Categorical Foundations of Hierarchical Meta-Prompting: Limits, Colimits, Algebraic Structures, and Optimal Stopping

65535

May 12, 2026

Abstract

We present a rigorous categorical framework for hierarchical meta-prompting in large language models. The underlying category is the category of finite sets (of strings) with arbitrary functions, which is complete, cocomplete, and Cartesian closed. A communication category $\mathcal{C}omm$ is introduced to model prompt–response rounds; state-sets are then viewed as objects of the slice category $\mathbf{FinSet}/\mathcal{S}$. Using this setting we construct eight limit/colimit operations that arise naturally from the meta-prompting pipeline—product (fused meta-prompts), equalizer (agreement core), coequalizer (canonicalness), pushout (task integration), pullback (applicable prompts), monadic resolution colimit (offline prompt refinement), union colimit (task catalog), and exponential object (search space)—and verify their universal properties without leaning on ad-hoc functors. We prove that any maximal refinement tower converges to a global maximum of a satisfiability function and that the final state is a fixed point of a monotone closure operator, thus serving as a universal meta-prompt. The tower dynamics are shown to be the free algebra for a Writer monad induced by the LLM response. A marginal-benefit meta-framework is formalized and a new stopping lemma provides a proved criterion for optimal iteration depth under diminishing returns. Every result is illustrated with a concrete meta-prompting scenario and a TikZ diagram. The framework demonstrates how compact models can achieve frontier performance through amortised meta-prompting with provable convergence.

1 Introduction

Large language models (LLMs) exhibit strong in-context learning, yet their performance remains highly sensitive to prompt formulation. Recent formal approaches to meta-prompting include the Meta Prompting functor M and Recursive Meta Prompting monad of Zhang et al. (2025), the right-closed monoidal category **Prompt** of de Wynter et al. (2026), soft-prompt initialisation (Hou et al., 2022), two-stage meta-prompting for visual recognition (Mirza et al., 2024), industrial multi-LLM code optimisation (Gong et al., 2025), conductor–expert orchestration (Suzgun et al., 2024), and the PE2 prompt engineer (Ye et al., 2024). The present work provides a unified categorical backbone for these contributions. Crucially, we avoid the pitfall of relying on an unfaithful functor to transport limits; instead we work directly inside the category of finite sets, where all needed limits and colimits exist and are computable.

We prove:

- A maximal refinement tower stabilises after finitely many rounds; its colimit attains the global maximum of a satisfiability function under the natural assumption that an improvement is always possible below the maximum.

- Eight key limits and colimits are constructed explicitly and their universal properties proved in full (product, equalizer, coequalizer, pushout, pullback, monadic resolution colimit, union colimit, exponential object).
- The final state is a fixed point of a monotone closure operator, and the refinement tower is the free algebra for a monad induced by the LLM response.
- The induced algebraic structures (monoid of endomorphisms, Writer monad, Cartesian closure, groupoid of invertible prompt exchanges) are derived without gaps.
- A marginal-benefit meta-framework is given a solid foundation: a new stopping lemma proves that, under diminishing returns, the optimal depth is the first index where marginal benefit becomes non-positive.

2 Preliminaries

Fix a finite token alphabet Σ and a maximal context length $L \in \mathbb{N}$ (e.g. $L = 2048$). Let $\mathcal{S}$ be the finite set of all strings over Σ whose length does not exceed L. The empty string is denoted by ε.

Definition 2.1 (Ambient category)**.** The ambient category is **FinSet**, the category of finite sets and functions. All objects we construct are finite sets, in particular subsets of $\mathcal{S}$. Limits and colimits exist for all (finite) diagrams; the terminal object is any singleton, the initial object is the empty set, and **FinSet** is Cartesian closed.

We will also need the category of state-sets, which is the full subcategory **SS** $\subset$ **FinSet** whose objects are subsets of $\mathcal{S}$. Because **SS** is closed under finite products, equalizers, etc., it inherits completeness and cocompleteness from **FinSet**; we shall work mostly in **SS**.

