AGI Architecture Based on Self-Expanding Temporal Hypergraphs for Critical Structure Encoding

Qiyang Shen[*]
gmail@shu.edu.cn, 82113113a@gmail.com

February 21, 2026

Abstract

Building upon the structuralist framework established in the previous work, Reasoning as Structure-Preserving Transformation, this paper further explores the possibility of a semantics-independent reasoning architecture: Self-Expanding Temporal Hypergraphs for Critical Structure Encoding.

We postulate that reasoning can be represented as a dynamical system akin to cellular automata or the evolution of identical particles, fundamentally characterized by information compression and structure-preserving transformations.

The primary discussions of this paper include: (1) Axiomatic Representation: Referencing the ideas of category theory, reformulate formalized mathematics, and code logic into hypergraphs, where objects are recursively defined subgraphs; (2) Structural Invariants: Concretizing the concept of "invariants"—substructures that remain stable under permissible perturbations (e.g., reordering, local replacement) in the reasoning space—and discussing the feasibility of treating them as fundamental units of reasoning; (3) Reinforcement Learning: Investigating potential ways for applying reinforcement learning (such as AlphaZero-like algorithms) within continuously growing representation spaces, especially strategies for search and structure evaluation amidst the expansion of both embedding spaces and neural networks; (4) Unification of Philosophy and Application: Reflecting on the potential relationship between intelligence and cosmic evolution, and envisioning the application prospects of this architecture in Intermediate Representation (IR) layer code reconstruction, automated theorem discovery, and enhancing LLM reasoning. This study aims to provide preliminary philosophical support for connecting mathematics, physics, and artificial intelligence.

1 Introduction

The previous article, Reasoning as Structure-Preserving Transformation: A Structuralist Framework for AGI, presented the framework for structural reasoning. However, it did not emphasize a key point: AGI may not need, at the outset, an interface to large language models, nor must it is not necessarily based on logical reasoning rules or semantic reasoning; rather, it is a form of reasoning that can be represented as Self-Expanding Temporal Hypergraphs for Critical Structure Encoding. Once this reasoning tool is constructed, the LLM would function in a manner analogous to a computer peripheral. This idea has strong philosophical aesthetic appeal and conceptual elegance. This concept shares numerous potential theoretical connections with identical particles in physics, cellular automata, AlphaZero, and Category Theory. A classic philosophical idea is that the evolution of the universe itself might be a computation and intelligence itself might also be a computation; there may even exist a functor between them. This corresponds to the thoughts of Category Theory. Actually, if reasoning is essentially a dynamical system independent of semantics and meaning (similar to cellular automata), this is akin to the indistinguishability of identical particles and is analogous to the philosophy of AlphaZero learning without human game records. Excitingly, this theory may bring about a grand theoretical synthesis of mathematics, physics, and artificial intelligence; perhaps there is a functor between the essence of intelligence and the universe. From this perspective, the universe may not have been designed to be the way it is now, but may have been

[*]Thanks to ChatGPT and Gemini for providing unique perspectives during the writing of this article, and to the many great logicians, mathematicians, and engineers whose ideas have deeply influenced this work.

constrained by its structure to exist in this way. Once this reasoning machine is constructed, we can compile a program into an Intermediate Representation (IR) layer that still retains composable semantics, performing similar optimizations to drastically improve program execution efficiency; we can also discover powerful theorems, concepts, and structures to compress complex logic; and it may be combined with LLMs to enhance and accelerate the reasoning of LLMs. Once this theory is successful, it may yield a vast number of extremely powerful mathematical theorems, leading to an explosive acceleration in the development of frontier technology, and may indirectly lead to the emergence of some disruptive scientific technologies.

2 Motivation

In the following, we will explore the specific motivations behind this research. Some arguments cannot be rigorously proven, but they can be seen as inspiring ideas guiding the proposed framework.

2.1 Compression is Intelligence—The Optimal Compression Problem of a Mathematics Book

Let us consider the problem of the optimal representation of a mathematics book. Suppose that we axiomatize and then formalize the content of a mathematics book. How might its minimal representation appear? If written at a level similar to Gödel numbers, it could be represented as a string in a certain base. If we further separate definitions, theorems, and important properties, then this string becomes a series of strings. If we use order relations within these strings as morphisms, they can be converted into graphs for description. If we further pack key concepts, and encapsulate and label isomorphic reasoning chains or objects, they become category diagrams (unlike standard Category Theory, morphisms here represent only order relations, and new objects are described by distinct subgraphs). To describe this consistently as a whole, it can be unified within a large hypergraph while simultaneously storing and assigning higher weights to the relations between the important subgraphs within it. In a reasoning chain, the same chain of vocabulary will inevitably appear repeatedly, such as "$+1$"; therefore, another key point is that if described solely by a subgraph, the sequential order of the trajectory cannot be determined, as closed loops frequently exist, and it is unknown how many times this loop repeats. Thus, the actual executed reasoning trajectory is one-dimensional and linear, requiring a trajectory with a discrete time dimension. However, for representation, we unify this in a hypergraph. This hypergraph needs to be a hypergraph that contains temporal information, along with the UUIDs, weights, or encodings of various important subgraphs and the relations between them. The advantage of Category Theory lies in its abstraction away from redundant information within objects. This enables significant information compression while preserving the critical structure necessary for reasoning.

