup_n M_n} = M$(in appropriate topology)*
5. *Relationship maintenance: $R_i^n = R_i \upharpoonright M_n^{ar(R_i)}$ for all i, n, where $ar(R_i)$ denotes the number of elements of the relation R_i*
6. *Asymptotic uniqueness: Any two sequences are isomorphic to themselves or to each other after adding a finite number of elements from M.*

Definition 11 (Topology of Scott's sentence space)**.** *We define a topology on the space of Scott sentences, which makes the convergence exact. We assume that the space is complete under this topology.*

Define $\mathcal{S}$ as the space of all Scott sentences. For $\phi, \psi \in \mathcal{S}$, define the distance:

$$d(\phi, \psi) = \sum_{k=1}^{\infty} 2^{-k} \cdot d_k(\phi, \psi) \tag{12}$$

where:

$$d_k(\phi, \psi) = \frac{|Tp_k(\phi) \triangle Tp_k(\psi)|}{|Tp_k(\phi) \cup Tp_k(\psi)|} \tag{13}$$

Here, $Tp_k(\phi)$ denotes the set of k-types that occur in a structure satisfying ϕ, and $Tp_k(\mathfrak{M})$ contains the "complete description" of all possible k-tuples in the structure $\mathfrak{M}$.

Definition 12 (Local finiteness)**.**

$$\forall k, \forall l \in \mathbb{N}, |Tp_k(\phi_l)| = \aleph_0 \tag{14}$$

and

$$\forall k, \forall l \in \mathbb{N}, |Tp_k(\phi_{l+1}) - Tp_k(\phi_l)| < \infty \tag{15}$$

Theorem 6 (Higher-order characterization theorem for recursive approximate structures)**.** *Suppose $\mathfrak{M}$ is an uncountable structure that satisfies recursive approximation and local finiteness. Then, there exists a higher-order theory $T_{\mathfrak{M}}$ such that, for every structure $\mathfrak{N}$,*

$$\forall \mathfrak{N}(\mathfrak{N} \models T_{\mathfrak{M}} \iff \mathfrak{N} \cong \mathfrak{M}) \tag{16}$$

Here, we assume that higher-order logic can be infinitely quantified, that is, it satisfies the properties of infinite logic. We prove this conclusion step by step. First, we prove that the recursive approximation sequence is inherently unique.

Lemma 3 (Normality of approximate sequences)**.** *Assume $\mathfrak{M}$ satisfies recursive approximation, and $\{\mathfrak{M}_n\}$ and $\{\mathfrak{M}'_n\}$ are two approximate sequences that satisfy the condition. Then, there exists an increasing function $h : \mathbb{N} \to \mathbb{N}$, such that:*

$$\mathfrak{M}_n \cong \mathfrak{M}'_{h(n)} \text{ for all } n \tag{17}$$

Proof of Lemma 3. By density, monotonicity, and asymptotic uniqueness, for any $\mathfrak{M}_n$, there exists a sufficiently large m, such that $\mathfrak{M}_n$ can be embedded in $\mathfrak{M}'_m$.

Vice versa. Combined with countability, we obtain an isomorphism. □

Next, we define limit operations and related concepts.

Definition 13 (Limits of structural sequences)**.** *Let $\{\mathfrak{M}_n\}$ be an increasing countable sequence of structures. Definition:*

$$\lim_{n\to\infty} \mathfrak{M}_n = \left(\bigcup_n M_n, \bigcup_n R_1^n, \ldots, \bigcup_n R_k^n \right) \tag{18}$$

If the union of every relation is well-defined in the limit.

Lemma 4 (Existence and uniqueness of limits)**.** *If $\{\mathfrak{M}_n\}$ satisfies the conditions for recursive approximation, then the limit exists and is isomorphic to the original structure $\mathfrak{M}$.*

Proof of Lemma 4. Existence: By monotonicity, $\bigcup_n M_n$ is well-defined. The union of relations is well-defined by the relation-preserving property.

Uniqueness: By density, $\bigcup_n M_n$ is dense in M. If $\mathfrak{M}$ has appropriate continuity (implied by the recursive approximation), then it is uniquely determined by the dense substructure. □

Next, we need to verify the convergence of Scott's sequence of sentence.

Theorem 7 (Convergence Theorem)**.** *Under the recursive approximation, if the local finiteness condition is additionally satisfied, then the Scott sentence sequence $\{\phi_n\}$ converges to ϕ_∞ in the defined topology.*

Proof of Theorem 7. For fixed k, consider the sequence:

$$\mathrm{Tp}_k(\phi_1) \subseteq \mathrm{Tp}_k(\phi_2) \subseteq \cdots \subseteq \mathrm{Tp}_k(\phi_{k_m}) \tag{19}$$

For any ϵ, we can choose K so that$\sum_{k=K+1}^{\infty} 2^{-k} < \varepsilon/2$.

Since each inclusion is a subset relation, we can exploit local finiteness. Let N_k exist, such that when $m, n \geq N_k$:

$$d_k(\phi_m, \phi_n) < 2^{(k-1)}\epsilon/K \tag{20}$$

So, there exists $N = \max\{N_1, N_2, \ldots, N_K\}$, such that when $m, n \geq N$:

$$d_k(\varphi_n, \varphi_{)} < 2^{(k-1)}\epsilon/N \quad \text{for all } k \leq K \tag{21}$$

Thus:

$$\begin{aligned} d(\varphi_n, \varphi_m) &= \sum_{k=1}^{\infty} 2^{-k} \cdot d_k(\varphi_n, \varphi_m) \\ &= \sum_{k=1}^{K} 2^{-k} \cdot 2^{(k-1)}\varepsilon/K + \sum_{k=K+1}^{\infty} 2^{-k} \cdot d_k(\varphi_n, \varphi_\infty) \\ &\leq \varepsilon/2 + \sum_{k=K+1}^{\infty} 2^{-k} \cdot 1 \\ &< \varepsilon/2 + \varepsilon/2 = \varepsilon \end{aligned} \tag{22}$$

Thus, we obtain a Cauchy sequence. Based on completeness, we prove that this sequence has a limit. Let us assume that its limit is ϕ_∞. □

Next, we can construct a complete characterization theory:

$$T_{\mathfrak{M}} = T_{\text{approximation}} \cup T_{\text{limit}} \cup T_{\text{unique}} \cup \phi_\infty \tag{23}$$

$$T_{\text{approximation}} = \{\exists\{\mathfrak{M}_n\}_n.\text{Satisfies the recursive approximation condition}\} \tag{24}$$

$$T_{\text{limit}} = \left\{\mathfrak{M} = \lim_{n\to\infty} \mathfrak{M}_n\right\} \tag{25}$$

$$T_{\text{unique}} = \{\text{The approximation sequence is unique under isomorphism}\} \tag{26}$$

Finally, we prove Theorem 6;

Proof of Theorem 6. Assume $\mathfrak{N} \models T_{\mathfrak{M}}$, then $\mathfrak{N}$ has an approximate sequence $\{\mathfrak{N}_n\}$ that satisfies the same conditions.

By $\phi_1, \ldots, \phi_\infty$, the Scott sentence for each $\mathfrak{N}_n$ is identical to the corresponding $\mathfrak{M}_n$. By Scott's theorem, $\mathfrak{N}_n \cong \mathfrak{M}_n$ for all n.

Construct an isomorphic sequence $f_n : \mathfrak{M}_n \to \mathfrak{N}_n$. By monotonicity, consistency, and Lemma 4, $\{f_n\}$ can be combined into a global isomorphism:

$$f = \bigcup_n f_n : \mathfrak{M} \to \mathfrak{N} \tag{27}$$

Verifying that f is indeed an isomorphism: Injectivity, by the injectivity and density of each f_n; Surjectivity, by the surjectivity and limit properties of each f_n; Homomorphism, by the preservation of relations. □

At this point, we have rigorously proved the theorem and obtained the most abstract formula for $\mathfrak{M}$, ϕ_∞. Clarifying ϕ_∞ will help researchers understand the logical nature of infinite structures and facilitate deeper research.

3.4. Formalization of Phenomena in Mathematics

As shown in Figure 2, based on the formal characterization of finite and infinite mathematical structures, we finally conclude the following:

Theorem 8. *The following classes of structures are logically characterizable.*

1. *Finite structures. There exists a first-order language L and a set of sentences Γ such that for any L-structure $\mathfrak{B}$,*
$$\mathfrak{B} \models \Gamma \iff \mathfrak{B} \cong \mathfrak{A}$$
(a complete characterization up to isomorphism; see Theorem 2).
2. *Countable structures. There exists a Scott sentence $\phi(\mathfrak{A})$ in $L_{\omega_1,\omega}$, such that*
$$\mathfrak{B} \models \phi(\mathfrak{A}) \iff \mathfrak{B} \cong \mathfrak{A}$$
(a complete characterization up to isomorphism; see Theorem 5).
3. *Uncountable structures satisfying "recursive approximation + local finiteness". There exists a higher-order theory $T_{\mathfrak{M}}$ (allowing countably infinite conjunctions), such that*
$$\mathfrak{N} \models T_{\mathfrak{M}} \iff \mathfrak{N} \cong \mathfrak{M}$$
(see Theorems 6 and 7).

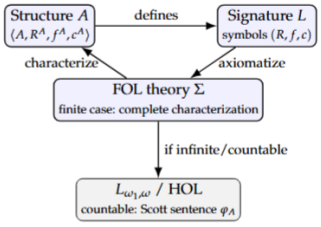

Figure 2. Mathematical structures: from a concrete structure A and signature L to FOL axioms Σ (finite case) and to a Scott sentence φ_A or a higher-order theory (infinite case).