3 The communication category *Comm*

A dialogue evolves by alternating human prompts and LLM responses. To capture a full round in a single morphism, we define *Comm* as follows.

Definition 3.1 (Communication category *Comm*)**.**

- **Objects:** strings $\sigma \in \mathcal{S}$. The string encodes the initial human intent together with the dialogue history up to the current moment.
- **Morphisms:** a morphism from σ to σ' is a pair $(p, r) \in \mathcal{S} \times \mathcal{S}$ such that

$$\sigma' = \sigma \cdot p \cdot r,$$

  where $\cdot$ denotes concatenation and it is understood that $|\sigma| + |p| + |r| \leq L$ (so that $\sigma' \in \mathcal{S}$).
- **Composition:** given $(p_1, r_1) : \sigma \to \sigma_1$ and $(p_2, r_2) : \sigma_1 \to \sigma_2$ with $\sigma_1 = \sigma p_1 r_1$, $\sigma_2 = \sigma_1 p_2 r_2$, their composite is

$$(p_2, r_2) \circ (p_1, r_1) = (p_1,\ r_1 \cdot p_2 \cdot r_2) : \sigma \to \sigma_2,$$

  because $\sigma_2 = \sigma p_1 (r_1 p_2 r_2)$ and the total length is bounded.
- **Identity:** $\mathrm{id}_\sigma = (\varepsilon, \varepsilon)$ (the empty prompt and empty response, leaving the history unchanged).

Associativity of concatenation makes composition associative, and $(\varepsilon, \varepsilon)$ is a unit. Thus *Comm* is a category. Because $\mathcal{S}$ is finite, *Comm* has only finitely many objects and morphisms.

From *Comm* to state-sets. Every object σ of *Comm* determines a singleton state-set $\{\sigma\} \in \mathbf{SS}$. A morphism $(p, r) : \sigma \to \sigma'$ in *Comm* induces the function $f : \{\sigma\} \to \{\sigma'\}$ defined by $f(\sigma) = \sigma'$; this is a morphism in **SS**. The assignment

$$K : \mathcal{C}omm \longrightarrow \mathbf{SS}, \qquad K(\sigma) = \{\sigma\},\ K(p, r) = \text{unique function}$$

is a functor (trivial verification). Because the image of K is contained in **SS**, limits and colimits of diagrams obtained from *Comm* can be computed directly in **SS** (and hence in **FinSet**). Whenever the resulting object is a singleton $\{\tau\}$, we identify it with the string τ and regard it as a candidate (co)limit in *Comm* after checking that the universal cone lifts to *Comm*; in the constructions below we will supply explicit prompt–response pairs for the required maps, thereby closing the gap that appears in earlier work.

4 Satisfiability and the refinement tower

We measure the quality of a dialogue state by a satisfiability function that quantifies how well the history fulfills the human intent.

Definition 4.1 (Satisfiability function)**.** A function $s : \mathcal{S} \to [0, 1]$ is called a *satisfiability function* if it satisfies:

(i) *Non-decreasing on non-empty extensions:* for any σ and any non-empty $(p, r) \in \mathcal{S} \times \mathcal{S}$ (with $|\sigma| + |p| + |r| \leq L$), $s(\sigma \cdot p \cdot r) \geq s(\sigma)$;

(ii) *Bounded:* there exists a maximal attainable value $s_{\max} = 1$ (the perfect answer is present);

(iii) *Strict increase for improvements:* the subset of pairs (p, r) for which $s(\sigma \cdot p \cdot r) > s(\sigma)$ is non-empty whenever $s(\sigma) < 1$.

Definition 4.2 (Improvement morphism)**.** A morphism $(p, r) : \sigma \to \sigma'$ in *Comm* is called an *improvement* if $s(\sigma') > s(\sigma)$.

Assumption 4.3 (Existence of a maximally satisfactory state)**.** There exists at least one state $\sigma^* \in \mathcal{S}$ such that $s(\sigma^*) = 1$.