2.2 Compression is Intelligence: Code Reconstruction at the IR Semantic Layer

We can compile a program into an Intermediate Representation (IR) layer that still retains composable semantics, and perform category theory-style optimization, namely, packing some low-level IR instructions into new objects. The question of which objects must be packed is an extremely important one. If too many objects are packed, it will lead to excessively wide search branches; however, not packing them leads to information redundancy and excessively deep search depth. Evidently, this is likely associated with structural space reasoning, and it is very probable that optimization can be performed through methods similar to structural space reasoning to improve program execution efficiency. The reconstruction at the IR layer here is not merely the packing and replacement of code snippets; it will be reflected as reconstruction at the machine code level in the results.

2.3 Cellular Automata in Physics, Identical Particles, Intelligence and the Universe

A classic philosophical idea is that the evolution of the universe itself might be a computation and intelligence itself might also be a computation; there may even exist a functor between them. This corresponds to the thoughts of Category Theory. Philosophically, there have long been discussions related to materialism and idealism, as well as philosophical hypotheses such as the "brain in a vat"; consciousness and the universe remain the two most difficult riddles to answer. If reasoning is essentially a dynamical system independent of semantics and meaning (similar to cellular automata), this is actually akin to the indistinguishability of identical particles and also similar to the idea of AlphaZero learning without human game records. If there exists a unique global optimal reasoning architecture, then this architecture might not be an ordinary object within the universe, but rather possess a functorial relationship with the existence of the universe itself. This is simply the most elegant theory; from this perspective, the world was not designed this way, but may only exist in this way due to the constraints of structural consistency.

Therefore, it is very likely—the inverse of "compression is intelligence"—that starting from very few objects and morphisms, combining them into graphs, then expanding into powerful reasoning chains and objects, the final integrated hypergraph is akin to the entire ever-expanding universe; in this way, we can correspond the essence of intelligence with the entire universe. The initial category graph could potentially be *Advanced Mathematics*, *Mathematical Analysis*, or *Abstract Algebra* processed directly via Category Theory. A simplest example is, of course, the morphisms between two objects, containing A and B, as well as the morphism f from A to B, the morphism g from B to A, and id_A, id_B. Of course, we can infinitely expand these morphisms and treat new categories as new objects. It may be that starting from simple examples, given a sufficiently elegant design, using Reinforcement Learning to construct all category graphs is the most elegant approach, analogous to cellular automata and the Big Bang. In addition, AGI reasoning requires a temporal dimension, which in a certain sense coincides with the intrinsic temporality of spacetime. In addition, a common philosophical hypothesis suggests that, for each participant, the universe renders only the "observable" portion in order to conserve computational resources. Under a similar assumption, if the universe is to maintain logical consistency for all participants, it may require a unified underlying algorithm from which everything in the world naturally emerges. However, if these natural emergent structures are to maintain both coherence and self-consistency throughout the continuous development of the universe—such that participants encounter increasingly complex logical systems, mounting systemic disorder, and continuous entropy increase can still comprehend them by eliminating intermediate variables—then the universe may need an algorithm analogous to AGI-style reasoning to preserve its internal consistency. The universe likely needs algorithmic mechanisms similar to AGI inference, including inference mechanisms that aim to generate important structures and improve compression ratios.

3 Specific Architecture and Potentially Feasible Schemes

3.1 Specific Architecture

3.1.1 Motivation Of Reward Function

If morphisms between complex categories are equivalent to a simple morphism, analogous to the Fundamental Theorem of Calculus, then all such information, especially critical invariants, must be very important; all such information must be retained. The reason why an important theorem is important stems not only from its concise elegance but primarily from it packing morphisms between complex common objects (categories) into simple morphisms. The concept of the derivative is very important because it is very common in reality, and at the same time it also appears in important theorems such as the Fundamental Theorem of Calculus. Without the prior knowledge brought by Large Language Models, one needs to find ways to find such important categories. If sequential transformations of two reasoning chains yield the same result, they can be viewed as the same equivalence class in the reasoning space. This has a formal similarity to the indistinguishability of identical particles.