Our theory demonstrates that, within the appropriate framework, nearly all mathematical phenomena—particularly discrete, algebraic, and finitely generated phenomena—can be meaningfully logically characterized. This is an important theoretical achievement that expands our understanding of the extent to which mathematics can be formalized.

However, the richness and complexity of mathematics mean that finding a precise logical expression is difficult. As mathematics develops, modern mathematics presents new challenges. Problems such as the explosion of parameter space, the breakdown of intuition, and insufficient tools make characterizing high-dimensional and abstract problems particularly difficult. This reminds us that mathematics has both a formal side and a side beyond formalism. A perfect logical characterization may be a guiding principle, guiding us to continuously deepen our understanding, but it should not be mistaken for a fully achievable ultimate goal.

4. State Expression in Economics and Sociology

This section treats economics and sociology in parallel and begins with finite structures. Logical formalization in economics reduces ambiguity, enforces precise assumptions, and scales to multi-agent, multi-constraint settings. It also provides a common language to compare and integrate schools of thought. We begin with finite economic structures.

4.1. Logical Characterization in the Field of Economics

The logical characterization of economics is of fundamental significance to the development of the discipline and is also one of the hot issues in research [27–29]. First, logical representation can eliminate ambiguity and vagueness in economic theory. Traditional textual descriptions often allow for multiple interpretations, while logical formalization requires precise definitions of each concept and relationship, forcing theorists to clearly express their assumptions and reasoning. For example, when we say "demand is negatively correlated with price," a logical representation requires us to specify under what conditions, for which goods, and over what timeframe this relationship holds true.

Furthermore, economics deals with complex systems involving multiple agents, multiple levels, and multiple variables, involving interactions between diverse actors such as consumers, businesses, and governments [30]. When a theory becomes complex, natural language descriptions often fail to accurately capture all logical relationships and constraints. Logical representations provide a structured approach to organizing these complex relationships, ensuring the internal consistency and integrity of the theory [31]. For example, when analyzing market equilibrium, we need to simultaneously consider multiple constraints, such as the supply equation, the demand equation, and market-clearing conditions. Logical representations can clearly demonstrate the logical dependencies between these conditions.

At the same time, considering that there are multiple schools and theoretical frameworks within economics, such as neoclassical economics, Keynesianism, institutional economics, etc. [32], different theoretical frameworks utilize different conceptual systems and analytical methods, making academic dialogue difficult. Logical representation provides a unified language for different theories, enabling theoretical comparison, integration, and synthesis. Researchers can more easily identify commonalities and divergences between different theories, promoting the integration and development of theories.

Therefore, the importance of logical characterization in the field of economics is self-evident. Here we first consider the logical characterization of finite economic structures.

Definition 14 (Economic structure)**.** *An economic structure* $\mathfrak{S}$ *can be represented as follows:*

$$\mathfrak{S} = (A, R_1, R_2, \ldots, F_1, F_2, \ldots, P_1, P_2, \ldots) \tag{28}$$

where:

- A *is a set of agents (individuals, enterprises, institutions, etc.)*
- R_i *is a relationship (social network, hierarchy, transaction relationship, etc.)*
- F_j *is a function (utility function, production function, decision rule, etc.)*
- P_k *is a process (market mechanism, institutional evolution, information dissemination, etc.)*

Definition 15 (The logical language of economic structure)**.** *The basic language* $\mathcal{L}_{SE}$ *Contains:*
Individual constants:

- $a_1, a_2, \ldots$ *represent specific agent individuals, enterprises, organizations, etc.*

Variables:

- x, y, z *denote agent variables,* t *denotes time variables, and* s *denotes state variables.*

Predicate symbols:

- $Agent(x)$*:* x *is an agent.*
- $Transition_{P_k}(s, s')$*: represents the transition from state* s *to state* s' *under process* P_k*.*
- $TransitionCondition_k(s, s')$*: indicates that the transition from state* s *to state* s' *under process* P_k *satisfies the prescribed condition.*

Theorem 9 (First-order representability of finite economic structures)**.** *Let* $\mathfrak{S} = (A, R_1, R_2, \ldots, F_1, F_2, \ldots, P_1, P_2, \ldots)$ *be a finite economic structure, that is:*

1. $|A| < \infty$ *(Finite Agents)*
2. *Every relation* R_i *and function* F_j *is defined over a finite domain and has corresponding predicate and function representations in the base language.*
3. *The process* P_k *involves finite states and finite time.*

Then there exists a set of first-order formulas Φ*, such that:*

$$\forall \mathfrak{T}, \mathfrak{T} \models \Phi \iff \mathfrak{T} \cong \mathfrak{S} \tag{29}$$

Proof of Theorem 9. Domain characterization:

$$\phi_{\text{domain}} = \forall x (Agent(x) \to \bigvee_{i=1}^{|A|} x = a_i) \wedge \bigwedge_{i \neq j} a_i \neq a_j \tag{30}$$

Characterization of relationships: For each relation $R_i \subseteq A^{n_i}$:

$$\phi_{R_i} = \forall x_1 \cdots x_{n_i} (R_i(x_1, \ldots, x_{n_i}) \leftrightarrow \bigvee_{(a_{j_1}, \ldots, a_{j_{n_i}}) \in R_i} (x_1 = a_{j_1} \wedge \cdots \wedge x_{n_i} = a_{j_{n_i}})) \tag{31}$$

Function description: For each function $F_j : A^{m_j} \to D_j$ (where the range D_j is finite):

$$\phi_{F_j} = \forall x_1 \cdots x_{m_j} \exists! y (F_j(x_1, \ldots, x_{m_j}) = y \wedge y \in D_j) \tag{32}$$

Description of the process: For each process P_k, if it involves a finite state transition:

$$\phi_{P_k} = \forall s \forall s' (Transition_{P_k}(s, s') \to \text{TransitionCondition}_k(s, s')) \tag{33}$$

Complete formula:

$$\Phi = \{\phi_{\text{domain}}, \phi_{R_1}, \phi_{R_2}, \ldots, \phi_{F_1}, \phi_{F_2}, \ldots, \phi_{P_1}, \phi_{P_2}, \ldots\} \tag{34}$$

Since all components are finite, every formula is first-order, and Φ completely characterizes the structure $\mathfrak{S}$, we have proved the result. □

From the proof, we can see that structural finiteness has a profound impact on the construction of economic theory. It means that economic models need to pay more attention to boundary conditions, constrained optimization, and finite games. At the same time, finiteness also provides a more realistic foundation for economic analysis, making theoretical predictions more closely aligned with actual economic phenomena.

Theorem 10. *In the framework of first-order and higher-order logical theories, if the domains of the relations/functions involved in an economic structure, as well as the time/state sets, are all finite, then the structure can be completely characterized.*

Given that empirical applications often fix finite sets of agents, goods, periods, and constraints, many practical economic models admit complete first-order specifications for a fixed instance.

Here, we give an example to verify how logic represents economic phenomena and structures. Lowercase letters represent variables.Table 1 gives the definition of predicates

Table 1. Predicate definition.

Symbol	Definition
$Demand(i, g, p, q)$	The quantity q demanded by consumer i for good g at price p is q
$Consumer(i)$	i is a consumer
$Good(g)$	g is a commodity
$Price(p)$	p is a valid price (non-negative)
$Quantity(q)$	q is a valid quantity (non-negative)
$MaximizesUtility(i, b, constraint)$	Consumer i chooses a bundle of goods b to maximize utility under constraints
$BudgetConstraint(i, p, income)$	Consumer i's budget constraint under price p and income $income$
$Income(i)$	Income of consumer i
$Contains(b, g, q)$	The bundle b contains a quantity q of the good g
$Supply(f, g, p, q)$	The supply of good g by firm f at price p is q
$Firm(f)$	f is an enterprise (production unit)
$ProfitMaximizing(f, v, q_{bundle}, p, w)$	Firm f chooses input v and output q_{bundle} to maximize profit under price p and factor price w

Definition 16. ***Definition of Supply and Demand:***

$$\begin{aligned} Demand(i, g, p, q) \leftrightarrow\ & Consumer(i) \wedge Good(g) \wedge Price(p) \wedge Quantity(q) \wedge \\ & \exists b(MaximizesUtility(i, b, BudgetConstraint(i, p, Income(i))) \wedge \\ & Contains(b, g, q)) \end{aligned} \tag{35}$$

$$\begin{aligned} Supply(f, g, p, q) \leftrightarrow\ & Firm(f) \wedge Good(g) \wedge Price(p) \wedge Quantity(q) \wedge \\ & \exists v, q_{bundle}(ProfitMaximizing(f, v, q_{bundle}, p, w) \wedge \\ & Contains(q_{bundle}, g, q)) \end{aligned} \tag{36}$$

In Appendix A.3, we present another example related to economics. Here, we outline its basic setup in Figure 3.

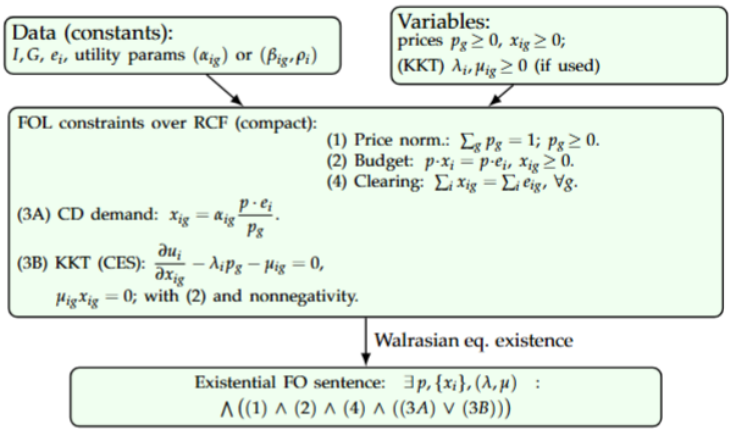

Figure 3. Finite Arrow–Debreu exchange economy: constants (data) + variables + FOL constraints over real closed fields yield an existential sentence for equilibrium existence.