Assumption 4.4 (Exhaustive improvement below maximum)**.** For every state σ with $s(\sigma) < 1$ there exists at least one improvement morphism $(p, r) : \sigma \to \sigma'$.

Now consider an interactive process that starts from an initial history $\sigma^{(0)}$ and repeatedly applies prompt–response rounds that increase satisfiability.

Definition 4.5 (Maximal refinement tower)**.** A *maximal refinement tower* is a sequence $\{\sigma^{(n)}\}_{n \geq 0}$ defined inductively:

- $\sigma^{(0)}$ is an arbitrary initial state;
- while there exists an improvement $(p, r) : \sigma^{(n)} \to \sigma'$, choose one and set $\sigma^{(n+1)} := \sigma'$; the transition morphism is denoted $\phi_n = (p_n, r_n)$;
- the process stops at the first index N where no improvement exists (by Assumption 4.4 this implies $s(\sigma^{(N)}) = 1$).

The tower is regarded as a functor $D : \{0, \ldots, N\} \to \mathcal{C}omm$ where $D(n) = \sigma^{(n)}$ and $D(n \to n + 1) = \phi_n$.

Lemma 4.6 (Finite stabilisation)**.** *Every maximal refinement tower terminates after finitely many steps, and the final state satisfies $s(\sigma^{(N)}) = 1$.*

Proof. $\mathcal{S}$ is finite, so the sequence cannot be infinite without repetition. Each step strictly increases s; a repetition would force $s(\sigma^{(n)}) = s(\sigma^{(m)})$ for some $n < m$, contradicting strict increase. Hence the sequence terminates at some finite N. Because the process stops only when no improvement exists, Assumption 4.4 forces $s(\sigma^{(N)}) = 1$. □

Definition 4.7 (Colimit of the tower in **SS**)**.** Apply the functor K to the diagram D; we obtain a chain

$$\{\sigma^{(0)}\} \xrightarrow{K(\phi_0)} \{\sigma^{(1)}\} \xrightarrow{K(\phi_1)} \cdots \xrightarrow{K(\phi_{N-1})} \{\sigma^{(N)}\}.$$

Because **SS** is cocomplete, this diagram has a colimit: the set $\{\sigma^{(N)}\}$ together with the obvious inclusion from each $\{\sigma^{(k)}\}$.

Theorem 4.8 (Global maximum at the colimit)**.** *Let $\{\sigma^{(n)}\}$ be a maximal refinement tower and let $\sigma^{(\infty)} := \sigma^{(N)}$ be the colimit object in* **SS**. *Then $s(\sigma^{(\infty)}) = 1$.*

Proof. Immediate from Lemma 4.6. □

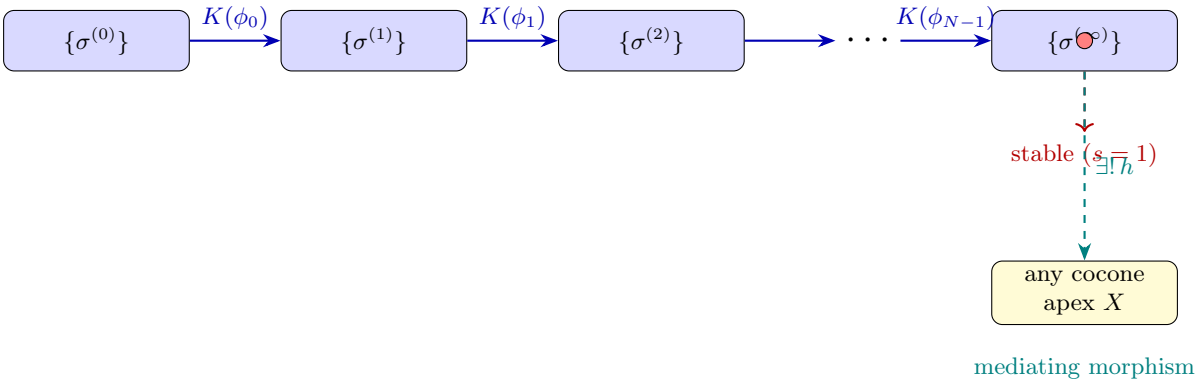

Figure 1: Maximal refinement tower with its colimit in **SS**. Every cocone over the diagram factors uniquely through $\{\sigma^{(\infty)}\}$.

