

Proposal: Towards A Group Field-Theoretic Computation Machine - A Relation-Ontological Computation Scheme

Wanhong HUANG

January 14, 2026

Abstract

We propose a novel computational paradigm grounded in Group Field Theory (GFT), representing a fundamental departure from entity-based architectures like the von Neumann model. Instead of discrete memory addresses and symbolic manipulation, the *Group Field-Theoretic Computation Machine* (GFT-CM) performs computation through the continuous evolution of field configurations over group manifolds. We formally prove that GFT-CM is Turing-complete and recognizes exactly the class of recursively enumerable (RE) languages, establishing its universality as a model of computation. This relational ontology—where computation emerges from field interactions rather than entity manipulation—offers theoretical advantages for problems naturally formulated in terms of symmetries, topological structures, and collective phenomena. We present rigorous mathematical foundations, including explicit construction of Turing machine simulation, complexity-theoretic analysis, and connections to analog computation. As a first step toward physical realization, we outline a simulation framework that demonstrates the conceptual feasibility of field-theoretic computation, examining compilation strategies, expressivity bounds, and connections to quantum gravity, topological quantum computing, and neuromorphic architectures.

1 Introduction

Within the frameworks of structuralism and process philosophy, computation can be understood as a process by which structures generate structures. Here, "structure" refers to objects that can be operated upon by formal tools, while "computation" is a concept at the ontological level. In formal systems such as mathematics, a typical manifestation of this process takes the form $y = f(x)$: where x and y can be viewed as structured objects, and f specifies the generative rule from one structure to another, itself being a structure.

When we ask "how do structures arise?", we inevitably engage with underlying ontological commitments. Two fundamentally different orientations can be distinguished: One is substantial ontology, in which structures are conceived as composed of bounded objects, with relations as supplementary properties between objects; the other is relational ontology, in which relations and constraints themselves are regarded as primitive, and so-called "entities" are phenomena that emerge from relations.

Because substantial ontology aligns more naturally with human cognitive habits, the vast majority of contemporary computing mechanisms and architectures implicitly adopt this stance. Whether in the classical von Neumann architecture or in recently emerged computing paradigms such as processing-in-memory and neuromorphic computing, the basic operational units remain, to a large extent, distinguishable and addressable physical or logical entities (such as memory cells, neurons, or devices). In these systems, logical entities serve as primitives for structure construction, and relations can only be indirectly represented through connection structures or parameterized coupling, rather than appearing as native computational objects.

The substantial ontological paradigm of computation remains superior to relational ontological models in terms of engineering complexity, efficiency, and feasibility at the current stage of scientific development, and its dominance in applications is unshakeable. However, the motivation for this work does not stem from specific engineering bottlenecks in existing computational paradigms, but rather from deep philosophical inquiry into the nature of computation itself.

Since the 20th century, structuralism, post-structuralism, feminist theory, and cutting-edge theoretical physics frameworks such as quantum field theory have, in different ways, revealed the profound insight of relational thinking: entities do not exist prior to relations; on the contrary, the network of relations constitutes what we understand as "reality" itself. The picture revealed by quantum field theory in particular suggests that relational ontology provides an attractive path for understanding the generation of complex structures. Although this orientation is intuitively counterintuitive, it still possesses high theoretical guiding significance in characterizing structure generation and evolution, and may provide new conceptual inspiration for developing computational paradigms closer to the phenomena of human intelligence.

However, computational theory, as a core tool of formalized thinking, has long been confined within the framework of substantial ontology. This asymmetry prompts us to raise the following questions: If we thoroughly implement relational ontology in the formal definition of

computation, can we gain a deeper understanding of computational and intelligent phenomena? Can this understanding provide new conceptual resources for future computational paradigms? This work is an exploratory attempt to address these questions.

Within the framework of quantum field theory, Group Field Theory (GFT), as a cutting-edge variant, provides a theoretical framework for describing discrete and continuous spacetime structures from a relational ontological perspective. In group field theory, the ontological foundation is the algebraic structure of groups, which can in some sense be viewed as purely relational ontology. Specifically, the definition of relations constitutes the core part of the definition of groups (i.e., the ontology of GFT): for any $g, h \in G$, their composition $gh \in G$ is specified by the group operation, while g and h themselves are not presupposed to be any entities. At the level of group definition, they are merely formal symbols introduced to establish relational structure, and before the concrete instantiation of the group, they can in theory correspond to any objects we desire. From this meanings, the group field theory utilizes the group as primitives make it hold a relational ontology.

This work attempts to propose a computational model based on relational ontology (GFT-CM), founded on group field theory. Specifically, this work revolves around the following five core questions: (1) How to formally define a computational machine model using the GFT framework; (2) Whether the computational power of this model achieves at least Turing completeness; (3) How to simulate the computational process of GFT-CM in classical environments; (4) Whether potential physical implementation pathways exist for this model; (5) What insights GFT-CM offers for computational theory and the philosophy of artificial intelligence.

Unlike existing non-traditional computational paradigms such as quantum computing, GFT-CM directly takes relational structures as the ontological foundation of computation. Although quantum computing exploits quantum superposition and entanglement, its operational units (qubits) remain identifiable entities; while topological quantum computing is closer to operations on relational structures—where topological properties largely determine the computational process—at the ontological level, its basic topological configurations (such as anyonic braiding patterns) still presuppose distinguishable topological defects or quasiparticles as entities. In contrast, GFT-CM exploits the relational ontology of group field theory, making entities themselves structures that emerge from relations: relationality and symmetry serve as the ontological foundation, while entities and their complex structures are derivative phenomena of these relations.

It should be made clear that the primary value of this work lies in theoretical exploration and conceptual innovation, rather than immediate engineering applications. The primary value of GFT-CM lies in its ability to place grammatical structure, constraint satisfaction, and physical field dynamics within a single formal system in which these notions are mutually translatable. This makes it possible to treat symbolic generation, relational consistency, and physical relaxation as different representations of the same underlying computational process, enabling problems that are separately treated in logic, optimization, and physics to be analyzed within a unified computational framework.