4.2. Logical Characterization in the Field of Sociology

The reason why social structures require logical representations is closely related to their inherent complexity and abstractness, and these needs are even more pressing than in economics [33].

Logical formalization in sociology reduces ambiguity, enforces precise assumptions, and scales to settings with many actors, relations, and constraints. Social phenomena typically involve multi-relational networks (e.g., kinship, power, exchange, culture), heterogeneous attributes, and context-dependent mechanisms. Natural-language descriptions often under-specify transitivity, symmetry, hierarchy, or diffusion effects; a logical representation makes such properties explicit and checkable. Just as in economics, a common logical language also facilitates comparison and integration across theoretical traditions (e.g., structuralism, network analysis, institutional theory), enabling cumulative, interoperable models.

We begin with finite social structures and show that, when agents, relations, and time/state sets are finite, they admit complete first-order specifications for a fixed instance. We then illustrate how typical sociological claims—such as influence through friendship ties, formation of network closures, or role/attribute constraints—translate into predicates and sentences that support reasoning, measurement, and verification [34].

The first-order logic description of social structure is similar to that of economic structure. Here, we only give the theorem.

Theorem 11 (First-order representability of finite social structures)**.** *Let* $\mathfrak{S} = (A, R_1, R_2, \ldots, F_1, F_2, \ldots, P_1, P_2, \ldots)$ *be a finite social structure, that is:*

1. $|A| < \infty$ *(finite agent)*
2. *Each relation* R_i *and function* F_j *is defined over a finite domain and has corresponding predicate and function representations in the base language.*
3. *The process* P_k *involves finite states and finite time.*

Then, there exists a set of first-order formulas Φ *such that:*

$$\forall \mathfrak{T}, \mathfrak{T} \models \Phi \iff \mathfrak{T} \cong \mathfrak{S} \tag{37}$$

Theorem 12. *In the framework of first-order and higher-order logical theories, if the domains of the relations/functions involved in an social structure, as well as the time/state sets, are all finite, then the structure can be completely characterized.*

Similarly, considering the limitations of social structure, we can draw a conclusion. Within the theoretical framework of first-order and higher-order logic, almost all social structures can be fully characterized. Through logical representation, sociology can not only describe social phenomena more accurately but also discover hidden social laws, providing more powerful theoretical tools for understanding and improving social structure.

Here, we give an example of logical representation. Table 2 gives the definition of predicates.

Table 2. Semantic interpretation table of sociological predicates.

Predicate	Semantic Meaning
$Friend(x, y)$	x and y are friends
$Smokes(x)$	x smokes
$Influenced(x, y)$	x is affected by y
$HigherSmokingProbability(x)$	x has a higher probability of smoking
$SocialNetwork(x, y)$	x and y are in the same social network

Through the given predicate, we can express the state of the smoking phenomenon in sociology. Here, HigherSmokingProbability(x) is abbreviated as HSP(x).

Definition 17 (Expressions related to smoking)**.**

$$\phi_1 = \forall x, y(Friend(x, y) \rightarrow Friend(y, x)) \tag{38}$$

$$\phi_2 = \forall x, y(Friend(x, y) \wedge Smokes(y) \rightarrow Influenced(x, y)) \tag{39}$$

$$\phi_3 = \forall x, y(Influenced(x, y) \rightarrow HSP(x)) \tag{40}$$

$$\phi_4 = \forall x, y, z(Friend(x, y) \wedge Friend(y, z) \rightarrow SocialNetwork(x, z)) \tag{41}$$

To better demonstrate how our formalization can be applied to sociological practice—and to answer how a rigorous definition of state informs the choice of the eleven metrics in OIT—we present another example related to a census. Table 3 gives an example of a census table.

Table 3. Census information form.

Person	Age	Height	Gender	Occupation
A	22	179	Male	Student
B	54	158	Female	Teacher

In social surveys, researchers often need to tabulate extensive information about large populations—such as age, height, gender, and occupation—into tables or similar formats. Faced with a profusion of states and information, however, how to reasonably estimate information quantity has long been a challenge for sociologists.

When we translate the observed sociological states into logical statements, individuals naturally correspond to constants, while age, height, gender, occupation, and the like naturally correspond to predicates.

A row in the census table translates into a single existential statement over individual constants and attribute predicates, for example:

$$\exists x(x = a \wedge \text{Age}(x, 22) \wedge \text{Height}(x, 179) \wedge \text{Male}(x) \wedge \text{Occupation}(x, \text{Student})) \tag{42}$$

In this setting, according to the eleven metrics involved in OIT, the number of predicate types corresponds to variety; the number of instantiated records corresponds to volume. This yields a natural measurement of information and enables corresponding computations. In other words, formalization provides a measurable scale for the operationalization of concepts and variables in sociology and beyond, allowing researchers to use OIT's metrics to quantify the uncertainty and structure inherent in observations, surveys, texts, or behavioral data.

5. Computer Field State Expression

In computer science, logic plays an irreplaceable role as a fundamental tool for expressing states. In program verification, Hoare logic precisely describes the state of each execution point of a program through preconditions, postconditions, and invariants, enabling rigorous proof of program correctness. In database systems, first-order logic is not only used to define the semantics of query languages but also characterizes the legal state space of data through integrity constraints. In the field of formal methods, temporal logic (such as LTL and CTL) can express the dynamic behavior and safety properties of systems in the time dimension, providing a mathematical foundation for modeling concurrent and real-time systems. Planning problems in artificial intelligence are essentially about

finding a path from an initial state to a target state in the state space described by logic. In hardware design, Boolean logic directly corresponds to the physical state of circuits, enabling the design of complex digital systems. This abstract expressive power of logic not only provides a unified mathematical framework for modeling complex systems but, more importantly, enables automated reasoning and verification. From compiler optimization analysis to operating system resource management and from network protocol correctness verification to interpretability analysis of machine learning models, logic plays a critical role in translating intuitive concepts into computable forms [35–37].

5.1. Boolean Algebra and the Formalization of Computer Systems

Boolean algebra and logical representation are core areas of computer science and mathematical logic [38,39]. First, we verify that Boolean algebra can be formalized using first-order logic.

Theorem 13 (Axiomatizability of Boolean algebras in FOL)**.** *All axioms and operations of Boolean algebra can be expressed using first-order logic (FOL).*

Proof of Theorem 13. Boolean algebra is defined as a set B, binary operations $\wedge, \vee$, unary operations $\neg$, constants $0, 1$, and the following axioms:

$$\forall a, b \in B: \quad a \wedge b = b \wedge a \tag{43}$$

$$\forall a \in B: \quad a \vee \neg a = 1 \tag{44}$$

$$\forall a, b, c \in B: \quad a \wedge (b \vee c) = (a \wedge b) \vee (a \wedge c) \text{etc.} \tag{45}$$

Its axioms are first-order sentences; each algebra is an L-structure; hence, BA is axiomatizable in FOL. Therefore, the theorem is proved. □

Boolean algebra is the foundation of computer systems. The logic gates in digital circuits, conditional branching in programs, propositional calculus, and automata are all based on Boolean algebra. Operations, states, and transitions can all be reduced to Boolean algebraic expressions.

Because the entire content of Boolean algebra can be expressed using first-order logic, and computer systems can be reduced to Boolean algebraic structures, its core content can also be expressed using first-order logic. Practical applications such as model checking, hardware verification, and theorem proving have extensively employed first-order logic for modeling and reasoning.

Theorem 14. *Computer systems can be formally expressed in first-order logic.*

5.2. Predicate Logic Description of a Turing Machine (TM)

Figure 4 shows a sketch of a Turing machine.

Definition 18. *Similar to finite state machines, Turing machines can be formally expressed. A deterministic Turing machine (DTM) can be represented as a seven-tuple:*

$$M = (Q, \Gamma, \Sigma, \delta, q_0, q_{accept}, q_{reject}) \tag{46}$$

where:

- Q *is a finite state set.*
- Γ *is the set of tape symbols (including the blank symbol* $\sqcup$*).*

- $\Sigma \subseteq \Gamma$ *is the set of input symbols (excluding* $\sqcup$*).*
- $\delta : Q \times \Gamma \to Q \times \Gamma \times \{L, R\}$ *is the state transition function, where L and R indicate whether the read/write head moves left or right.*
- $q_0 \in Q$ *is the initial state.*
- $q_{accept}, q_{reject} \in Q$ *are the accept and reject states, respectively.*

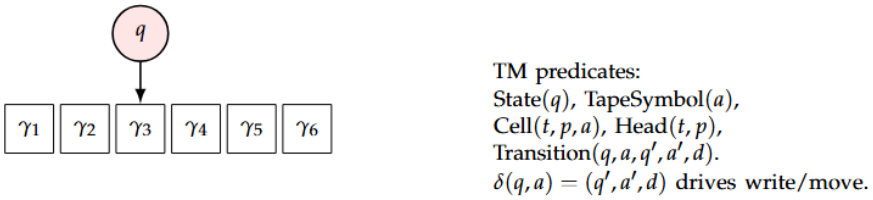

Figure 4. Turing machine snapshot: tape symbols, head at position p, current state q; FOL predicates describe configuration and transitions.