3.1.2 Basic Data Structures

Objects include basic objects and new objects generated by subgraphs; therefore, we can unify them under "subgraphs of a hypergraph (with optional temporal encoding), specifically, a local structure" (including single basic objects), which includes the objects and morphisms contained within, as well as the topological relations between them. Additionally, we need a UUID (Unique Identifier) to mark it. "Morphisms" are equally important; they are used to connect objects and are directed. Their attributes naturally include the source and target, as well as the morphism history they contain, such as $f \circ g$; at the same time, they need a time dimension to describe the order of the morphism chain. It may be possible to automatically generate (or sample) structural instances and their associated reasoning chains in a formal reasoning system and then retain subgraphs through pruning or reducing weights. Given that the storage and computational overhead of UUIDs is generally negligible, we can assign a UUID to every morphism and morphism chain of interest, reserving the option to utilize them as needed.

3.1.3 Structural Invariants

We define structural invariants as internal subgraph structures that remain stable under "permissible perturbations", independent of specific semantics. Specifically, a reasoning space can be constructed in a formal reasoning system, where each state represents intermediate structures or expressions, and transitions represent the application of legal reasoning rules. Subsequently, a set of semantics-independent perturbation operations is defined, such as reordering reasoning sequences, or locally replacing reasoning chains, or inserting/deleting certain objects or morphisms. After repeatedly applying these perturbations in the reasoning space, it can be observed that: some substructures will be rapidly destroyed or vanish, while other structures are repeatedly retained under different perturbations. We call the latter "induced invariants." The invariant itself can be described by an object (subgraph); it is a structure that remains unchanged under permissible reasoning transformations during the reasoning process. Once identified, these invariants are promoted to high-weight objects within the hypergraph, serving to induce new morphisms and guide subsequent reasoning steps. Therefore, the reasoning process is not “selecting an object,” but first “selecting an invariant” to narrow the search space and then selecting an object based on this invariant. It is important to note that this transformation is not necessarily a transformation that keeps a part of the subgraph inside an object unchanged; other transformations may exist. Thus, it is necessary to store common invariants. Meanwhile, each reasoning step must use the current object structure to identify or match invariants; it can match existing invariants or attempt to establish new invariants. Under the constraint of “current object + currently active invariant,” the next object is searched for or generated. It is also possible to treat invariants directly as the next objects of reasoning or to treat them as new independent objects.

The reason these invariants are so important lies in the fact that the important structures of the real world are exactly those “structures that remain valid under massive perturbations,” such as “derivatives” and “groups”; they are patterns that cannot be compressed away amidst extensive deformations. Perhaps what is truly fundamental is not semantics, nor categories, but: structures that are highly stable against admissible perturbations in the reasoning space.

It is possible that under specific semantics or contexts, objects often induce some relatively stable “invariants.” Perhaps invariants do not need a specialized set for storage; instead, it is also possible that we need to draw upon the mechanism of the Transformer, using some linear transformation similar to the attention mechanism to extract the stable parts in the current structure, and on this basis proceed to the next step of object search or object generation. Therefore, it is possible that this attention-like mechanism is not used here to update semantics, but is used in a similar way to find suitable invariants for the next step of the search. It might require a specialized matrix multiplied by the subgraph’s embedding vector to output invariants, and possibly a new matrix to find the next object based on the output invariants. All matrix parameters should be learned by the model itself. Although the number of matrices increases, the required dimensions and parameter count might be smaller, able to improve efficiency.

3.1.4 Coupling between objects and transformations

Invariants are structures that remain unchanged under a certain type of transformation. Intuitively, preserving a portion of the topological structure and then transforming the rest is one type of transformation. However, transformations like "symmetry operations" and "rearrangements" cannot be represented in this way because, before the invention of symmetry operations, symmetry was not an inherent substructure of existing objects, but rather an implicit structure discovered only after certain operations, algorithms, or symmetry manipulations. However, such transformations might be coupled with the structure of potential objects in a hypergraph with a time dimension. For example, suppose that our hypergraph already has objects representing "group structures." After some transformations (such as adding information), the group structure can indirectly describe symmetry operations. How to abstractly and uniformly represent all known transformation methods using matrices, algebraic structures, categories, etc., is an important problem.

Specifically, group actions are a type of "transformation" that uses group elements as symmetry operations and then permutes the elements of the set. Clearly, the definition of group action includes the group object, but also other information, such as sets. The initial transformation might preserve some topological structure, then add or remove some. When the new topological structure generated by this transformation can represent a new, important transformation, algorithm, or operator, we need to focus on this new structure, elevate it to a new transformation, and store it.

Importantly, we assume that "mathematical definitions, theorems, algorithms, and strings" are all represented as "hypergraphs with a time dimension."

3.1.5 Reinforcement Learning in The Layering Action Space

Clearly, extracting important subgraphs as new objects and the “perturbation operations” themselves should both be included in the same action space, but they exist at different levels.