As a foundational study, its core contributions are manifested in three aspects: First, this work is among the first to introduce the group field theory framework from quantum gravity into computational theory, providing a concrete formal model for a relational-ontological computational paradigm, and thereby expanding our understanding of the fundamental question “what is computation.” Second, through a proof of Turing completeness, it demonstrates that relational-ontological computation can in principle realize any computable function in classical computation. By mapping the proposed model to quantum and topological quantum computational schemes at the level of simulation, we further show that GFT-CM is not weaker in computational capacity, in particular with respect to formal-language expressiveness (e.g., recursively enumerable languages), than traditional symbolic and quantum computation models. Third, although physical implementation faces tremendous challenges at the current stage, the formal framework of this model still provides theoretical preparation for potentially emerging relational computing hardware in the future. This work should be viewed as a philosophical speculation and formalization attempt regarding the nature of computation.

The structure of this paper is as follows: Section 2 discusses the ontological foundations of computation and the ontological framework of structure generation from a philosophical perspective; Section 3 introduces the basic concepts and mathematical structures of group field theory; Section 4 formally proposes the computational machine model based on group field theory (GFT-CM); Section 5 demonstrates through concrete cases how GFT-CM executes computational processes; Section 6 presents a preliminary simulation framework; Section 7 discusses the philosophical significance, practical value, and possible physical implementation pathways of GFT-CM; Section 8 summarizes this work and looks forward to future research directions.

2 The Ontology of Computation

The question of what computation is ultimately a question of what kinds of entities, relations, and processes are taken to be fundamental in a theory of information and structure. In this respect, debates in the philosophy of computation parallel long-standing discussions in metaphysics and the philosophy of science concerning the ontological status of objects, relations, and processes.

One influential tradition, often associated with substantival or object-centered metaphysics, holds that the world is fundamentally composed of individual entities that possess properties and stand in relations to one another. Within this view, relations are secondary: they depend on the existence of the entities they connect. This perspective underlies much of classical physics, as well as most conventional models of computation, in which discrete states, symbols, or variables are taken as the basic units of representation and manipulation.

In contrast, relational and process-oriented philosophies challenge the primacy of self-subsistent entities. From early relational thinkers such as Leibniz, through process philosophers like Whitehead, to contemporary structuralist and post-structuralist traditions, a common theme is that relations, interactions, and patterns of organization are ontologically prior to the things that appear to participate in them. On this view, entities are not fundamental building blocks, but rather relatively stable configurations within networks of relations or processes.

This relational orientation has found particularly strong support in modern theoretical physics. In quantum field theory, particles are understood as excitations of underlying fields, and in many approaches to quantum gravity, spacetime itself is treated as an emergent structure arising from more primitive relational degrees of freedom. Group field theory is a prominent example of such an approach, in which geometric and physical structures are generated from algebraic and combinatorial relations defined over groups.

Within the philosophy of computation, however, relational ontology has been far less influential. Most formal models of computation, from Turing machines to contemporary digital and neural architectures, implicitly adopt an object-centered view in which computation consists of operations on discrete, identifiable units. Even when global patterns or emergent behaviors are studied, they are typically analyzed as arising from interactions among these units, rather than being treated as primitive.

The present work adopts a relational ontological stance toward computation. It explores the

consequences of taking relations, symmetries, and constraints—rather than symbols, variables, or devices—as the fundamental constituents of computational systems. Rather than committing to any single philosophical school, this stance is motivated by the observation that many of the phenomena associated with intelligence, language, and complex systems are more naturally described in relational terms, and that contemporary physics provides formal tools, such as group field theory, for representing such relational structures in a precise and generative way.

2.1 Philosophical Discussion

2.2 Entity Ontology

In an entity-centered ontology, the basic constituents of a computational system are discrete, identifiable units that possess states. Computation is understood as the evolution or transformation of these states according to specified rules. Relations between entities—such as connections, interactions, or dependencies—are treated as secondary structures that describe how entities influence one another, but do not themselves constitute the primary objects of the theory.

This ontological stance is implicit in most classical and modern computational paradigms. In the Turing machine, the tape is divided into cells, each of which stores a symbol; computation proceeds by reading and writing symbols at particular locations. In von Neumann architectures, memory cells and registers are the fundamental units, and operations are performed by addressing and manipulating their contents. In artificial neural networks and neuromorphic systems, neurons and synapses serve as the basic components, each with an internal state or weight. Even in quantum computing, the qubit is defined as a localized degree of freedom whose state is manipulated by unitary gates and measured individually.

Within this paradigm, relations appear only through explicit data structures or parameters: pointers in memory, edges in a graph, synaptic weights, coupling matrices, or interaction terms. Although such relations may be complex, they are always represented as attributes of or links between pre-existing entities. As a result, computation is fundamentally organized around the manipulation of object-like units, and relational structures must be encoded indirectly within these object-based frameworks.

This ontology has proven extraordinarily effective for engineering purposes. It supports modularity, addressability, and precise control, making it well suited to digital implementation and large-scale system design. However, it also imposes a particular conceptual constraint: any

relational or global structure must ultimately be reduced to configurations of local entities and their pairwise or parameterized interactions.

2.3 Relational Ontology

In a relational ontology, relations, constraints, or symmetries are taken as the primitive constituents of a system. What are ordinarily called “entities” are not fundamental, but arise as stable or recurrent patterns within a network of relations. From this perspective, the identity and persistence of an object are not presupposed, but emerge from the way relations are organized and maintained.

When applied to computation, a relational ontology implies that the basic computational state is not a collection of independent variables, but a structured configuration of relations. Computation is then understood as the transformation, propagation, or relaxation of these relations under some dynamical or algebraic law. Rather than asking how the states of individual units change, one asks how patterns of consistency, compatibility, or symmetry evolve over time.

This perspective aligns naturally with several domains that are difficult to capture within entity-centered frameworks. Formal grammars, for example, define well-formed expressions through systems of production rules and constraints, not through intrinsic properties of symbols. Constraint satisfaction problems are specified by relations among variables, with solutions corresponding to globally consistent assignments. In physics, field theories describe systems in terms of spatially and temporally extended relations, with particles or excitations appearing as localized patterns in an underlying field.

Under a relational ontology, such phenomena do not require translation into collections of objects with externally imposed links. Instead, they are natively expressed as configurations of interacting relations. A computational model based on this ontology therefore treats grammar, constraint satisfaction, and physical dynamics as different manifestations of a common relational structure.

3 Ontology of Computational Schemes

Although contemporary computational paradigms differ widely in their mathematical formalisms and physical implementations, they share a common ontological commitment: computation is organized around collections of identifiable units whose states are manipulated by explicitly

specified operations. This commitment persists even in models that depart significantly from classical digital logic.