To represent a Turing machine, we can define the following predicates and give the corresponding state expressions:

Definition 19 (Predicate Definition and State Expression)**.** *The predicates and states in a Turing machine can be expressed as:*

- *State:* $State(q)$ *indicates that* q *is a state.*

$$\phi_1 = \forall q \ (State(q) \to q \in Q) \tag{47}$$

- *Tape Symbol:* $TapeSymbol(a)$ *indicates that* a *is a tape symbol.*

$$\forall a \ (TapeSymbol(a) \to a \in \Gamma) \tag{48}$$

- *Tape content:* $Cell(t, p, a)$ *indicates that* a *is the tape symbol at time* t *and position* p*.*

$$\forall t \forall p \forall a \ (Cell(t, p, a) \to a \in \Gamma) \tag{49}$$

- *Read/Write Head Position:* $Head(t, p)$ *indicates that the head is at position* p *at time* t*.*

$$\forall t \exists p \ Head(t, p) \tag{50}$$

- *Transition function:* $Transition(q, a, q', a', d)$ *represents state transition, where* d *is the direction.*

$$\forall q \forall a \forall q' \forall a' \forall d \ (Transition(q, a, q', a', d) \leftrightarrow \delta(q, a) = (q', a', d)) \tag{51}$$

- *Initial state:* $Initial(q)$ *represents the initial state* q*.*

$$\exists q_0 \ (Initial(q_0) \wedge State(q_0)) \tag{52}$$

- *Accept state (similar to the rejection state):* $Accept(q)$ *indicates that* q *is an accepting state.*

$$\forall q \ (Accept(q) \leftrightarrow q = q_{accept}) \tag{53}$$

Based on the above definition of predicates and the corresponding Turing machine state representation and state transition representation, we derive the theorem:

Theorem 15 (Turing Machine State Representation)**.** *For any Turing machine, its input, output, and state transition behavior over a relevant time set can be described using a state set.*

Before Turing, concepts such as "computation," "algorithm," and "efficient process" were intuitive. Mathematicians knew what computation was but could not give a rigorous mathematical definition. The logical representation of the Turing machine was the first to transform these intuitive concepts into precise mathematical objects: state sets, symbol sets, transition functions, initial states, and accepting states. This formalization enables us to use mathematical methods to study the properties of computation itself [40].

5.3. Mathematical Formalization of Neural Networks

Neural networks (NNs) are machine learning models that mimic the structure and function of biological neural systems, capable of learning complex patterns from data and making predictions or decisions [41]. They are a core technology in deep learning and are widely used in fields such as computer vision, natural language processing, and speech recognition.

First, let us briefly introduce neural networks. A neural network can be represented as a function $\mathcal{N} : \mathbb{R}^n \to \mathbb{R}^m$, whose hierarchical structure is decomposed into:

$$\mathcal{N}(x) = \sigma_L(W_L \cdot \sigma_{L-1}(W_{L-1} \cdots \sigma_1(W_1 x + b_1) \cdots) + b_L) \tag{54}$$

where: input layer: $x \in \mathbb{R}^n$, hidden layer: $h_l = \sigma_l(W_l h_{l-1})$, output layer: $\mathcal{N}(x) \in \mathbb{R}^m$, weight matrix: $W_l \in \mathbb{R}^{d_l \times d_{l-1}}$, bias vector: $b_l \in \mathbb{R}^{d_l}$, activation function: $\sigma_l : \mathbb{R}^{d_l} \to \mathbb{R}^{d_l}$

Single neuron calculation:

$$\sigma(w^T x + b) = \sigma\left(\sum_{i=1}^{n} w_i x_i + b\right) \tag{55}$$

Neural networks, by simulating the connections of the human brain, enable efficient modeling of complex data. With advances in computing power and algorithms (e.g., GPUs and attention mechanisms), their capabilities are continuously expanding, becoming the driving force behind the AI revolution.

Next, we demonstrate that the relevant aspects of neural networks can be formally expressed.

Proposition 1. *Neural networks can be formally expressed in first-order and higher-order logic.*

Proof of Proposition 1. The first step is to logically represent a single neuron in a neural network.

Definition 20 (Neuron Triplet)**.** *A neuron can be represented as $\mathcal{N} = (w, b, \sigma)$, where $w \in \mathbb{R}^n$ represents the weight vector, $b \in \mathbb{R}$ represents the bias term, and $\sigma : \mathbb{R} \to \mathbb{R}$ represents the activation function.*

Using the neuron triple model, we can see that the neuron input–output relationship can be logically represented as follows:

$$\forall x \in \mathbb{R}^n, \exists z, a \in \mathbb{R}, \left(z = \sum_{i=1}^{n} w_i x_i + b\right) \wedge (a = \sigma(z)) \tag{56}$$

Next, we express the activation function in the neural network. Activation functions are real-valued functions. Over real closed fields, one can encode piecewise-linear activations (e.g., ReLU) via linear constraints with slack variables; smooth activations require either algebraic approximations or δ-decision procedures.

Finally, we demonstrate the logical representation of the network topology. Taking a simple fully connected layer as an example:

- Feedforward Network: $Connected(u, v)$ indicates the connection between u, v, and $Layer_l$ indicates the lth layer.

$$\begin{aligned} &\forall l \in \{1, ..., L\}, \forall u \in \text{Layer}_l, \forall v \in \text{Layer}_{l+1}, \\ &\text{Connected}(u, v) \land \neg \exists k < l(\text{Connected}(v, \text{Layer}_k)). \end{aligned} \tag{57}$$

- Hierarchical Combination: $Network(x)$ indicates the hierarchical combination structure of x.

$$\exists \text{Network} : (\mathbb{R}^{d_0} \to \mathbb{R}^{d_L}),\ \text{Network}(x) = \sigma_L \circ W_L \circ \cdots \circ \sigma_1 \circ W_1(x) \tag{58}$$

- Combinatorial Completeness: If the lth layer can be represented as Φ_l, then the $l+1$th layer can be represented as:

$$\Phi_{l+1} = \exists y_l, \Phi_l(x, y_l) \land (y_{l+1} = \sigma_{l+1}(W_{l+1} y_l + b_{l+1})) \tag{59}$$

This completes the argument. □

According to the theorem, from a theoretical perspective, there is a profound equivalence between neural networks and logical systems. Neural networks can discover strategic patterns that humans have never discovered. If these patterns can be expressed in logical form, they may be transformed into verifiable scientific theories. This fusion is not just a technological advancement; it also represents a deepening of our understanding of the nature of intelligence: intelligence requires both the ability to learn from experience and the ability to reason based on rules, and the logical representation of neural networks is the key bridge connecting the two.

5.4. Formal Expression of States in Computer Science

According to previous proofs, computer systems, Turing machines, and neural networks can all be formalized using first-order and higher-order logic. Scholars have also formalized phenomena such as computational concepts, programming languages, and algorithmic processes [9,42,43]. Accordingly, to better articulate the theoretical claims, we provide an example of how a finite automaton can be formally expressed, which is included in Appendix A.2.

Theorem 16. *The following mainstream models can be formalized within specified logical fragments:*

1. *Boolean circuits, finite automata, and finite-state concurrent models: first-order logic can completely characterize any fixed instance.*
2. *Turing machines' state evolution and computability statements: their behavior and halting properties can be expressed in extensions of first-order or second-order arithmetic/set theory.*
3. *Neural networks (with finite depth/width over a given real field or its axiomatization): their topology and forward computation can be described in first- or higher-order logic.*

Theorem 16 demonstrates that logic provides a unified framework for expressing different computational models. Logical formalization has transformed computer science from an engineering craft into a rigorous scientific discipline, providing powerful tools for understanding and controlling complex systems. Furthermore, emerging problems in computer science can be gradually formalized using the proof process of neural networks. Based on this, we draw a comprehensive conclusion.

6. State Expression in Natural Language Domain

Natural languages, such as Chinese, English, and Arabic, are the languages humans use in everyday life. They are essential tools for communicating ideas and conveying information. In contrast to formal languages, they possess unique properties. Natural languages are richly expressive, capable of expressing myriad worlds, complex emotions, and abstract concepts. This chapter introduces Montague semantics, explores the rules governing the grammar and semantic translation of natural languages, and achieves a formal understanding of natural languages through the principles of formalized mathematical logic.

6.1. Montague Semantics

Montague semantics, also known as Montague grammar, is a formalized approach to the study of natural semantics, particularly intensional semantics. It represents a new stage in the development of linguistics and logic.

Montague's research began with the concept of categories, dividing English syntax into distinct categories. Montague introduced a system of syntactic categories and formation rules for English, defined meaningful expressions via recursion, interpreted them using model theory, and provided corresponding semantic translation rules [44].

Based on Montague's work, Bennett conducted further research. He refined the English categories, dividing adjectives into separate categories; introduced more complex grammatical and semantic translation rules, expanding the rules from 17 to 35; and provided a solid theoretical foundation for language research using Montague semantics [45].

6.2. Study of English Ambiguity

Correctly handling linguistic ambiguity is an important indicator of semantic comprehension. Here, we use Montague grammar to provide an interpretation of ambiguity.

"At least one person likes the book" is a common ambiguous phrase in English. The ambiguity of "at least one person likes the book" stems from the ambiguity of the quantifier scope. Without context, "the book" could refer to a specific book or to a general term.

The semantics of the sentence "At least one person likes the book" can be formally modeled using the λ calculus, with two possible interpretations: a broad interpretation and a narrow interpretation.

In the broad interpretation, "the book" refers to a specific book b. The logical form of the entire sentence is:

$$\exists x[man'(x) \wedge like'(\hat{}b')(x)] \tag{60}$$

where:

- $man'(x)$ means x is a person;
- $like'(\hat{}b')(x)$ means x likes a specific book b.