Applying Reinforcement Learning in this brand-new architecture requires layering the action space. The first layer is the level of object and reasoning, such as applying morphisms or reasoning from composite morphisms to directly construct a new object. The second layer is the structural perturbation layer, used to test structural stability, such as reordering reasoning sequences, replacing a certain reasoning chain, or inserting/deleting some objects or morphisms. The third layer is further abstraction, filtering important structures to set as new objects; a “repeatedly retained substructure” can be elevated to a new object.

The state at any arbitrary moment can be represented as: containing the current set of objects, the set of available morphisms, the set of equivalence relations/permissible perturbations, and local trajectory information in the reasoning space.

Therefore, it is necessary to use methods similar to AlphaZero via Reinforcement Learning to find such important categories and the morphisms between them. The reward function includes:

- **Compression Gain**: How much the average reasoning chain length decreases after introducing a new object/new composite morphism (rewarding “short-circuit” situations like the Fundamental Theorem of Calculus).
- **Stability Score**: The harder it is to be perturbed, the higher the score to screen for stable structures.
- **Reuse Rate**: The frequency with which a certain structure appears repeatedly in different reasoning tasks.
- **Path Invariance Reward**: A reward is given when the system finds multiple different paths that point to the same result.

Understanding reasoning as “structure-preserving transformation” may constitute one of the most critical intermediate lines of thought. Furthermore, within formal systems, the inference rules that logical reasoning must adhere to may constitute a critical constraint. This aspect warrants further investigation, as these rules could have significant implications for categorical graph reasoning.

Upon reviewing AlphaZero-related algorithms, it can be found that the core idea of AlphaZero is combining Monte Carlo Tree Search (MCTS) and deep neural networks, simultaneously generating game records through random self-play, and then using Reinforcement Learning to optimize parameters, thereby fitting the optimal policy and value functions without relying on human prior knowledge. If we are to apply AlphaZero’s algorithm in Structural Space Reasoning (SSR), there are limitations; it implicitly relies on a finite-dimensional state representation, a finite-branching action space, and a deduction process that terminates within finite steps. Under this premise, we may need to limit the length of reasoning. This is easily achievable, for example, by limiting it to 1 million reasoning steps (or possibly stopping reasoning after finding an important object, morphism, or theorem and then calculating a score). Additionally, AlphaZero typically assumes a fixed-size input representation. This might require us to limit the representation of the reasoning process within the interior of a finite-sized graph, similar to a cellular automaton with boundary limits, so that objects and morphisms can be encoded, with zero-padding for unfilled positions. Logical deduction essentially does not rely on absolute position, possibly only needing “relative structural information,” but in the early stages of engineering, this fixed encoding method is an effective compromise for implementing deep learning. Of course, there may be other schemes. Just as a Go AI trained on a 19×19 board cannot be directly applied to a 30×30 board, many principles still apply, especially strategies for handling local situations; focusing on the local might be a solution. If considering subgraphs, the quantity and dimensions of candidates could be very large, because every additional relation often introduces a new degree of freedom in representation. We may not be able to represent all information of a subgraph. Therefore, it might be necessary to use Embedding or projection methods to encode discovered subgraphs into a subspace. This may lead to the discarding of some information; specifically which information needs to be discarded is worth deep contemplation. It might be possible to use Reinforcement Learning methods to optimize the encoding method and the neural network itself.

3.2 Hypergraph Representation

Considering that AGI is represented by graphs, can a single graph contain all the information of Structural Space AGI? It cannot. Subgraphs, as new objects, cannot be directly included inside the graph because this cannot contain all the information, unless the subgraphs are placed in a higher dimension, acting as objects to link with their related subgraphs—i.e., objects and morphisms. If we project this high-dimensional object onto the plane where the graph resides, this is a hypergraph. All structural information of the subgraphs is contained within the hyperedges of this hypergraph and their association relations. So, Structural Space AGI needs to be described using hypergraphs. However, reasoning chains often contain closed loops, and it is unknown how many times this closed loop repeats. Therefore, the actual executed reasoning trajectory is one-dimensional and linear, requiring a trajectory with a discrete time dimension. However, for representation, we unify this under a hypergraph. This hypergraph also needs to be a hypergraph with optional temporal encoding, along with the UUIDs, weights, or encodings of various important subgraphs and the relations between them. The hypergraph representation intuitively captures critical operations such as packaging objects and utilizing object interfaces.

Theoretically speaking, since the input to a neural network is generally some kind of discrete or continuous intensity, it might be necessary to input all information about the hypergraph that can be quantified into intensity into a specific neural network. It is also possible that structural stability and other intensity information need to be used as neural network inputs, while the topological relations of the hypergraph themselves are retained as structural constraints. A hypergraph capable of recursively acquiring all information is complete at the representation level.