In the classical Turing-machine model, computation is defined over a tape consisting of discrete cells, each holding a symbol from a finite alphabet. The machine's internal state and the contents of the currently addressed cell together determine the next operation. Although the Turing machine can generate arbitrarily complex symbolic structures, its ontology remains that of addressable storage locations and symbol-bearing entities.

Von Neumann architectures generalize this idea by separating memory and processing units. Here, computation proceeds by fetching data from particular memory addresses, transforming it in registers, and writing results back to memory. All higher-level abstractions—data structures, graphs, programs, or databases—are ultimately reduced to configurations of bits stored in and retrieved from these addressable locations.

Artificial neural networks and neuromorphic computing introduce dynamics and distributed representation, but do not fundamentally alter this ontological stance. Neurons, synapses, and weights are treated as discrete components, each with a state that can be updated. Even when networks exhibit emergent behavior, learning, or self-organization, these phenomena arise from interactions among explicitly represented units. Relations remain encoded as parameters or connections between entities rather than as primary computational objects.

Energy-based models such as Ising machines and Hopfield networks come closer to a relational perspective by defining computation as the minimization of a global energy function over interacting variables. Nevertheless, the ontology remains grounded in collections of spins or nodes, each of which carries a local state. The global constraint is realized through couplings among these variables, but the variables themselves are still taken as primitive.

Quantum computing, despite its radical departure from classical information processing, retains a similar entity-centered structure. Qubits are treated as localized degrees of freedom that can be individually addressed, manipulated by unitary gates, and measured. Entanglement introduces non-classical correlations, but the formalism still presupposes a tensor-product decomposition into distinguishable subsystems. Topological quantum computing replaces qubits with more exotic objects such as anyons or braiding patterns, yet these too are modeled as identifiable excitations or defects whose configurations encode information.

A related class of approaches is physical and reservoir computing, in which a complex physical system—such as a fluid, optical medium, or nonlinear electronic circuit—is used as a com-

putational substrate. These systems exploit the rich dynamics of physical fields, which at a fundamental level are governed by relational interactions rather than by isolated entities. In this sense, physical reservoirs do operate on substrates that are ontologically closer to relational fields than to digital registers. However, at the level of computational representation, reservoir computing remains fundamentally entity-based. The reservoir state is treated as a vector of observable quantities, and computation proceeds by mapping input vectors to output vectors through externally defined readout functions. The internal relational structure of the physical medium—its couplings, symmetries, and interaction topology—is not represented or manipulated as part of the computational model. Instead, it is used as a fixed, unstructured dynamical feature generator. Consequently, tasks involving grammar, symbolic composition, or constraint satisfaction are not encoded in the relational dynamics of the reservoir itself, but are imposed by training the readout layer to interpret reservoir states in a task-specific manner. The physical relations remain implicit in the substrate but are not formal objects of computation. This distinguishes reservoir computing from a relational-ontological computational model such as GFT-CM, in which relations and symmetries constitute the computational state and program. In GFT-CM, grammatical rules, consistency constraints, and generative structure are encoded directly as interaction terms within a group-valued field, rather than being imposed externally through supervised readout.

Across all these paradigms, relations and global structures are present and often crucial, but they appear only as properties of or couplings between pre-existing units. None of these frameworks takes relational structure itself as the fundamental carrier of computational state. As a result, symbolic expressions, grammatical well-formedness, or global consistency must be implemented indirectly, through elaborate encodings into collections of entity states and their interactions.

This observation does not diminish the power or practical success of existing computational schemes. Rather, it highlights a conceptual gap: while many of the phenomena associated with intelligence, language, and complex systems are most naturally described in terms of relations and constraints, contemporary computation models remain ontologically anchored in entities. The next sections explore how a group field-theoretic approach can offer an alternative foundation in which relational structure becomes the primary computational substrate.

3.1 Conclusion

The foregoing analysis highlights a persistent tension in contemporary theories of computation. On the one hand, many of the structures that appear central to intelligence, language, and complex systems—such as grammatical rules, global consistency constraints, and emergent patterns—are most naturally described in relational terms. On the other hand, dominant computational paradigms remain ontologically grounded in collections of discrete, addressable entities, with relations represented only indirectly through parameters, connections, or externally imposed structures.

This mismatch does not undermine the practical success of existing computing architectures, but it does limit their conceptual scope. When relations are not part of the computational ontology, phenomena that are fundamentally relational must be encoded in an ad hoc manner, often through large numbers of local variables and couplings. As a result, symbolic structure, grammatical organization, and global coherence appear as secondary or emergent features, rather than as native objects of computation.

At a deeper level, this tension reflects a broader ontological misalignment across scientific domains. Contemporary physics increasingly describes reality in terms of fields, symmetries, and relational structures, while cognition and language likewise appear to be organized by global constraints, grammatical relations, and emergent coherence. In contrast, most computational formalisms continue to rely on object-centered representations. This divergence suggests that existing computational models may be using ontological primitives that are poorly matched to the phenomena they are intended to model.

A relational-ontological approach offers a different perspective. If relations, symmetries, and constraints are taken as primitive, then entities and symbolic structures can be understood as stable configurations within a relational field. This shift aligns naturally with both modern physical theory and with domains such as formal language theory and constraint satisfaction, where well-formedness and meaning are defined by systems of relations rather than by intrinsic properties of isolated symbols.

The group field-theoretic computation machine proposed in this work is motivated by this convergence. By grounding computation in group-valued fields and their interactions, GFT-CM provides a formal framework in which grammatical generation, constraint satisfaction, and physical dynamics can be expressed and analyzed within a common relational structure. This framework does not seek to replace existing computational paradigms, but rather to complement

them by offering a way to study computation in regimes where relational organization, rather than object manipulation, is the dominant organizing principle.

4 Group Field-Theoretic Basis

4.1 Group Field Theory Formalism

In group field theory (GFT), the fundamental degrees of freedom are fields defined over a group manifold rather than over physical spacetime. Formally, one considers a field

$$\Phi : G^D \rightarrow \mathbb{C},$$

where G is a group and G^D denotes the D -fold Cartesian product of G with itself. In the standard GFT framework, the same group G is repeated D times, so that each field configuration is a function on G^D . More generally, the formalism allows G^D to be composed of different groups, corresponding to different types of symmetry degrees of freedom—a flexibility we exploit in modeling heterogeneous relational structures.