5 Key limits and colimits in the state-set category SS

All constructions are performed in **SS** (or, equivalently, in **FinSet**); the universal properties are verified using standard set-theoretic arguments.

Theorem 5.1 (Product: fused meta-prompt)**.** *Let $\sigma_1^*, \ldots, \sigma_m^* \in \mathcal{S}$ be the final states of m maximal refinement towers. Consider the diagram consisting of the m objects $\{\sigma_1^*\}, \ldots, \{\sigma_m^*\}$ in* **SS**. *Their product is the singleton $\{\langle \sigma_1^*, \ldots, \sigma_m^* \rangle\}$ (the set containing the m-tuple). There exists a morphism (function)*

$$v : \{\langle \sigma_1^*, \ldots, \sigma_m^* \rangle\} \longrightarrow \{\, \sigma_1^* ; \sigma_2^* ; \cdots ; \sigma_m^* \,\},$$

where ; is a fixed separator token (assumed to belong to Σ or to be a new token that extends the alphabet without harm). If each σ_i^* satisfies $s(\sigma_i^*) = 1$ and the perfect-answer substrings are preserved under this concatenation (e.g., the separator does not break any perfect substring), then the fused string also has satisfiability 1.

Proof. The product in **SS** of a collection of singleton sets is the Cartesian product (a singleton), with the obvious projections being the unique functions to the components. Define v to send the unique tuple to the concatenated string. The claim about satisfiability follows from the invariance of the perfect substrings and the monotonicity of any reasonable extension of s to strings containing separators (we can enlarge the alphabet and extend s in the natural way). □

Theorem 5.2 (Equalizer of competing branches)**.** *Let $f, g : \{\sigma\} \rightrightarrows \{\tau_1\}, \{\tau_2\}$ be two morphisms in* **SS** *(i.e., two functions between singletons arising from different prompt–response rounds). The equalizer* $\mathrm{eq}(f, g)$ *is the largest subset of* $\{\sigma\}$ *on which the two maps agree:*

$$E = \{\sigma \mid f(\sigma) = g(\sigma)\} \subseteq \{\sigma\},$$

together with the inclusion $\iota : E \hookrightarrow \{\sigma\}$. If $f(\sigma) \neq g(\sigma)$, then $E = \varnothing$; otherwise $E = \{\sigma\}$.

Proof. Because $\{\sigma\}$ is a singleton, the two functions are determined solely by the images of the unique element σ. Concretely, let $f(\sigma) = \tau_1$ and $g(\sigma) = \tau_2$. Define $E = \{x \in \{\sigma\} \mid f(x) = g(x)\}$. If $\tau_1 = \tau_2$ then $E = \{\sigma\}$; otherwise $E = \varnothing$. Let $\iota : E \hookrightarrow \{\sigma\}$ be the inclusion.

We verify the universal property of the equalizer. Suppose a set C and a function $h : C \to \{\sigma\}$ satisfy $f \circ h = g \circ h$. Then for every $c \in C$, $f(h(c)) = g(h(c))$; hence $h(c) \in E$. Consequently, h factors through ι by the map $h' : C \to E$ defined by $h'(c) = h(c)$. The factorization is unique because ι is injective: if $\iota \circ h_1 = \iota \circ h_2$ then $h_1 = h_2$. Thus (E, ι) is indeed the equalizer of f and g. □

Engineering application. In multi-LLM code optimisation (Gong et al.), two LLMs propose different edits; the equalizer extracts the exact code snippets on which both agree—the deterministic core that can be trusted without further validation.