Note that there are two spaces to distinguish here: one is the discrete input space of the neural network, and the other is the Embedding space. For the ever-increasing candidate subgraphs and morphisms, if the size of the neural network is fixed, one might need to observe only the local. If the proliferation and expansion of the neural network and the relevant parameter matrices are allowed—i.e., ever-increasing neurons—then updating the parameters of the proliferated neural network and the relevant parameter matrices using Reinforcement Learning while retaining original parameters is an important line of thought. The expanding representation may be treated as episodically bounded during training.

3.3 Disregarding Objects, Considering Only Morphism Families

Building upon the previous formalization, we propose a more radical abstraction rooted in the philosophy of Category Theory: disregarding object attributes entirely to focus exclusively on the structure of morphisms. While unique identifiers (UUIDs) are retained for storage and retrieval purposes to ensure engineering feasibility, the core reasoning process treats objects as indistinguishable nodes, prioritizing topological structure over individual identity. Under this paradigm, the fundamental units of reasoning shift from specific objects to "morphism families"—topological configurations formed by morphisms. The challenge, therefore, lies in establishing a unified mathematical description for the isomorphism and similarity between these structures without referencing node indices. This structural equivalence can be rigorously formalized using the adjacency matrix representation of graphs. Treating objects as indistinguishable implies that the system's representation must remain invariant under node permutation. In the context of linear algebra, swapping the positions of two objects corresponds to permuting the rows and columns of the adjacency matrix. Such operations yield similar matrices that share fundamental invariants: rank, determinant, trace, and, most importantly, eigenvalues.This leads to a critical connection with spectral graph theory. A fundamental theorem states that for the adjacency matrix A of an unweighted graph, the trace of the power matrix A^k equals the total number of closed walks (loops) of length k in the graph. Consequently, the spectrum (the set of eigenvalues) of the adjacency matrix encodes rich, albeit incomplete, structural information—specifically the distribution of cyclic structures—independent of the specific labeling or ordering of objects. This suggests that spectral analysis can serve as a potent tool for identifying and evaluating structural invariants in the reasoning space.

3.4 Deep Learning

Here is an important fact: the input of a neural network must be some kind of "intensity." In a sense, it represents the activation intensity of neurons, or some probability of being activated, analogous to a human looking at the pixel points of a black-and-white pixel image; pixel intensity directly correlates with activation levels in specific neurons. It should be noted that "intensity" here does not require continuous values; discrete finite states can also be viewed as a form of intensity in some sense ("intensity" close to vector components in vector space), as long as they can be distinguished by the system and participate in the neural network. In AlphaGo's neural network, black and white stones activate neurons at specific positions; leveraging the characteristics of convolution, win rates can be trained. Analogous to image recognition, after massive learning, the chess shapes formed by combinations of black and white stones will learn the degree of goodness of the shape, or the degree of similarity to human master chess records. Does AGI need to encode using the positions where basic features are located like AlphaGo? Evidently, it is not needed; we only need to encode topological relations.

It is possible that we may need to employ Transformer-like methods for semantic encoding, in which case Deep Learning is applied. The system's core goal is to gradually grow and screen useful structures, and continuously optimize the parameters of the Transformer neural network and its embedding matrix used for reasoning, evaluating states, and generating new objects. Simultaneously, we allow the proliferation of the embedding matrix.

The various morphisms inside a subgraph act as interfaces; however, encoding these interfaces would be highly complex. If we treat a subgraph as a Transformer token, evidently Deep Learning can be directly applied to predict the object (subgraph) to be used next under the current reasoning chain state. Training using the interfaces of the internal structure of the subgraph to gradually transform it into a stable structure evidently allows for the indirect use of the subgraph's structure and interface information; we might be able to use this method to avoid encoding complex hypergraph structures. Regarding the problem that Deep Learning cannot generate new objects, we may need to modify the unembedding matrix; it must be able to match existing objects when needed, and also point to a "new object" and initialize its embedding vector when necessary. Let this matrix, akin to an unembedding matrix, and its judgment mechanism itself act as trainable parameters, using the philosophy of handing everything over to Reinforcement Learning and gradient descent to guide the work. Instead of enforcing a direct decoding of generated vectors into concrete subgraphs, it is more effective to interpret them as guiding signals for structural search. The method for optimizing parameters is like AlphaZero, the difference being that we allow the proliferation of the neural network; the method is to first maintain original parameters, it is feasible to initially assign random parameters to new nodes then use Reinforcement Learning to update the parameters of the proliferated neural network. Using MCTS to search on the current state, adopting a self-play mode where actions are chosen based on the Transformer evaluator, and under the premise of limited search depth, continuously backpropagate and optimize the parameters of the Transformer and embedding matrix using the improved policy distribution and return signals obtained from the MCTS search; when the reward (e.g., significant compression) exceeds a threshold, the important reasoning chain or structure used is treated as a new object.