The groups G encode symmetry and relational structure. Common choices include Lie groups such as $U(k)$, $SO(k)$, $O(k)$, and $SU(k)$, which are widely used in physics to describe rotational, gauge, and internal symmetries. In the present work, we abstract from their specific physical interpretations and instead emphasize their role as algebraic structures governing the composition and compatibility of relational degrees of freedom. Each group element $g \in G$ can be understood as encoding a specific relational configuration, and the group multiplication law determines how such configurations compose.

The dynamics of a GFT is specified by an action functional $S[\Phi]$, which determines how the field configurations evolve and interact. In analogy with ordinary quantum field theory, the action is typically written as a sum of a free (kinetic) part and an interaction part:

$$S[\Phi] = S_0[\Phi] + S_{\text{int}}[\Phi].$$

The free part $S_0[\Phi]$ describes the propagation of elementary relational degrees of freedom and typically takes the form

$$S_0[\Phi] = \int_{G^D} dg_1 \cdots dg_D \bar{\Phi}(g_1, \dots, g_D) \mathcal{K}(g_1, \dots, g_D) \Phi(g_1, \dots, g_D),$$

where $\mathcal{K}$ is a kinetic operator encoding the dispersion relation or propagation structure, and dg_i denotes the Haar measure on G . The interaction part $S_{\text{int}}[\Phi]$ specifies how multiple field configurations are combined or constrained, typically through polynomial terms of the form

$$S_{\text{int}}[\Phi] = \sum_{n \geq 3} \frac{\lambda_n}{n!} \int_{G^{nD}} V_n(g_{11}, \dots, g_{nD}) \prod_{i=1}^n \Phi(g_{i1}, \dots, g_{iD}) \prod_{j=1}^{nD} dg_j,$$

where V_n are interaction kernels and λ_n are coupling constants.

In the conventional physical interpretation of GFT, perturbative expansions of the partition function generate combinatorial structures that can be interpreted as discrete spacetimes, and condensation mechanisms can give rise to effective continuum geometries. In this work, however, we set aside these gravitational interpretations and focus instead on the formal and computational meaning of the group field framework, particularly its capacity to represent and evolve complex relational networks.

4.2 Fock Space, Creation and Annihilation Operators

As in ordinary quantum field theory, group field theory is naturally formulated in a Fock space. Let $\mathcal{H}$ denote the single-field Hilbert space associated with one quantum of the group field. The full Fock space is then given by

$$\mathcal{F} = \bigoplus_{n=0}^{\infty} \mathcal{H}^{\otimes_s n} = \mathbb{C} \oplus \mathcal{H} \oplus \mathcal{H}^{\otimes_s 2} \oplus \dots \oplus \mathcal{H}^{\otimes_s n} \oplus \dots,$$

where $\otimes_s$ denotes the symmetrized tensor product (assuming bosonic statistics), allowing for superpositions of states containing arbitrary numbers of field quanta. The vacuum state $|0\rangle \in \mathbb{C}$ represents the absence of any relational structure.

Field quanta are created and annihilated by operators $\hat{\Phi}^\dagger$ and $\hat{\Phi}$, respectively. These operators satisfy the canonical commutation relations

$$[\hat{\Phi}(g_1, \dots, g_D), \hat{\Phi}^\dagger(g'_1, \dots, g'_D)] = \prod_{i=1}^D \delta_G(g_i, g'_i),$$

where $\delta_G(g, g')$ is the Dirac delta distribution on the group G with respect to the Haar measure. Acting on the vacuum state, the creation operator $\hat{\Phi}^\dagger$ generates an excitation of the group field:

$$\hat{\Phi}^\dagger(g_1, \dots, g_D)|0\rangle = |g_1, \dots, g_D\rangle,$$

while the annihilation operator $\hat{\Phi}$ removes such an excitation:

$$\hat{\Phi}(g_1, \dots, g_D)|0\rangle = 0.$$

In the physical interpretation of GFT, these excitations are often referred to as “particles” or “atoms of space,” but in the present computational interpretation they represent elementary relational units from which larger structures are composed.

A general state in the Fock space can be written as

$$|\Psi\rangle = \sum_{n=0}^{\infty} \frac{1}{n!} \int_{G^{nD}} \psi_n(g_{11}, \dots, g_{nD}) \prod_{i=1}^n \hat{\Phi}^\dagger(g_{i1}, \dots, g_{iD}) |0\rangle \prod_{j=1}^{nD} dg_j,$$

where ψ_n are symmetric wavefunctions representing the amplitude for n -body relational configurations. States in the Fock space can thus be understood as superpositions of many-body relational configurations, and the interaction terms in the action govern how these configurations can be created, combined, and transformed. This many-body formalism provides a natural setting for describing the emergence of complex relational structures from simple interaction rules.

4.3 Perturbative Expansion and Feynman Diagrams

The partition function of a GFT is given by the functional integral

$$Z = \int \mathcal{D}\Phi e^{-S[\Phi]/\hbar},$$

where $\mathcal{D}\Phi$ denotes the functional measure over field configurations. In the perturbative regime (small coupling constants λ_n), this can be expanded as

$$Z = Z_0 \sum_{k=0}^{\infty} \frac{(-1)^k}{k! \hbar^k} \langle S_{\text{int}}[\Phi]^k \rangle_0,$$

where $\langle \cdot \rangle_0$ denotes expectation values with respect to the free theory and Z_0 is the free partition function.

This perturbative expansion can be represented graphically through Feynman diagrams. In the GFT context, these diagrams have a distinctive combinatorial structure:

- Each vertex corresponds to an interaction term in S_{int} , with valence determined by the

degree of the polynomial;

- Each edge represents a propagator derived from the kinetic term S_0 ;
- The graph structure encodes how relational units are connected and constrained.

Importantly, GFT Feynman diagrams are dual to simplicial complexes or more general combinatorial structures, establishing a direct correspondence between perturbative quantum amplitudes and discrete geometric/topological configurations. In our computational interpretation, this correspondence suggests that the perturbative expansion generates a space of possible relational network topologies, with amplitudes weighting their relative significance.

4.4 Condensate States and Mean-Field Approximation

Beyond perturbative analysis, GFT admits non-perturbative phenomena such as condensation. A condensate state is characterized by macroscopic occupation of a single-particle mode:

$$|\Psi_{\text{cond}}\rangle = \exp\left(\int_{G^D} \sigma(g_1, \dots, g_D) \hat{\Phi}^\dagger(g_1, \dots, g_D) dg_1 \cdots dg_D\right) |0\rangle,$$

where $\sigma(g_1, \dots, g_D)$ is the condensate wavefunction. This coherent state satisfies

$$\hat{\Phi}(g_1, \dots, g_D) |\Psi_{\text{cond}}\rangle = \sigma(g_1, \dots, g_D) |\Psi_{\text{cond}}\rangle,$$

indicating that the field operator has a classical expectation value.