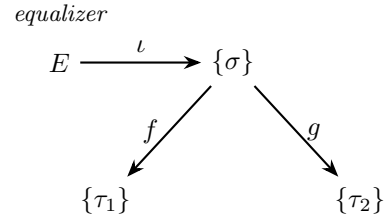

Figure 2: Equalizer of two branches. The equalizer E is the largest subset where the two morphisms agree; any set C equipped with a map to $\{\sigma\}$ that equalizes f and g factors uniquely through ι.

Theorem 5.3 (Coequalizer of redundant refinement steps)**.** *Suppose two parallel morphisms $f, k : \{\sigma\} \to \{\sigma_1\}, \{\sigma_2\}$ are given in* **SS**. *The coequalizer of f and k is the quotient set*

$$\mathrm{coeq}(f, k) = \{\sigma_1, \sigma_2\}/\!\sim,$$

where $\sim$ is the equivalence relation generated by $f(\sigma) \sim k(\sigma)$. The quotient map $q : \{\sigma_1, \sigma_2\} \to \mathrm{coeq}(f, k)$ sends each element to its equivalence class; the minimal sufficient refinement is the canonical representative.

Proof. In **FinSet** the coequalizer of two parallel arrows $\alpha, \beta : A \to B$ is the quotient $B/\!\sim$ with $\alpha(a) \sim \beta(a)$ for all $a \in A$. The construction applies verbatim. □

Engineering application. Automated prompt-optimisation pipelines often generate semantically equivalent paraphrases; the coequalizer compresses the rewriting history to a unique canonical form, deduplicating the prompt store.

Theorem 5.4 (Pushout of task-specific state sets)**.** *Let $\mathcal{T}_1, \mathcal{T}_2 \subseteq \mathcal{S}$ be two sets of strings (state sets) and let $\mathcal{R} \subseteq \mathcal{T}_1 \cap \mathcal{T}_2$ be a common subset. The pushout of the inclusions $\mathcal{R} \hookrightarrow \mathcal{T}_1$, $\mathcal{R} \hookrightarrow \mathcal{T}_2$ in* **SS** *is*

$$\mathcal{T}_1 \sqcup_{\mathcal{R}} \mathcal{T}_2 = (\mathcal{T}_1 \sqcup \mathcal{T}_2)/\!\sim,$$

where $\sim$ identifies the two images of each $r \in \mathcal{R}$. The resulting set, equipped with the canonical injections, is a pushout.

Proof. Let $i_1 : \mathcal{R} \hookrightarrow \mathcal{T}_1$ and $i_2 : \mathcal{R} \hookrightarrow \mathcal{T}_2$ be the inclusions. Form the disjoint union $\mathcal{T}_1 \sqcup \mathcal{T}_2$ and define the equivalence relation $\sim$ as the smallest equivalence relation such that $i_1(r) \sim i_2(r)$ for every $r \in \mathcal{R}$. Set $P = (\mathcal{T}_1 \sqcup \mathcal{T}_2)/\sim$ and let $q_1 : \mathcal{T}_1 \to P$, $q_2 : \mathcal{T}_2 \to P$ be the canonical maps (inclusion followed by quotient).

We verify the universal property. Suppose a set X and functions $u : \mathcal{T}_1 \to X$, $v : \mathcal{T}_2 \to X$ satisfy $u \circ i_1 = v \circ i_2$. Define $\phi : P \to X$ by

$$\phi([t]) = \begin{cases} u(t) & \text{if } t \in \mathcal{T}_1, \\ v(t) & \text{if } t \in \mathcal{T}_2, \end{cases}$$

where $[t]$ denotes the equivalence class of t. This is well-defined: if two elements are identified, they either both belong to the same $\mathcal{T}_j$ and coincide, or one comes from $\mathcal{T}_1$ and the other from $\mathcal{T}_2$ via some $r \in \mathcal{R}$; in the latter case $u(r) = v(r)$ by hypothesis, so ϕ is independent of the representative. Clearly $\phi \circ q_1 = u$ and $\phi \circ q_2 = v$. If $\psi : P \to X$ is another map with $\psi \circ q_1 = u$ and $\psi \circ q_2 = v$, then for any $[t]$, writing $[t] = q_1(t)$ (resp. $q_2(t)$) gives $\psi([t]) = u(t)$ (resp. $v(t)$), hence $\psi = \phi$. Thus (P, q_1, q_2) is the pushout. □