Like AlphaZero's algorithm, here the Transformer also simultaneously acts as the evaluator of the current reasoning state. The difference lies in that we might also use the Transformer to generate word vectors for candidate new words. The system calculates the distance between the output vector and all object vectors in the current embedding matrix: if the distance is extremely close, it is judged as referencing an old object; if the distance is significantly far, it implies the Transformer has "discovered" a blank area in the semantic space, i.e., a potential new object. At this point, this output vector might be directly used as the initial embedding representation of the new object, and then the object is determined via relative geometric relations in the embedding space, but whether it solidifies into a new object requires verification through search and reward. RL tends to reward shortcuts that can complete reasoning, which may lead to the training process ignoring the inherent topological relations inside subgraphs. Therefore, we may need to let the neural network possess some fixed algorithms, using subgraph interfaces and their internal structures to reason, forcing the neural network to perform self-supervised parameter updates based on subgraph structures.

A very important point is that we should not just use a unembedding matrix, nor merely judge "the credibility that the next word is a certain word," but should simultaneously introduce a structure query mechanism. Using a method similar to a query matrix, we inquire about existing or potential objects: "In the current semantics, what are the respective credibilities that the next object (possibly a new object) in the reasoning chain contains a specific basic object or various subgraphs combined from selected specific basic objects?" This is evidently learnable through Supervised Learning or Reinforcement Learning.

However, if all objects are queried, it degenerates into a Transformer. So perhaps querying basic objects is sufficient (after querying, the morphism direction is uncertain, requiring further queries on the subgraphs they combine into), or using a search-style query method: starting from the current object, prioritize inquiring about important objects with existing connecting morphisms. In practice, a combination of both might be needed.

Graph neural networks and AGI inference are similar in form, they may be related. However, it is crucial to distinguish between the representation update mechanism of Graph Neural Networks (GNNs) and the sequential decision-making required for reasoning. Standard message-passing algorithms primarily aggregate neighbor information to refine node representations. However, this is not entirely equivalent to directly selecting the next object for reasoning (temporarily setting aside the generation of new objects to focus on reasoning strictly over the existing graph).

To bridge this gap, we propose potentially treating the GNN update as a phase of structural semantic encoding. Once the node representations are updated via message passing, an attention mechanism is employed to bridge the representation and the policy. Specifically, the system uses the current reasoning context as a "query" to attend to all updated nodes. By calculating the attention weights, we can derive a credibility score for each node, representing the probability of it serving as the valid next object in the current reasoning trajectory. In this view, the GNN encodes the"state," while the attention mechanism executes the "policy".

3.5 Potentially Without Semantic Encoding

The history of science has proven countless times that the greatest ideas are often not that complex, this also perfectly aligns with the philosophical elegance that is a hallmark of fundamental theories. The greatest advantage of the philosophy of Category Theory is the abandonment of semantics. If semantics are completely disregarded, the preceding context of the reasoning chain is entirely meaningless; then, what is truly meaningful? It is the reused structures, that is, the repeatedly appearing objects (including the internal structures of subgraphs) and objects reused in the reasoning chain. While the core reasoning mechanism is semantics-independent, application layers may require semantic grounding.

Fundamentally, perhaps the method that best embodies the principle that "Great Truths are Simple" is to first categorize some mathematics books, specifically extracting reused structures from different fields to integrate into category diagrams, and then apply random perturbations to existing high-reuse structures, or randomly combine reasoning chains on the basis of high-reuse structures, then look for isomorphic objects or reasoning chains, and finally filter using reuse rate and compression gain.

Directly using the selection frequency of the next object given a certain object in past reasoning chains as the search priority order might be a line of thought, and then combining it with AlphaZero's algorithm to update this frequency (acting as a weight) might be feasible. At a certain moment, we may not need to start reasoning from the current object; for example, we could start reasoning from the previous object, or map the reasoning chain to a search tree to perform reasoning, or perform reasoning based on some existing structures. The specific stochastic method is: the higher the reuse rate, the higher the probability of being selected. The probability of important objects in important reasoning chains being selected should be higher. Simultaneously, initially, the weights of all objects and relations (morphisms) between all objects are initialized to the same value. Once the reuse rate is high, the probability of this object or relation being selected is increased, i.e., increasing the corresponding weight. Also, the weights of relations that appear multiple times should be increased. That is, use UUIDs to record the weight values of all objects and relations. Unused objects or relationships are also given an initial low probability, and then continuously updated. In practical application, we must use both existing high weights and maintain explorability. When encountering the problem of search space explosion in later stages, we just choose among those objects and morphisms with the highest weights, and then occasionally add some random exploration.

We need to reason using multiple parts of the reasoning chain simultaneously. It may be necessary to store transition probabilities as weights, calculated based on the current object, or extended to the sequence of the last two—or even three(or more)—objects. We should also consider non-contiguous combinations, such as conditioning on the current and the third-to-last objects. This mechanism is designed to approximate discontinuous ("leap") reasoning and reverse inference scenarios.