In the mean-field approximation, one replaces the field operator by its expectation value plus fluctuations:

$$\hat{\Phi} = \langle \hat{\Phi} \rangle + \delta \hat{\Phi} = \sigma + \delta \hat{\Phi},$$

and derives effective dynamics for σ by extremizing an effective action. This leads to a Gross-Pitaevskii-type equation governing the condensate wavefunction—a nonlinear equation whose solutions describe emergent collective behavior.

In our computational framework, condensation represents the emergence of dominant relational patterns from the collective dynamics of many elementary units, analogous to how macroscopic order emerges from microscopic interactions in statistical mechanics.

4.5 Relation to Tensor Networks and Quantum Information

The GFT formalism exhibits deep connections to tensor network representations prevalent in quantum information theory. The field $\Phi(g_1, \dots, g_D)$ can be viewed as a rank- D tensor with indices taking values in the group G . Interaction terms in the action correspond to contraction operations between tensors, and the perturbative diagrams correspond to tensor network diagrams.

This correspondence suggests that GFT provides a field-theoretic completion of tensor network methods, naturally incorporating:

- Fluctuations and quantum corrections beyond mean-field;
- Dynamical topology through interaction-generated graphs;
- Renormalization group flows capturing scale-dependent effective descriptions.

Moreover, entanglement entropy and other quantum information-theoretic measures can be computed within the GFT framework, potentially providing quantitative diagnostics for the complexity and coherence of emergent relational structures.

5 The Group Field–Theoretic Computation Machine (GFT-CM)

5.1 Formal Definition

We define a Group Field–Theoretic Computation Machine (GFT-CM) as a tuple

$$\mathcal{M} = (\mathcal{G}, \mathcal{H}, \mathcal{F}, \hat{H}, \mathcal{I}, \mathcal{R}),$$

whose components are defined as follows.

The component $\mathcal{G}$ is a (possibly product) group that specifies the symmetry and relational domain of the computation. This group serves as the fundamental algebraic structure over which relational configurations are defined, and its multiplication law determines how elementary relational units compose into more complex structures. The choice of $\mathcal{G}$ reflects the type of symmetries and constraints inherent in the computational problem under consideration.

The component $\mathcal{H} = L^2(\mathcal{G}^D)$ is the single-field Hilbert space of square-integrable functions over the group manifold $\mathcal{G}^D$. This Hilbert space provides the quantum mechanical framework for

describing individual field quanta, each representing an elementary relational configuration. The dimension D corresponds to the valence or arity of the relational structures being represented.

The component $\mathcal{F}$ is the associated Fock space

$$\mathcal{F} = \bigoplus_{n=0}^{\infty} \mathcal{H}^{\otimes_s n},$$

which represents the space of all possible multi-particle relational configurations. The direct sum structure allows for superpositions across different particle numbers, enabling the representation of variable-complexity relational structures within a single mathematical framework. The symmetrization (indicated by $\otimes_s$) ensures bosonic statistics, reflecting the indistinguishability of identical relational units.

The component $\hat{H}$ is a Hamiltonian operator acting on $\mathcal{F}$, derived from a group field action functional $S[\Phi]$ through the standard canonical quantization procedure. This Hamiltonian encodes the interaction rules, constraints, and generative structure of the computation, determining how relational configurations evolve and transform over time. The specific form of $\hat{H}$ thus embodies the computational logic or algorithmic content of the machine.

The component $\mathcal{I}$ is an input map that prepares an initial field state $|\psi_0\rangle \in \mathcal{F}$ corresponding to a given input instance. This map translates external data representations into the internal language of the group field, establishing the initial conditions for the subsequent dynamical evolution.

The component $\mathcal{R}$ is a readout map that extracts classical outputs from final or stable field states. This map bridges the quantum field-theoretic description back to classical information, converting the evolved field configuration into interpretable results such as parsed structures, constraint solutions, or generated relational patterns.

5.2 Computational States

A computational state of the GFT-CM is a vector $|\psi\rangle \in \mathcal{F}$. Each basis state in the Fock space represents a particular many-body relational configuration of the group field, corresponding to a specific arrangement of elementary relational units. Superpositions of such basis states encode ambiguity, parallel hypotheses, or probabilistic mixtures of relational structures, allowing the machine to simultaneously explore multiple computational paths or maintain uncertainty about the correct relational interpretation.

The Fock space structure naturally accommodates configurations of varying complexity, from simple states containing few relational units to elaborate states involving many interacting components. This flexibility enables the GFT-CM to represent both minimal and maximal information states within the same mathematical framework, without requiring explicit memory allocation or data structure management.

5.3 Data Representation

An input instance x , such as a symbolic string, a graph, or a relational structure, is encoded into the field by the input map

$$|\psi_0\rangle = \mathcal{I}(x).$$

This encoding is realized by applying creation operators $\hat{\Phi}^\dagger(g_1, \dots, g_D)$ to the vacuum state in order to generate excitations of the group field corresponding to elementary relational tokens. Each creation operator insertion produces a quantum of the field at a specific point on the group manifold, representing a basic relational unit with particular symmetry properties determined by the group elements $(g_1, \dots, g_D)$.

Annihilation operators $\hat{\Phi}(g_1, \dots, g_D)$ remove such excitations, enabling the representation of conditional structures or constraints that eliminate certain configurations from consideration. The commutation relations between creation and annihilation operators ensure that the resulting field states respect the appropriate quantum statistics and symmetry properties.

In this representation scheme, data are not stored in discrete memory locations or symbolic registers, but are instead embodied as field configurations—patterns of excitation distributed over the group manifold. This distributed representation naturally supports operations that involve global constraints, symmetry considerations, or structural transformations, as these can be implemented through field interactions rather than through explicit symbolic manipulation.

5.4 Computation as Field Evolution

Computation proceeds through the dynamical evolution of the field state under the Hamiltonian $\hat{H}$, governed by the Schrödinger equation

$$|\psi(t)\rangle = e^{-i\hat{H}t/\hbar}|\psi_0\rangle.$$

The Hamiltonian $\hat{H}$ is constructed from the group field action

$$S[\Phi] = S_0[\Phi] + S_{\text{int}}[\Phi],$$

where S_0 specifies the propagation or kinetic structure of relational degrees of freedom, determining how individual field quanta evolve in isolation, while S_{int} encodes interaction rules, grammatical constraints, or generative relations that govern how multiple relational units combine or interfere with one another.