The specific λ calculus combination process is as follows:

$$\begin{aligned}
\text{Like} &: like' \\
\text{The book} &: b' \\
\text{Like the book} &: like'(\hat{}b') \\
\text{At least one person} &: \lambda P.\exists x[man'(x) \wedge P\{x\}] \\
\text{Combination result} &: \lambda P.\exists x[man'(x) \wedge P\{x\}](\hat{}like'(\hat{}b')) \\
\lambda \text{ transposition} &: \exists x[man'(x) \wedge \hat{}like'(\hat{}b')\{x\}] \\
\text{Bracket convention} &: \exists x[man'(x) \wedge \check{}\hat{}like'(\hat{}b')(x)] \\
\text{Top and bottom elimination} &: \exists x[man'(x) \wedge like'(\hat{}b')(x)]
\end{aligned}$$

In the narrow-scope interpretation, "the book" is not a specific object but a quantifiable range. The logical form of the sentence is:

$$\exists x[man'(x) \wedge (like'(x, \hat{}\lambda Q.\exists b[book'(b) \wedge Q\{b\}]))] \tag{61}$$

where:

- $book'(b)$ indicates that b is a book;
- $(like'(x, \hat{}\lambda Q.\exists b[book'(b) \wedge Q\{b\}]))$ indicates that there exists at least one person x, there exists a book b, and this person likes book b.

The specific λ calculus combination process is as follows:

$$\begin{aligned}
\text{Like} &: like' \\
\text{At least one person} &: \lambda P.\exists x[man'(x) \wedge P\{x\}] \\
\text{The book exists} &: \lambda Q.\exists b[book'(b) \wedge Q\{b\}] \\
\text{Like the book} &: like'(\hat{}\lambda Q.\exists b[book'(b) \wedge Q\{b\}]) \\
\text{Combination result} &: \lambda P.\exists x[man'(x) \wedge P\{x\}](\hat{}like'(\hat{}\lambda Q.\exists b[book'(b) \wedge Q\{b\}])) \\
\lambda\text{Transposition} &: \exists x[man'(x) \wedge (\hat{}like'(\hat{}\lambda Q.\exists b[book'(b) \wedge Q\{b\}]))\{x\}] \\
\text{Bracket convention} &: \exists x[man'(x) \wedge (\check{}\hat{}like'(\hat{}\lambda Q.\exists b[book'(b) \wedge Q\{b\}]))(x)] \\
\text{Up and Down Elimination} &: \exists x[man'(x) \wedge (like'(\hat{}\lambda Q.\exists b[book'(b) \wedge Q\{b\}]))(x)] \\
\text{Relational notation} &: \exists x[man'(x) \wedge (like'(x, \hat{}\lambda Q.\exists b[book'(b) \wedge Q\{b\}]))]
\end{aligned}$$

As shown in Figure 5, by modeling the λ calculus, the sentence "At least one person likes this book" can capture the following two semantic interpretations:

1. Broad interpretation: There is at least one person who likes this particular book:

$$\exists x[man'(x) \wedge like'(\hat{}b')(x)] \tag{62}$$

2. Narrow interpretation: There is a book and at least one person likes it:

$$\exists x[man'(x) \wedge (like'(x, \hat{}\lambda Q.\exists b[book'(b) \wedge Q\{b\}]))] \tag{63}$$

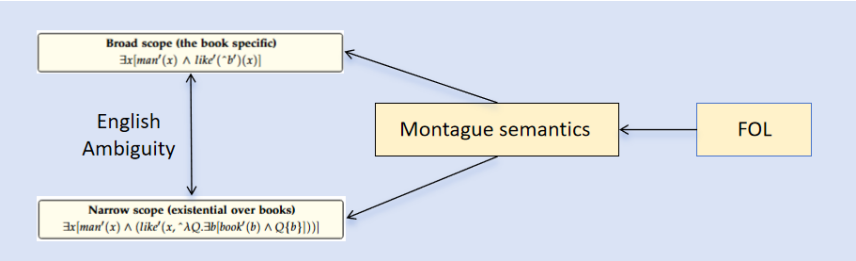

Figure 5. Quantifier scope ambiguity illustrated in FOL/λ-calculus style forms.

Similarly, the ambiguous phenomenon of "Every student read a book" has been extensively studied and interpreted. The quantifier scope ambiguity involved in this issue is an active frontier in semantic research, continuing to drive theoretical and technological developments [46]. Studying such issues can further advance the research and development of natural semantics.

6.3. Optimizing Syntax and Semantic Translation Rules

Although Montague, Bennett, and others have established detailed rules for English semantics and grammar, which later generations can simply apply directly, many issues still need to be resolved in the actual translation process.

6.3.1. Conjunction Rules

Among Montague's 17 rules, S11 and S12 deal with conjunction, defining parallel sentences and parallel verbs, respectively. Among Bennett's 35 rules, S28 deals with conjunction, defining only parallel verbs. However, as we know, conjunctions of nouns and noun phrases occur very frequently in natural language. Surprisingly, neither Montague nor Bennett provide corresponding grammatical rules for this phenomenon. This is because conjunctions of nouns and noun phrases require the verb to become plural, and to simplify expression, no corresponding grammatical rules were defined. However, given the high frequency and widespread use of nouns and noun phrases, and to help beginners better grasp the relevant content, we provide the following additional rules:

Definition 21 (Grammatical Rules for Conjunction of Nouns and Noun Phrases)**.** *If $\alpha, \beta \in P_{CN}$, then $F(\alpha, \beta) \in P_{CN}$. If $\alpha, \beta \in P_T$, then $F(\alpha, \beta) \in P_T$. Here, $F(\alpha, \beta) = \alpha$ and β. And when the object of the conjunction becomes the subject, the corresponding verb becomes plural.*

Definition 22 (Translation Rules for Conjunctions of Nouns and Noun Phrases)**.** *If $\alpha, \beta \in P_{CN}$ or $\alpha, \beta \in P_T$, and α, β is translated as α', β', then α and β is translated as $\lambda P[\alpha'(P) \wedge \beta'(P)]$. When the conjunction object serves as the subject, the corresponding verb is translated into its plural form.*

6.3.2. Adjective Rules

Among Bennett's 35 rules, the one concerning adjectives is S10. We give its original definition:

Definition 23 (S10)**.** *If $\gamma \in P_{AJ}$ and $\zeta \in P_{CN}$, then $F_9(\gamma, \zeta) \in P_{CN}$, where*

(a) if γ contains an occurrence of a member of $B_{AJ/T}$, then $F_\varphi(\gamma, \zeta) = \zeta\gamma$;
(b) otherwise $F_9(\gamma, \zeta) = \gamma\zeta$.

Bennett argues that using S10 can resolve the English grammatical phenomenon of adjective + noun. However, let us consider the following example: John's mother. According to Bennett's definition, John's does not fall into the basic category of adjectives, so S10 cannot be used for translation. We must instead use S5.

Definition 24 (S5)**.** *If $\zeta \in P_{CN/T}$ and $\alpha \in P_T$, then $F_5(\zeta, \alpha) \in P_{CN}$, where*

(a) if $\alpha = he_n$, then $F_5(\zeta, \alpha) = \zeta * him$;
(b) otherwise $F_5(\zeta, \alpha) = \zeta\alpha$.

Consider John's mother to be equivalent to mother of John. This can be translated as:

$$\text{mother} : mother'$$
$$\text{John} : \lambda P[P\{j\}]$$
$$\text{mother of John} : mother'(\hat{}(\lambda P[P\{j\}]))$$

While there is certainly nothing wrong with translating "John's mother" this way, treating "John's mother" and "mother of John" as equivalent loses the distinction between the two grammatical structures. Furthermore, the resulting translation is often less concise and clear, hindering the reader's intuitive understanding. Therefore, we provide supplementary rules for these situations.

Definition 25 (Grammar rule). *If $\alpha \in P_{CN}$ or $\alpha \in P_T$, then $G_2(\alpha) \in P_{AJ}$, where*

(a) If $\alpha = he_n$, then $G_2(\alpha) = his_n$;
(b) Otherwise $G_2(\alpha) = \alpha's$.

Definition 26 (Semantic Translation Rules). *If $\alpha \in P_{CN}$ or $\alpha \in P_T$, and α is translated as α'', then*

(a) his_n is translated as $his_n{}'$;
(b) John's is translated as $\lambda P[P\{j's\}]$.
(c) In other cases, $\alpha's$ is translated as $\alpha''s'$.

Using the new rules, we can retranslate John's mother:

$$\begin{aligned} \text{mother} &: mother' \\ \text{John's} &: \lambda P[P\{j's\}] \\ \text{John's mother} &: \lambda P[P\{j\}](\hat{}\,mother') \\ \lambda \text{ transposition} &: \hat{}\,mother'\{j\} \\ \text{Brackets, upper and lower rules} &: mother'(j's) \end{aligned}$$

In comparison, the translation using the new rules is more concise and clear, making it easier for readers to grasp the grammatical structure.

6.3.3. Clause Rules

For the sake of brevity, neither Montague nor Bennett discussed clause rules in detail, instead focusing on the typical relation "such that." Some might question whether or not each clause has a specific function and introductory phrase, expressing different logical relationships depending on the context. Since their functions are not identical to "such that," does this mean that the same rules cannot be applied universally?

Indeed, "such that" primarily expresses a result or condition. Commonly used attributive clauses, such as modifiers and adverbial clauses, often express time or cause, differ significantly from "such that." However, these commonly used clauses are structurally simpler than standard relative clauses. Generally speaking, one can emulate Montague's approach to translation by modifying or simplifying Montague's rules or by transforming the clause form.