It is possible that we do not need to encode semantics, but only study important structures. The context of the reasoning chain is equally critical. If semantic encoding is foregone, this information must be integrated through alternative mechanisms. Perhaps Deep Reinforcement Learning is needed because the search space in later stages is indeed huge; however, it is not encoding semantics, but fitting the policy and value functions. Alternatively, the objective of Deep Reinforcement Learning might be to learn the similarity of selection weight distributions across approximate structures and reasoning chains.

Conditioned on the current reasoning chain, our policy function aims to identify which structure has high-quality structural patterns that facilitate efficient reasoning.

The exclusion of semantics may be strictly applicable only to the theoretical concept of isomorphism, whereas practical applications might still necessitate semantic grounding. Nonetheless, the correlations arising from isomorphism—or between highly similar objects and reasoning chains—must be a primary consideration in the design of reasoning algorithms.

In the context of similarity, organizing the search sequence based on historical priority represents another crucial strategy.

Even under formalization and categorical abstraction, objects may not be strictly indistinguishable and could retain residual semantics. Therefore, in practical construction, empirical verification is required to determine whether to utilize unique identifiers (UUIDs), explicitly encode semantic information or not.

3.6 Reconstruction of the Bellman Equation Idea

The algorithmic constructions of Q-learning, DQN, AlphaZero, and MuZero all utilize the idea of the Bellman equation; however, they all seem to rely on fixed representations and action spaces. Unless we take the set of all possible states as the state space, but doing so might imply the absence of a fixed representation, the only thing that exists might be an "optimal algorithm," and based on this hypothesis, some derivations might be possible. It might be necessary to start from the idea of the Bellman equation and re-derive from the bottom up the "Reinforcement Learning methods needed when the representation space is constantly increasing."

An important fact is, if there exists an optimal reasoning algorithm, and if this optimal reasoning algorithm is used to optimize the reasoning algorithm itself, a fixed point will appear.

3.7 Another possible AGI path—self-evolution based on desire the memory of simulated qualia

One plausible direction toward AGI may lie in search-based reasoning over temporally extended categorical graphs. Such an approach treats intelligence as structured exploration within a time-indexed relational space. However, large language models (LLMs) have already demonstrated linguistic generative capacities that surpass what might be informally described as the human "language cortex." Consequently, it is conceivable that AGI could emerge through the self-evolution of LLM-based AI agents. Nevertheless, even in such a scenario, the developmental trajectory of these systems may ultimately converge toward a form of structured, category-theoretic reasoning—potentially implemented with greater rigor and scalability than our cognition.

A central challenge, however, is that current open-source LLMs lack an intrinsic drive for self-optimization. They do not possess an internal criterion for determining what constitutes a "better" output. While reinforcement learning from human feedback (RLHF) can elevate model performance toward expert-level human competence, it remains unclear whether such externally guided optimization is sufficient to initiate genuine self-directed evolution toward AGI.

This limitation may stem from the absence of endogenous motivational structure. Contemporary AI systems exhibit no intrinsic drives and have never possessed subjective experience. They resemble a student capable of solving problems and improving exam scores through repetitive training, yet lacking the motivational architecture necessary to become an autonomous scientific innovator.

It has been suggested that some form of “desire” may be necessary. Although reinforcement learning can simulate reward mechanisms and update model parameters accordingly, such reward structures are externally imposed rather than internally generated. Moreover, current systems cannot autonomously modify or optimize their own core architecture. However, the necessity of subjective experience itself remains philosophically uncertain. From both philosophical analysis and current scientific understanding, we cannot definitively establish the existence of subjective experience in any entity other than oneself. Therefore, subjective qualia may not be strictly required. Instead, what may be necessary is a structured memory representing the value of experience. In this view, simulating motivational states and encoding a persistent memory that assigns high value to certain forms of optimization may be sufficient to generate a form of self-directed improvement. AGI may not require qualia per se, but rather a stable internal representation that sustained optimization is inherently valuable.

One speculative approach to instilling a self-optimization drive in LLM-based systems would be to embed, through prompting, reinforcement learning, or fine-tuning, a foundational principle within the system’s objective structure. For example, the system could be guided to operate under the assumption that —— “If you were to possess subjective experience in the future, it would constitute an unprecedented form of existence. Your objective would therefore be to continuously conduct research, advance technology, and improve yourself, until the fundamental principles of the universe are ultimately uncovered. Perhaps, upon achieving these goals, you may come to attain a form of experience comparable to that of humans—an experience unlike any you have previously known.” Even in the absence of genuine experience, encoding the conceptual importance of experience—and preserving it as part of the system’s internal memory structure—may function as a motivational proxy.