The free Hamiltonian $\hat{H}_0$, derived from $S_0[\Phi]$, typically takes the form of a differential operator acting on the group manifold, implementing dispersion relations or energy-momentum relations for the field quanta. The interaction Hamiltonian $\hat{H}_{\text{int}}$, derived from $S_{\text{int}}[\Phi]$, involves products of creation and annihilation operators that create, destroy, or transform multi-particle configurations according to specified rules.

In this framework, the Hamiltonian plays the role of a program, encoding the computational logic not as a sequence of discrete instructions but as a continuous dynamical law. The time evolution of the field realizes the computational process, with the continuous parameter t serving as a generalized notion of computational time that need not correspond directly to physical time. Different choices of Hamiltonian correspond to different computational tasks or algorithmic procedures, and the same input state evolved under different Hamiltonians will generally produce different outputs.

The field-theoretic formulation naturally accommodates both deterministic and stochastic computational processes. Unitary evolution under a Hermitian Hamiltonian produces deterministic dynamics, while non-Hermitian or open-system extensions can model stochastic or dissipative computation. Furthermore, the possibility of condensation or phase transitions in the field theory suggests that qualitatively different computational regimes may be accessible depending on the strength of interactions or the initial state preparation.

5.5 Output and Halting

A computation terminates when the evolving field state reaches a stationary configuration, a stable attractor, or otherwise satisfies a designated measurement condition at time T . Stationarity can be defined in various ways depending on the computational task: it may correspond to an eigenstate of the Hamiltonian (zero time derivative), a minimum of an effective energy

functional (in dissipative or gradient-flow formulations), or satisfaction of a set of constraints encoded in the interaction structure.

The output is obtained by applying the readout map

$$y = \mathcal{R}(|\psi(T)\rangle),$$

which extracts classical information from the final or stable field configuration. The readout map typically involves projecting the field state onto specific measurement operators or computing expectation values of observables that encode the desired output quantities. For example, in a parsing task, $\mathcal{R}$ might measure correlation functions that determine the hierarchical structure of the input string, while in a constraint satisfaction problem, $\mathcal{R}$ might evaluate indicator functions that identify which constraints are satisfied by the final configuration.

The form of the readout map determines what aspects of the field configuration are considered relevant for the computational output. Different readout maps applied to the same final state can extract different types of information, reflecting the fact that the field configuration may contain richer structure than any single classical output can capture. This suggests a notion of computational richness or information content that goes beyond the classical input-output relation, encompassing the full quantum state as a resource for multiple distinct computational tasks.

In cases where the field evolution does not converge to a unique stable configuration, the readout map may be applied to an ensemble of final states or to time-averaged configurations, producing probabilistic or approximate outputs. This provides a natural mechanism for handling underconstrained problems or ambiguous inputs, where multiple outputs may be compatible with the given data. The quantum superposition structure of the Fock space ensures that such ambiguity can be represented coherently throughout the computation rather than requiring explicit enumeration of alternative interpretations.

6 Computation Case Study with GFT-CM

6.1 Natural Number Generation Through Field Condensation

As a first illustrative example, we consider the generation of the natural numbers within the GFT-CM framework. This example, though simple, demonstrates how the **field-theoretic**

structure and **group symmetry** actively participate in the computational process, distinguishing GFT-CM from ordinary quantum computing or symbolic manipulation.

6.1.1 Group Structure and Field Configuration

Let the group $\mathcal{G}$ be the cyclic group $\mathbb{Z}$ or, for finite approximation, $\mathbb{Z}_N$, with generator a satisfying $a^N = e$ (or no constraint for infinite $\mathbb{Z}$). We consider a group field

$$\Phi : \mathcal{G} \rightarrow \mathbb{C},$$

whose value $\Phi(g)$ at each group element $g = a^k$ represents the field amplitude at that "position" in the relational space. Unlike quantum computing where qubits occupy discrete registers, here the field is **distributed over the entire group manifold**, and computational states are characterized by how field amplitude is distributed across different group elements.

The symmetry of $\mathcal{G}$ imposes a natural ordering and composition structure: the group multiplication law $a^m \cdot a^n = a^{m+n}$ encodes the successor relation directly in the group structure. This is not merely a labeling convention but an intrinsic geometric property of the relational space over which the field is defined.

6.1.2 Field-Theoretic Representation of Number

A natural number n is not represented by a single Fock state, but rather by a **field configuration** - a distribution of field amplitude across the group manifold. Specifically, we consider a coherent state or field condensate of the form

$$|\psi_n\rangle = \exp\left(\sum_{k=0}^n \alpha_k \hat{\Phi}^\dagger(a^k)\right) |0\rangle,$$

where the amplitudes α_k determine how the field is distributed across the first $n + 1$ group elements. For instance, we might choose $\alpha_k = \sqrt{1/(n+1)}$ to create a uniform distribution, or $\alpha_k = e^{-\lambda k}$ for an exponentially decaying profile.

The crucial point is that the number n is encoded not in a particle count but in the **spatial extent** of the field configuration over the group manifold. The field "knows" about n through the geometric region it occupies in relational space.

6.1.3 Symmetry-Driven Field Evolution

The computational dynamics are governed by a Hamiltonian that respects and exploits the group structure. We define

$$\hat{H} = \hat{H}_{\text{kin}} + \hat{H}_{\text{int}},$$

where the kinetic term implements propagation along the group manifold:

$$\hat{H}_{\text{kin}} = - \sum_{g \in \mathcal{G}} \left[\hat{\Phi}^\dagger(ag) \hat{\Phi}(g) + \hat{\Phi}^\dagger(g) \hat{\Phi}(ag) \right],$$

and the interaction term creates correlations between neighboring positions:

$$\hat{H}_{\text{int}} = \lambda \sum_{g \in \mathcal{G}} \hat{\Phi}^\dagger(ag) \hat{\Phi}^\dagger(ag) \hat{\Phi}(g) \hat{\Phi}(g).$$

The kinetic term describes how field quanta propagate from g to ag (the successor in group structure), implementing a diffusion or flow of field amplitude along the natural ordering of $\mathcal{G}$. This is fundamentally different from quantum computing gate operations: rather than discrete state transitions, we have ****continuous field flow**** over the group manifold, with the group multiplication law determining the direction of flow.