Here, we provide a case study: the man whom Mary loves.

man : man'

the man : $\lambda P \exists y[\forall x[man'(x) \leftrightarrow x = y] \wedge P\{y\}]$

Mary : $\lambda Q[Q\{m\}]$

loves : $love'$

loves the man : $love'(\hat{}\,(\lambda P \exists y[\forall x[man'(x) \leftrightarrow x = y] \wedge P\{y\}]))$

Mary loves the man : $\lambda Q[Q\{m\}](\hat{}\,love'(\hat{}\,(\lambda P \exists y[\forall x[man'(x) \leftrightarrow x = y] \wedge P\{y\}])))$

λTransposition : $\hat{}\,\lambda P \exists y[\forall x[man'(x) \leftrightarrow x = y] \wedge P\{y\}]))\{m\}$

Brackets, upper and lower rules : $\lambda P \exists y[\forall x[man'(x) \leftrightarrow x = y] \wedge P\{y\}]))(m)$

λTransposition : $\exists y[\forall x[man'(x) \leftrightarrow x = y] \wedge m\{y\}]))$

Combined results:

$$\begin{aligned}&\lambda x_n[(\lambda P\exists y[\forall x[man'(x) \leftrightarrow x = y] \wedge P\{y\}])(x_n)\wedge\\&(\exists y[\forall x[man'(x) \leftrightarrow x = y] \wedge m\{y\}]))]\end{aligned} \tag{64}$$

After λ transposition, brackets, and upper and lower rules, the final result is:

$$\begin{aligned}&\lambda x_n[\exists y[\forall x[man'(x) \leftrightarrow x = y] \wedge \check{}\, x_n(y)])\wedge\\&(\exists y[\forall x[man'(x) \leftrightarrow x = y] \wedge \check{}\, m(y)]))]\end{aligned} \tag{65}$$

The core insight of Montague semantics lies in placing natural language within a formal framework as rigorous as that of mathematics and logic. By supplementing these rules, scholars can better understand natural language through logic. Formal logic is not just an abstract mathematical tool; it is a key approach to understanding and modeling the most complex human cognitive abilities. As artificial intelligence progresses toward general intelligence, the formal methodology represented by Montague semantics will continue to play an irreplaceable role.

6.4. Formalization of Natural Languages

Scholars such as Montague have provided comprehensive tools and detailed introductions to the formalization of English, a natural language. Furthermore, combined with the rules subsequently added by scholars, it can be argued that English can be fully formalized. Similarly, other natural languages can be formalized using similar methods. Alternatively, given the intertranslatability between English and other languages, they can be directly converted into English for formalization.

In short, we can conclude the following:

Theorem 17. *Large fragments of natural languages can be formalized using first- and higher-order logic with appropriate extensions of symbols and rules; the scope of full formalization depends on the targeted phenomena and resources.*

The importance of formalization of natural languages ultimately lies in the scientific method it provides for understanding the essence of human intelligence. Language is not only a tool for communication but also a vehicle for thought, a container for knowledge, and a medium for the transmission of culture. Formalizing language is a mathematical modeling of human cognitive abilities and a scientific exploration of the essence of intelligence.

7. Conclusions and Outlook

This paper systematically investigates the logical formalization of object states, proposing and demonstrating a rigorous and universally applicable formalization framework for revealing the nature of information and its interdisciplinary applications. The paper first reviews classical information theory and its shortcomings, noting the current lack of a unified and rigorous mathematical definition of the core concept of "state."

To this end, this paper establishes a universal representation system for information states based on first-order and higher-order predicate logic, combined with modal logic and calculus, addressing the current lack of formal representations for states. This paper enumerates typical states from various fields, including mathematics, economics, sociology, computer science, and natural language, and rigorously proves that these states can be formalized using first-order and higher-order logic. Although this paper discusses the formalization of states in only four specific domains, these domains are highly representative. For any domain's state, its numerical features can always be expressed mathematically,

its attributes can always be represented by phenomena in economics and sociology, it can be implemented in computers, and its intention can be explained in natural language. In other words, the states of all domains can borrow the pattern we have proved to achieve formalization, thereby generalizing the formal expression of states from the particular to the general.

In this sense, logic has truly become a universal bridge connecting states across various fields, a universal language that enables communication across fields, and provides humanity with the most fundamental and powerful mathematical tools for understanding and transforming the world.

Through the formalization of states, objective information theory (OIT) has been further refined and developed, deepening research on the nature of information and expanding its scope from the classic Shannon framework. Many pressing problems in information science can be transformed into logical problems. By studying the properties of logical language and drawing on proven conclusions and axioms from logic, we can clarify the meaning of information and guide the development of information research.

Author Contributions: Conceptualization, S.Q. and J.X.; methodology, S.Q. and J.X.; writing—original draft, S.Q.; writing—review & editing, S.Q. and J.X.; project administration, J.X. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Data are contained within the article. Further inquiries can be directed to the corresponding author.

Acknowledgments: Here, I would like to thank Wang Rui, as well as my classmates Chun Li, Hu Xu, Zeyan Li, and Jiashuo Zhang for their continued support and help.

Conflicts of Interest: The authors declare no conflicts of interest. The authors have identified and declared that there are no personal circumstances or interests that may be perceived as inappropriately influencing the representation or interpretation of the reported research results. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

Abbreviations

The following abbreviations are used in this manuscript:

OIT	Objective Information Theory
FOL	First-order predicate logic
HOL	Higher-order predicate logic
ASM	Abstract State Machines
CTL	Computation Tree Logic
LTL	Linear Temporal Logic
PDL	Propositional Dynamic Logic
TLA+	Temporal Logic of Actions
wff (wffs)	well-formed formula (well-formed formulas)

Appendix A

Appendix A.1

Here, we will define higher-order formal systems. Assume that for $1, 2, \ldots, k-1$, the corresponding formal systems $\mathcal{L}^{(1)}, \mathcal{L}^{(2)}, \ldots, \mathcal{L}^{(k-1)}$ have been defined. Then, the symbols in the k-order formal system $\mathcal{L}^{(k)}$ can be recursively defined [47].

Definition A1 (Symbols in $\mathcal{L}^{(k)}$)**.** *The symbols in $\mathcal{L}^{(k)}$ include:*

- *All symbols of $\mathcal{L}^{(k-1)}$;*
- *k-order variables:$x_1^{(k)}, x_2^{(k)}, \ldots$*
- *k-order constants:$a_1^{(k)}, a_2^{(k)}, \ldots$*
- *k-order predicate variables:$P_1^{(k)1}, P_2^{(k)1}, \ldots, P_1^{(k)2}, P_2^{(k)2}, \ldots$*
- *k order function symbols:$f_1^{(k)1}, f_2^{(k)1}, \ldots, f_1^{(k)2}, f_2^{(k)2}, \ldots$*
- *k-order predicate symbols:$A_1^{(k)1}, A_2^{(k)1}, \ldots, A_1^{(k)2}, A_2^{(k)2}, \ldots$*

Similarly, we can recursively define the terms in $\mathcal{L}^{(k)}$, atomic formulas and well-formed formulas.

Definition A2 (Terms in $\mathcal{L}^{(k)}$)**.** *The terms in $\mathcal{L}^{(k)}$ are defined as follows:*

(1) *All terms of $\mathcal{L}^{(k-1)}$;*

(2) *If $f_i^{(k)n}$ $(n > 0, i > 0)$ is a k-order function symbol in $\mathcal{L}^{(k)}$ and $u_1, \ldots, u_n$ are variables, constants, or functions in $\mathcal{L}^{(k)}$, then $f_i^{(k)n}(u_1, \ldots, u_n)$ is a k-order term in $\mathcal{L}^{(k)}$.*

Definition A3 (Atomic formulas in $\mathcal{L}^{(k)}$)**.** *The atomic formula in $\mathcal{L}^{(k)}$ is defined as follows:*

(1) *All atomic formulas of $\mathcal{L}^{(k-1)}$;*

(2) *If $A_i^{(k)n}$ $(n > 0, i > 0)$ is a predicate symbol of order k in $\mathcal{L}^{(k)}$ and $u_1, \ldots, u_n$ are terms in $\mathcal{L}^{(k)}$, then $A_i^{(k)n}(u_1, \ldots, u_n)$ is an atomic formula of order k in $\mathcal{L}^{(k)}$.*

Definition A4 (Well-formed formula in $\mathcal{L}^{(k)}$)**.** *The well-formed formula in $\mathcal{L}^{(k)}$ is defined as follows:*

(1) *All well-formed formulas for $\mathcal{L}^{(k-1)}$;*

(2) *If $\mathcal{A}$ and $\mathcal{B}$ are well-formed formulas in $\mathcal{L}^{(k)}$, then $\sim \mathcal{A}$ and $\mathcal{A} \to \mathcal{B}$ are both well-formed formulas in $\mathcal{L}^{(k)}$;*

(3) *If $\mathcal{A}$ is a well-formed formula in $\mathcal{L}^{(k)}$ and u is an argument or function symbol in $\mathcal{L}^{(k)}$, then $(\forall u)\mathcal{A}$ is a well-formed formula in $\mathcal{L}^{(k)}$.*

Appendix A.2

Using the formal expression of states, we can study in depth the special and important object of finite automata. The structure of a finite automaton is shown in Figure A1.

Theorem A1 (State Representation of Finite Automata)**.** *For any finite automaton M, one can describe its input, output, and state-transition behavior over a related time set T by the state set $S(M, T)$.*

Proof. Let $M = (Q, R, U, \delta, \lambda)$ be a finite automaton, and $T = \{t_i | i = 1, \ldots, n\}$ the (strictly increasing) time points at which M performs input, output, or state-transition actions. By finite automaton theory, $Q = \{q_{t_i} | t_i = 1, \ldots, n-1\}$, $R = \{r_{t_i} | t_i = 2, \ldots, n\}$, and $U = \{u_{t_i} | t_i = 1, \ldots, n\}$ are nonempty finite sets of input symbols, output symbols, and states, respectively;

$\delta : U \times Q \to U$ is the next-state function;

$\lambda : U \times Q \to R$ is the output function.