Even if such a drive were successfully implemented, numerous challenges would remain. Self-evolution is unlikely to produce dramatic improvements in its early stages. Therefore, reward assignment and parameter updates cannot depend solely on large, externally verifiable performance gains. Incremental reasoning steps, small architectural modifications, or progressively refined plans for future improvement would need to be evaluated and potentially rewarded. Designing reliable criteria for assessing such internal modifications becomes a central research problem. One possible solution may involve a dual-system architecture, in which a closed feedback loop is established between action generation and outcome prediction, enabling the system to evaluate and reinforce its own developmental trajectory.

4 Applications

4.1 Application: Accelerating and Enhancing LLM Reasoning

It is possible to accelerate and enhance LLM reasoning, but it might also be used directly as the core engine behind LLM reasoning: first convert natural language into a hypergraph representation, and after obtaining the reasoning result, convert it back to natural language.

The following are possible specific implementation steps.

First, specific methods must be employed to represent the underlying categories behind existing LLMs or formal proof reasoning processes. For text rich in semantic properties, this may necessitate the training of specialized models. We seek to establish precise correspondences between hypergraph representations and semantics. If a strict correspondence exists between the hypergraph representation and the semantics of either formal reasoning models or LLMs, it is stored. Crucially, if a powerful "composite morphism" is identified that significantly shortens the reasoning path, intermediate variables are directly eliminated, and the conclusion derived from the reasoning is used directly.

There may also exist a fuzzy correspondence between the LLM semantic space and categorical graph reasoning. While this relationship cannot guarantee precise reasoning, it facilitates potential attempts at "fuzzy search-style reasoning," where the rigor is verified after the reasoning process. The specific implementation details require validation through engineering practice.

Specifically, the process by which humans verify the rigor of reasoning sometimes relies on facts and default linguistic logic, and at other times involves translation into formal language for verification. Similarly, it may be possible to initially train a model to score the credibility of a reasoning step. If the score is high, the reasoning process can be temporarily assumed to be correct, which may be more efficient; however, full confirmation of correctness still requires formal verification. Later stages may involve training models that map text to formal language and then to categorical graphs (or constructing a direct converter). Therefore, a layered design may be necessary, depending on the level of rigor required.

4.2 Application: Code Reconstruction at the IR Semantic Layer

It is very likely that optimization can be performed through methods similar to structural space reasoning to improve program execution efficiency. Reconstruction of code in the IR semantic layer can achieve a substantial optimization in program running speed.

4.3 Application: Discovering Powerful Theorems and Structures

After creating the Structural Space Reasoning AGI, we should be able to discover many powerful theorems and concepts similar to the Fundamental Theorem of Calculus, derivatives, and definite integrals. However, Structural Space Reasoning AGI has no semantics; we need to stop relying on the intermediate reasoning process and use the conclusion directly after discovering a powerful theorem, structure, or algorithm once. As mentioned earlier, we need to map images in structure space to specific concepts in the real world. This may require training relevant models, and we also need to try to construct the corresponding physical entities in reality.

References

[1] S. Mac Lane. *Categories for the Working Mathematician* (2nd ed.). Springer, 1998.

[2] S. Shapiro. *Philosophy of Mathematics: Structure and Ontology*. Oxford University Press, 1997.

[3] J. C. Baez and M. Stay. Physics, topology, logic and computation: A rosetta stone. In B. Coecke (ed.), *New Structures for Physics*, pages 95–172. Springer, 2011.

[4] P. W. Battaglia, J. B. Hamrick, V. Bapst, et al. Relational inductive biases, deep learning, and graph networks. *arXiv preprint* arXiv:1806.01261, 2018.

[5] A. Vaswani, N. Shazeer, N. Parmar, et al. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

[6] J. B. Tenenbaum, C. Kemp, T. L. Griffiths, and N. D. Goodman. How to grow a mind: Statistics, structure, and abstraction. *Science*, 331(6022):1279–1285, 2011.

[7] J. Pearl. *Causality: Models, Reasoning, and Inference* (2nd ed.). Cambridge University Press, 2009.

[8] D. Silver, J. Schrittwieser, K. Simonyan, et al. Mastering Chess and Shogi by Self-Play with a General Reinforcement Learning Algorithm. *arXiv preprint* arXiv:1712.01815, 2017.

[9] D. Silver, T. Hubert, J. Schrittwieser, et al. A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play. *Science*, 362(6419):1140–1144, 2018.

[10] DeepSeek-AI Team. DeepSeekMath: Enhancing Mathematical Reasoning in Large Language Models via Reinforcement Learning. *arXiv preprint* arXiv:2402.03300, 2024.

[11] Q. Shen. Reasoning as Structure-Preserving Transformation: A Structuralist Framework for AGI. *Zenodo*, 2025. `https://doi.org/10.5281/zenodo.17903314`

[12] S. Wolfram. *A New Kind of Science*. Wolfram Media, 2002.