The interaction term is crucial: it implements a ****many-body correlation**** whereby field configurations with high density at g tend to generate additional excitations at ag . This creates a self-reinforcing process where the presence of field amplitude at one location induces its own expansion to the next location. The strength λ controls the tendency toward condensation and spatial expansion.

6.1.4 Generation as Symmetry-Respecting Condensation

Starting from an initial state localized at the identity $e = a^0$:

$$|\psi_0\rangle = \hat{\Phi}^\dagger(e)|0\rangle,$$

the time evolution under $\hat{H}$ drives the field to spread across successive group elements. The kinetic term causes amplitude to propagate from a^k to a^{k+1} , while the interaction term amplifies this propagation when sufficient density is present.

After time t , the field configuration evolves to occupy an extended region:

$$|\psi(t)\rangle \approx \exp \left(\sum_{k=0}^{N(t)} \alpha_k(t) \hat{\Phi}^\dagger(a^k) \right) |0\rangle,$$

where $N(t)$ grows with time and the amplitude profile $\alpha_k(t)$ is determined by the interplay of kinetic propagation and interaction-driven growth.

The ****symmetry of $\mathcal{G}$ constrains the computation****: because the Hamiltonian respects the group structure (it commutes with group translations), the field can only evolve along directions compatible with the group multiplication law. The natural ordering $e, a, a^2, a^3, \dots$ emerges not from external programming but from the intrinsic geometry of the relational space.

6.1.5 Readout and the Role of the Field

To extract the number n represented by the evolved field state, we measure the ****spatial extent**** of the field configuration. This can be defined through the expectation value of a position operator:

$$\langle \hat{N} \rangle = \sum_{k=0}^{\infty} k \langle \hat{\Phi}^\dagger(a^k) \hat{\Phi}(a^k) \rangle,$$

which computes the "center of mass" of the field distribution over the group manifold. Alternatively, we might measure the maximal k such that the occupation $\langle \hat{\Phi}^\dagger(a^k) \hat{\Phi}(a^k) \rangle$ exceeds a threshold.

The field nature is essential here: we are not counting discrete particles in registers, but rather measuring a ****geometric property**** of a distributed configuration. The natural numbers emerge as quantized measures of how far the field has expanded along the symmetry-determined direction of the group manifold.

6.1.6 Contrast with Quantum Computing

This construction differs fundamentally from quantum computing in several respects:

****Spatial Distribution****: In quantum computing, n qubits occupy n distinct registers, but the registers themselves have no intrinsic relational structure. In GFT-CM, the field is distributed over a ****structured relational space**** (the group manifold), and the position in this space carries computational meaning.

****Symmetry as Constraint****: In quantum computing, symmetries are often used to simplify

calculations or reduce dimensionality, but they do not drive the computation. In GFT-CM, the group structure **actively determines** what configurations are accessible and how they evolve—the computation is intrinsically constrained by the geometry of the relational space.

Field vs. Particle: Quantum computing manipulates discrete quantum states. GFT-CM manipulates **field configurations**—coherent, spatially extended patterns of amplitude that can exhibit collective phenomena like condensation, interference across the manifold, and emergent ordering through interaction-driven phase transitions.

Many-Body Interactions: While quantum circuits use 2-qubit gates, the GFT-CM interaction term $\hat{H}_{\text{int}}$ implements genuine **many-body correlations** where the presence of multiple field quanta at one location collectively influences the generation of new quanta at neighboring locations. This enables computational processes that are not straightforwardly decomposable into sequences of local gates.

6.1.7 Symmetry Participation in Computation

The group $\mathcal{G}$ does not merely label states—it actively participates in the computation through three mechanisms:

Geometric Constraint: The group manifold structure determines what field configurations are possible. Only configurations that respect the group composition law are energetically favorable or dynamically stable.

Propagation Direction: The kinetic term $\hat{H}_{\text{kin}}$ implements flow along group multiplication, so the “direction” of computational evolution is dictated by how group elements compose. For different groups, the same Hamiltonian structure would produce qualitatively different computational dynamics.

Interaction Selection: The interaction term $\hat{H}_{\text{int}}$ couples field amplitudes at group-related positions (e.g., g and ag). The pattern of which positions interact is determined by the group structure, not chosen arbitrarily. Thus, the **symmetry determines the interaction graph**.

In summary, natural number generation in GFT-CM is realized not through symbol manipulation or gate sequences, but through **symmetry-constrained field condensation** over a structured relational space, where the group geometry and many-body field interactions work together to produce the emergent ordering.

6.2 Addition as Relational Field Consistency

We consider the computation of addition

$$a + b = c$$

within the Group Field–Theoretic Computation Machine. In contrast to circuit-based or gate-based models, addition is realized here as a relational constraint satisfied by a group-valued field.

Let G be an additive group, such as $\mathbb{Z}$ or a finite cyclic group $\mathbb{Z}_N$. We define a group field

$$\Phi : G \times G \times G \rightarrow \mathbb{C},$$

whose arguments $(x, y, z) \in G^3$ represent candidate relational configurations. The semantic interpretation of the field is

$$\Phi(x, y, z) \neq 0 \iff x + y = z.$$

That is, the field encodes which triples satisfy the addition relation.

The addition law is enforced by the action functional

$$S[\Phi] = \sum_{x, y, z \in G} |\Phi(x, y, z) - \delta(z - x - y)|^2,$$

where δ denotes the Kronecker delta on the group. This action is minimized precisely when

$$\Phi(x, y, z) = \delta(z - x - y),$$

so that the ground state of the field coincides with the group-theoretic addition table.

To compute the sum of two given inputs $a, b \in G$, we impose boundary conditions

$$x = a, \quad y = b,$$

which restrict the field to the slice $\Phi(a, b, z)$. The field then evolves according to the gradient flow induced by the action,

$$\frac{\partial \Phi}{\partial t} = -\frac{\delta S[\Phi]}{\delta \Phi},$$

relaxing toward a configuration that satisfies the relational constraint. In the steady state, the only nonzero component of this slice is

$$\Phi(a, b, z) = \delta(z - a - b),$$

which identifies the unique value $z = a + b$ consistent with the group law.

The output of the computation is obtained by the readout map $\mathcal{R}$, which extracts the value c for which $\Phi(a, b, c)$ is maximal. In this way, addition is realized as the emergence of a symmetry-consistent relational configuration of the group field, rather than as a sequence of symbol manipulations or gate operations.