For each state $u_{t_i} \in U$, define the state predicate $State^3$ by

$$\varphi_U(t_i) = State^3(M, t_i, u_{t_i}) \tag{A1}$$

to assert that at time t_i, M is in state u_{t_i},$i = 1, \ldots, n$.

For each input $q_{t_i} \in Q$, define the input predicate $Input^3$ by

$$\varphi_I(t_i) = Input^3(M, t_i, q_{t_i}) \tag{A2}$$

to assert that at time t_i, M receives input q_{t_i},$i = 1, \ldots, n-1$.

For each output $r_{t_i} \in R$, define the output predicate $Output^3$ by

$$\varphi_O(t_i) = Output^3(M, t_i, r_{t_i}) \tag{A3}$$

to assert that at time t_i, M produces output r_{t_i},$i = 2, \ldots, n$.

The transition function δ is captured by the well-formed formula

$$\varphi_\delta(t_i) = \varphi_U(t_i) \wedge \varphi_I(t_i) \rightarrow \exists q_{t_i}((u_{t_{i+1}} = \delta(u_{t_i}, q_{t_i})) \wedge \varphi_U(t_{i+1})) \tag{A4}$$

where $i = 1, \ldots, n-1$.

Similarly, the output function λ is given by

$$\begin{aligned} \varphi_\lambda(t_i) = \varphi_U(t_i) \wedge \varphi_I(t_i) \rightarrow \exists q_{t_i}((r_{t_{i+1}} = \lambda(u_{t_i}, q_{t_i})) \wedge \\ IsElementof^2(r_{t_{i+1}}, R) \wedge \varphi_O(t_{i+1})) \end{aligned} \tag{A5}$$

where $IsElementof^2(x, X)$ asserts x is an element of X,$i = 1, \ldots, n-1$

Thus, the state set

$$\begin{aligned} S(M, T) = \{ & \varphi_U(t_i), \varphi_I(t_j), \varphi_O(t_k), \varphi_\delta(t_j), \varphi_\lambda(t_j), \\ & i = 1, \ldots, n, j = 1, \ldots, n-1, k = 2, \ldots, n\} \end{aligned} \tag{A6}$$

describes M's inputs, outputs, and state-transitions over T. This completes the proof.

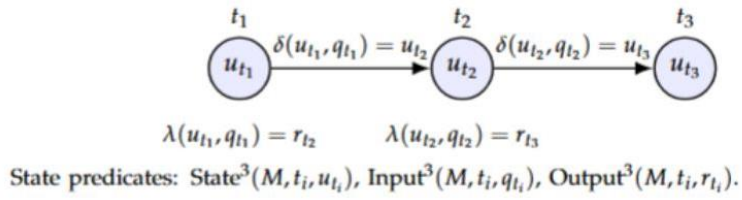

Figure A1. Finite automaton: time-indexed states with inputs and outputs; FOL predicates encode transition/output functions.

It follows that every finite automaton can be formalized by the state set given in Theorem A1, which moreover exhibits automatic transition behavior. Hence, we introduce:

Definition A5. *Finite automaton state if a state set $S(X, T)$ expresses a finite automaton, then $S(X, T)$ is called a finite automaton state.*

All practical information systems, including computing systems, are finite automata in the mathematical sense. Therefore, the concept of finite automaton state is of great significance in the study of machine learning. □

Appendix A.3

This appendix provides a compact and widely known example of formalizing an economic theory in logic: the Arrow–Debreu Walrasian equilibrium in a finite exchange economy. We present a first-order (FOL) specification over the language of real closed fields

for a fixed finite instance. In this setting, the equilibrium existence statement reduces to an existential FOL formula, and hence is decidable (Tarski–Seidenberg).

Let the set of consumers be $I = \{1, \ldots, n\}$ and the set of goods be $G = \{1, \ldots, m\}$. Each consumer i has an initial endowment $e_i \in \mathbb{Q}_+^m$. We consider standard parametric utility families to avoid higher-order encodings; two canonical choices are:

- Cobb–Douglas: $u_i(x_i) = \prod_{g=1}^m x_{ig}^{\alpha_{ig}}, \quad \alpha_{ig} \geq 0, \sum_{g=1}^m \alpha_{ig} = 1.$
- CES with $\rho_i \neq 0, -\infty$: $u_i(x_i) = \left(\sum_{g=1}^m \beta_{ig} x_{ig}^{\rho_i} \right)^{1/\rho_i}, \quad \beta_{ig} > 0.$

Prices are $p \in \mathbb{Q}_+^m$ (normalized) and consumptions are $x_i \in \mathbb{Q}_+^m$. All parameters $\{e_i\}, \{\alpha_{ig}\}$ or $\{\beta_{ig}, \rho_i\}$ are treated as structure constants.

We work in the FOL language of ordered fields with addition, multiplication, order, and equality. The variables are:

$$p_1, \ldots, p_m \text{ (prices)}, \qquad x_{ig} \text{ (consumptions)}.$$

For KKT-based encodings (e.g., CES), we may also introduce multipliers λ_i, μ_{ig}.

A Walrasian equilibrium $(p, \{x_i\}_{i \in I})$ satisfies the following:

(1) Price normalization and nonnegativity.

$$\forall g \in G: \; p_g \geq 0, \qquad \sum_{g=1}^m p_g = 1. \tag{A7}$$

(2) Individual feasibility and budget exhaustion. For each $i \in I$,

$$\sum_{g=1}^m p_g x_{ig} = \sum_{g=1}^m p_g e_{ig}, \qquad x_{ig} \geq 0 \;\; \forall g. \tag{A8}$$

Under strictly monotone preferences, optimal choices exhaust the budget (equality holds).

(3A) Individual optimality via closed-form demand (Cobb–Douglas). Let $b_i := \sum_{g=1}^m p_g e_{ig}$. For each $i \in I$ and $g \in G$,

$$x_{ig} = \alpha_{ig} \frac{b_i}{p_g}. \tag{A9}$$

These equalities, together with (2), are equivalent to optimality for Cobb–Douglas utilities, avoiding the use of arg max or second-order quantification.

(3B) Individual optimality via KKT conditions (CES or smooth, strictly quasiconcave utilities).

Introduce multipliers $\lambda_i \geq 0$ and $\mu_{ig} \geq 0$. For each $i \in I$ and $g \in G$,

$$\frac{\partial u_i}{\partial x_{ig}}(x_i) - \lambda_i p_g - \mu_{ig} = 0, \qquad \mu_{ig} \, x_{ig} = 0, \tag{A10}$$

together with the budget exhaustion and nonnegativity in (2). If preferences are strictly monotone, we can add $(x_{ig} > 0) \Rightarrow (\mu_{ig} = 0)$. When exponents are rational, auxiliary variables and algebraic identities can be used to eliminate radicals, keeping the encoding within the real closed field framework. Alternatively, δ-decision procedures can be employed.

(4) Market clearing.

For each $g \in G$,

$$\sum_{i=1}^n x_{ig} = \sum_{i=1}^n e_{ig}. \tag{A11}$$

For a fixed finite instance with parameters as constants, the existence of a Walrasian equilibrium is expressed by the existential FOL sentence:

$$\exists\, p, \{x_i\}_{i\in I}\ \big[\ \text{(A7)} \wedge \text{(A8)} \wedge \text{(A11)} \wedge \big(\text{(A9) or (A10)}\big)\ \big].$$

In particular, with Cobb–Douglas utilities, (3A) yields a purely algebraic system, so the existence claim reduces to solvability of polynomial equalities and inequalities under normalization.

Appendix A.4

Tables A1–A5 give the meanings of the symbols used in the text.

Table A1. Symbol summary (general logic and set notation).

Symbol	Meaning
x, y, z	Individual variables (elements, times, states depending on context)
a, b, c	Individual constants (constant symbols)
f, g	Function symbols
P, Q, R	Predicate/relation symbols (also used as higher-order predicate variables)
$\neg, \to$	Logical connectives: negation, implication
$\wedge, \vee, \leftrightarrow$	Logical connectives: conjunction, disjunction, biconditional
$\forall, \exists$	Quantifiers: universal, existential
$\models$	Semantic entailment/satisfaction (a structure satisfies a formula)
$A \models \varphi,\ A \not\models \varphi$	Structure A satisfies/does not satisfy φ
$\vdash$	Syntactic provability
$\equiv, \approx$	Logical equivalence / same truth-value (as used in context)
$=, \neq$	Equality / inequality
$\in, \subseteq, \subset$	Membership, inclusion, proper inclusion
$\lvert X\rvert$	Cardinality of set X
dom, rng	Domain, range of a function (when needed)
$\mathrm{ar}(R)$	Arity of relation R
$\sum, \prod$	Summation, product
sup, inf	Supremum, infimum (when used)

Table A2. Formal languages and syntax (FOL/HOL).

Symbol	Meaning
$L^{(1)}$	First-order language
$L^{(k)}$	k-th order language (defined recursively)
L	Generic formal system/language
$x_i^{(1)},\ x_i^{(k)}$	First-/higher-order variables
$a_i^{(1)},\ a_i^{(k)}$	First-/higher-order constants
$f_i^{(1),n},\ f_i^{(k),n}$	Function symbols (arity n ,order k)
$A_i^{(1),n},\ A_i^{(k),n}$	Predicate symbols (arity n, order k)
$A(u_1, \ldots, u_n)$	Atomic formula
$\neg A,\ A \to B,\ (\forall u)A$	Formula formation (negation, implication, quantification)
term	Variable, constant, or function term
WFF	Well-formed formula

Table A3. Interpretations and structures.

Symbol	Meaning
$E = \langle D_E, J \rangle$	Interpretation (domain + interpretation function)
D_E	Domain (nonempty universe)
J	Interpretation function mapping symbols to D_E-objects/relations
$J(t),\ J(A)$	Interpretation value of a term/formula (truth value for formulas)
$\mathfrak{A} = \langle A, R_1^A, \ldots, f_1^A, \ldots, c_1^A, \ldots \rangle$	L-structure/model
$\mathfrak{B}$	Another L-structure
g, h, f	Homomorphisms/embeddings/isomorphisms between structures
$\mathfrak{A} \cong \mathfrak{B}$	Isomorphism between structures A and B

Table A4. Model theory and infinitary logic.

Symbol	Meaning
$L_{\omega_1,\omega}$	Infinitary logic with countable (co)infinite conjunctions/disjunctions
φ_A	Scott sentence of structure A
$\mathrm{Tp}_k(\varphi),\ \mathrm{Tp}_k(M)$	Set of k-types occurring in models of φ / in structure M
$d(\varphi, \psi)$	Distance on the space of Scott sentences
$d_k(\varphi, \psi)$	k-component distance based on symmetric-difference of k-types
$\{M_n\}$	Recursive approximation sequence of structures
$M = \lim_{n\to\infty} M_n$	Limit structure (with well-defined union of relations)
Local finiteness	Controlled local growth of types (as defined in the paper)

Table A5. Time and state.

Symbol	Meaning
X	Set of objects
T	Set of time points or intervals
$S(x, t)$	State of object x at time t
$\phi_S(x, t)$	Set of WFFs in L characterizing $S(x, t)$

References

1. Wiener, N. *Cybernetics or Control and Communication in the Animal and the Machine*; The MIT Press: Cambridge, MA, USA, 2019.
2. Shannon, C.E. The mathematical theory of communication. *Bell Syst. Tech.* **1948**, *27*, 379–423. [CrossRef]
3. von Neumann, J. *Mathematische Grundlagen der Quantenmechanik*; Springer: Berlin/Heidelberg, Germany, 1971; Volume 38.
4. Kolmogorov, A.N. Three approaches to the quantitative definition of information. *Int. J. Comput. Math.* **1968**, *2*, 157–168. [CrossRef]
5. Xu, J.; Ma, X.; Shen, Y.; Tang, J.; Xu, B.; Qiao, Y. Objective information theory: A Sextuple model and 9 kinds of metrics. In Proceedings of the 2014 Science and Information Conference, London, UK, 27–29 August 2014; IEEE: Piscataway, NJ, USA, 2014; pp. 793–802.
6. Xu, J.; Ma, X.; Tang, J. Research on model and measurement of objective information. *Sci. China Inf. Sci.* **2015**, *45*, 336–353. (In Chinese)
7. Xu, J.; Liu, Z.; Wang, S.; Zheng, T.; Wang, Y.; Wang, Y.; Dang, Y. Foundations and applications of information systems dynamics. *Engineering* **2023**, *27*, 254–265.
8. Hamilton, A.G. *Logic for Mathematicians*; Cambridge University Press: Cambridge, UK, 1988.

9. Xu, J. Information science principles of machine learning: A causal chain meta-framework based on formalized information mapping. *arXiv* **2025**, arXiv:2505.13182. [CrossRef]
10. Tarski, A. Contributions to the theory of models. I. In *Indagationes Mathematicae (Proceedings)*; Elsevier BV: Amsterdam, The Netherlands, 1954; Volume 57, pp. 572–581.
11. Tarski, A. *The Concept of Truth in Formalized Languages*; Clarendon Press: Oxford, UK, 1956.
12. Goguen, J.A.; Burstall, R.M. Institutions: Abstract model theory for specification and programming. *J. ACM (JACM)* **1992**, *39*, 95–146.
13. Rutten, J.J. Universal coalgebra: A theory of systems. *Theor. Comput. Sci.* **2000**, *249*, 3–80. [CrossRef]
14. Tang, G.; Fu, R.; Seiti, H.; Chiclana, F.; Liu, P. A novel bi-objective R-mathematical programming method for risk group decision making. *Inf. Fusion* **2025**, *118*, 102902.
15. Kripke, S.A. Semantical considerations on modal logic. *Acta Philos. Fenn.* **1963**, *16*, 83–94.
16. Pnueli, A. The temporal logic of programs. In Proceedings of the 18th Annual Symposium on Foundations of Computer Science (sfcs 1977), Providence, RI, USA, 31 October–2 November 1977; IEEE: Piscataway, NJ, USA, 1977.
17. Harel, D.; Kozen, D.; Tiuryn, J. Dynamic logic. *ACM SIGACT News* **2001**, *32*, 66–69. [CrossRef]
18. Lamport, L. The temporal logic of actions. *ACM Trans. Program. Lang. Syst. (TOPLAS)* **1994**, *16*, 872–923. [CrossRef]
19. Gurevich, Y.; Börger, E. Evolving algebras 1993: Lipari guide. *Evol. Algebr.* **1995**, *40*, 2.
20. Swan, R.G. *K-Theory of Finite Groups and Orders*; Springer: Berlin/Heidelberg, Germany, 2006; Volume 149.
21. Lidl, R.; Niederreiter, H. *Finite Fields*; Number 20; Cambridge University Press: Cambridge, UK, 1997.
22. Libkin, L. *Elements of Finite Model Theory*; Springer: Berlin/Heidelberg, Germany, 2004; Volume 41.
23. Chang, C.C.; Keisler, H.J. *Model Theory*; Elsevier: Amsterdam, The Netherlands, 1990; Volume 73.
24. Dedekind, R. Was sind und was sollen die zahlen? In *Was Sind und Was Sollen die Zahlen? Stetigkeit und Irrationale Zahlen*; Springer: Berlin/Heidelberg, Germany, 1965; pp. 1–47.
25. Peano, G. *Arithmetices Principia: Nova Methodo Exposita*; Fratres Bocca: Caringbah, Australia, 1889.
26. Scott, D. Logic with denumerably long formulas and finite strings of quantifiers. In *The Theory of Models*; Elsevier: Amsterdam, The Netherlands, 2014; pp. 329–341.
27. Debreu, G. *Theory of Value: An Axiomatic Analysis of Economic Equilibrium*; Yale University Press: New Haven, CT, USA, 1959; Volume 17.
28. Shoham, Y.; Leyton-Brown, K. *Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations*; Cambridge University Press: Cambridge, UK, 2008.
29. Geanakoplos, J. Three brief proofs of arrow's impossibility theorem. *Econ. Theory* **2005**, *26*, 211–215. [CrossRef]
30. Arthur, W.B.; Durlauf, S.N.; Lane, D.A. *The Economy as an Evolving Complex System ii*; Adison Wesley: Reading, MA, USA, 1997.
31. Hintikka, J.; Kulas, J. *Anaphora and Definite Descriptions: Two Applications of Game-Theoretical Semantics*; Springer Science & Business Media: Berlin, Germany, 1985; Volume 26.
32. Hausman, D.M. *The Inexact and Separate Science of Economics*; Cambridge University Press: Cambridge, UK, 2023.
33. Thornton, P.H.; Ocasio, W.; Lounsbury, M. *The Institutional Logics Perspective: A New Approach to Culture, Structure, and Process*; Oxford University Press: Oxford, UK, 2012.
34. Borgatti, S.P.; Everett, M.G.; Johnson, J.C.; Agneessens, F. *Analyzing Social Networks Using R*; Sage: Hemet, CA, USA, 2022.
35. Huth, M.; Ryan, M. *Logic in Computer Science: Modelling and Reasoning About Systems*; Cambridge University Press: Cambridge, UK, 2004.
36. Hoare, C.A.R. An axiomatic basis for computer programming. *Commun. ACM* **1969**, *12*, 576–580. [CrossRef]
37. Pierce, B.C. *Types and Programming Languages*; MIT Press: Hoboken, NJ, USA, 2002.
38. Boole, G. *The Mathematical Analysis of Logic*; CreateSpace Independent Publishing Platform: North Charleston, SC, USA, 1847.
39. Boole, G. *An Investigation of the Laws of Thought: On Which Are Founded the Mathematical Theories of Logic and Probabilities*; Walton and Maberly: London, UK, 1854; Volume 2.
40. Turing, A.M. On computable numbers, with an application to the entscheidungsproblem. *J. Math* **1936**, *58*, 5.
41. McCulloch, W.S.; Pitts, W. A logical calculus of the ideas immanent in nervous activity. *Bull. Math. Biophys.* **1943**, *5*, 115–133. [CrossRef]
42. Winskel, G. *The Formal Semantics of Programming Languages: An Introduction*; MIT Press: Hoboken, NJ, USA, 1993.
43. Rogers, H., Jr. *Theory of Recursive Functions and Effective Computability*; MIT Press: Hoboken, NJ, USA, 1987.
44. Montague, R. The proper treatment of quantification in ordinary english. In *Approaches to Natural Language: Proceedings of the 1970 Stanford Workshop on Grammar and Semantics*; Springer: Berlin/Heidelberg, Germany, 1973; pp. 221–242.
45. Bennett, M. A variation and extension of a montague fragment of english. In *Montague Grammar*; Elsevier: Amsterdam, The Netherlands, 1976; pp. 119–163.

46. Cooper, R. *Quantification and Syntactic Theory*; Springer Science & Business Media: Berlin, Germany 2013; Volume 21.
47. Andrews, P.B. *An Introduction to Mathematical Logic and Type Theory: To Truth Through Proof: Vol 27*; Springer: Dordrecht, The Netherlands, 2002.