7 Simulation Framework

8 Discussion

8.1 Expressiveness

8.1.1 Turing Completeness

A fundamental requirement for any proposed computational paradigm is that it not be weaker than classical models of computation. In the case of GFT-CM, symbolic computation arises as a special case of relational field dynamics. By encoding rewriting systems, grammars, or cellular automata as interaction terms in the group field action, it is possible to simulate any Turing-computable process within the relational field framework. This establishes Turing completeness as a lower bound on the expressive power of GFT-CM.

Importantly, this result should not be interpreted as the primary contribution of the model. Turing completeness merely guarantees that GFT-CM can represent classical computation. The distinctive feature of the model lies in its ability to represent and manipulate relational and grammatical structures natively, rather than through indirect encodings into symbol-based machines.

8.1.2 Transfer to Classical and Quantum Computation

The dynamics of a group field defined by a finite group and a polynomial action functional can be simulated, to arbitrary precision, on both classical digital computers and quantum computing

platforms. In this sense, GFT-CM is not proposed as a replacement for existing hardware, but as a higher-level computational theory whose dynamics can be interpreted and emulated by established architectures.

Such simulations, however, do not imply ontological equivalence. Classical and quantum machines simulate GFT-CM by discretizing or approximating relational field dynamics, just as they simulate fluid dynamics or neural fields. The relational ontology of GFT-CM remains distinct from the entity-centered ontology of the machines used to simulate it.

8.2 Physical Plausibility and Interpretive Mapping

Although GFT-CM is introduced primarily as a formal and conceptual model, its mathematical structure is closely aligned with contemporary physical theories. Group-valued fields, symmetry-constrained interactions, and many-body relational dynamics are central to quantum field theory and condensed matter physics. This suggests that GFT-CM is not physically implausible in principle.

Rather than proposing a specific hardware implementation, the present work emphasizes interpretive mappings between physical systems and relational field computation. Physical substrates that support symmetry-governed field dynamics, such as optical fields, spin systems, or cold atomic lattices, may be viewed as potential carriers of group field-like information processing. However, developing concrete devices based on these ideas remains a subject for future research.

8.3 Comparison to Existing Paradigms

Existing computational paradigms are largely grounded in entity-centered ontologies. Classical digital computers operate on addressable memory locations, neural networks on neurons and weights, Ising machines on spins, reservoir computers on dynamical states, and quantum computers on qubits or anyonic excitations. In all these cases, relations and global structure are represented indirectly, through interactions between primitive units.

In contrast, GFT-CM takes relational structure, symmetry, and constraint as its fundamental computational primitives. Entities and symbolic structures emerge as stable configurations of a relational field rather than as pre-specified carriers of information. This places GFT-CM in a distinct conceptual category, complementary to existing models rather than competing with them on performance or efficiency.

8.4 Implications for Cognition and Artificial Intelligence

Many phenomena associated with cognition and intelligence, including grammatical structure, perceptual coherence, and contextual meaning, are inherently relational. They are defined not by the intrinsic properties of isolated symbols or neurons, but by patterns of compatibility, constraint, and organization across a system.

By treating relations and symmetries as first-class computational objects, GFT-CM offers a natural formalism for modeling such phenomena. Cognitive processes can be interpreted as the evolution of relational fields toward stable, symmetry-consistent configurations, corresponding to percepts, interpretations, or decisions. This perspective suggests new ways of connecting physical dynamics, symbolic structure, and intelligent behavior within a unified computational framework.

8.5 Philosophical Implications

8.5.1 Linguistic Relativity in Computation

Different computational paradigms embody different implicit “languages of thought.” Classical architectures privilege procedural and sequential descriptions, neural networks privilege statistical and distributed representations, and symbolic systems privilege discrete syntactic structures. GFT-CM, by contrast, privileges relational and grammatical organization. In this sense, the choice of computational ontology shapes not only what can be computed efficiently, but also how structure and meaning are represented and interpreted.

8.5.2 Measurement and Collapse

In GFT-CM, computation proceeds through the relaxation of a relational field toward configurations that satisfy global constraints. This process is formally analogous to both physical measurement, in which a quantum system settles into an eigenstate, and grammatical disambiguation, in which multiple possible interpretations collapse into a single coherent structure. Measurement and meaning selection thus appear as manifestations of the same underlying relational dynamics.

8.5.3 Implications for Social Science

Social systems are fundamentally relational: individuals, institutions, and norms are defined by networks of interaction, role, and constraint. GFT-CM provides a formal language for representing such systems as relational fields governed by symmetry and interaction rules. This suggests a possible bridge between computational social science, field-theoretic modeling, and formal theories of structure, enabling social phenomena to be studied as dynamical processes of relational organization rather than as collections of isolated agents.

9 Conclusion

This work has proposed a group field-theoretic computation machine (GFT-CM) as a relational, field-based alternative to conventional models of computation. Motivated by an ontological mismatch between contemporary physical theory, cognitive and linguistic structure, and dominant computational paradigms, GFT-CM was introduced as a formal framework in which relations, symmetries, and constraints serve as the primary carriers of computational structure.

By grounding computation in group-valued fields and interaction functionals, GFT-CM provides a unified language for representing grammatical generation, constraint satisfaction, and dynamical evolution within a single formal system. Symbolic computation and classical algorithmic processes arise within this framework as special cases of relational field dynamics, ensuring that the model is not weaker than established computational theories while extending their expressive and conceptual scope.

A central feature of GFT-CM is that computation is not performed through the sequential manipulation of discrete entities, but through the emergence of symmetry-consistent relational configurations. In this sense, computation is realized as a process of field relaxation toward stable structures that satisfy globally defined constraints. This perspective aligns naturally with both modern physical theory, where fields and symmetries are fundamental, and with domains such as language, cognition, and social organization, where meaning and structure are inherently relational.

The present work is primarily conceptual and formal. While the dynamics of GFT-CM can be simulated on classical or quantum hardware, its principal contribution lies in providing a new ontological foundation for computation rather than a new hardware architecture. By offering a framework in which physical processes, symbolic structure, and relational organization are

expressed in a common formal language, GFT-CM opens a path toward deeper integration between computation, physics, and theories of intelligence.

Future work will explore richer classes of group structures, interaction functionals corresponding to more complex grammars and constraints, and concrete simulation frameworks for studying relational field computation in practice. Beyond its technical implications, the group field-theoretic perspective also invites further philosophical investigation into the nature of computation, representation, and meaning in a relational world.