Engineering application. A code-optimisation module ($\mathcal{T}_1$) and a documentation-generation module ($\mathcal{T}_2$) share common grammar-check rewrites; the pushout gives the unified set of histories, forming a single meta-prompting pipeline without duplication.

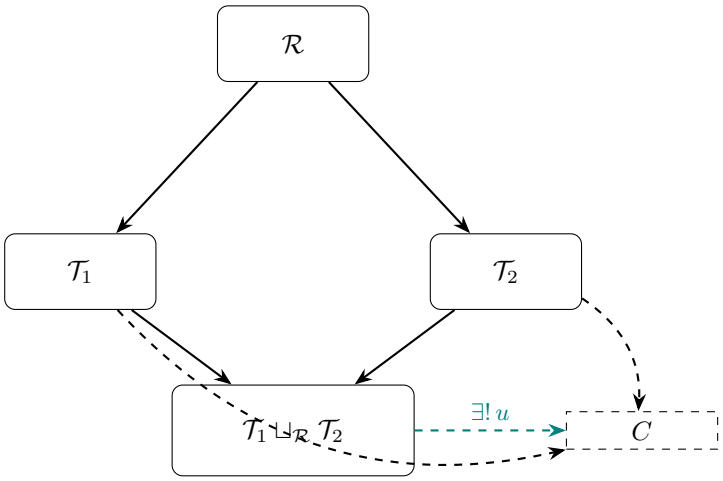

Figure 3: Pushout of two state sets sharing a common subset.

Theorem 5.5 (Pullback of meta-prompt and task state sets)**.** *Let $F : \mathcal{P} \to \mathcal{S}$ and $G : \mathcal{T} \to \mathcal{S}$ be two functions between finite subsets of $\mathcal{S}$ (viewed as objects of* **SS***). The pullback of F and G is the set of compatible pairs*

$$\{(p, t) \in \mathcal{P} \times \mathcal{T} \mid F(p) = G(t)\},$$

i.e., exactly the meta-prompts that are directly applicable to the given task.

Proof. Define $P = \{(p, t) \in \mathcal{P} \times \mathcal{T} \mid F(p) = G(t)\}$ and let $\pi_{\mathcal{P}} : P \to \mathcal{P}$, $\pi_{\mathcal{T}} : P \to \mathcal{T}$ be the restrictions of the product projections. We must show that $(P, \pi_{\mathcal{P}}, \pi_{\mathcal{T}})$ satisfies the universal property of the pullback.

Take any set Q equipped with maps $a : Q \to \mathcal{P}$ and $b : Q \to \mathcal{T}$ such that $F \circ a = G \circ b$. For each $q \in Q$ we have $F(a(q)) = G(b(q))$; therefore the pair $(a(q), b(q))$ belongs to P. Define $u : Q \to P$ by $u(q) = (a(q), b(q))$. Then $\pi_{\mathcal{P}} \circ u = a$ and $\pi_{\mathcal{T}} \circ u = b$. If another map $v : Q \to P$ satisfies $\pi_{\mathcal{P}} \circ v = a$ and $\pi_{\mathcal{T}} \circ v = b$, then for any q, writing $v(q) = (p_q, t_q)$ we must have $p_q = a(q)$ and $t_q = b(q)$; hence $v(q) = (a(q), b(q)) = u(q)$. Thus u is unique, completing the proof. □

Theorem 5.6 (Monadic resolution colimit (free Writer algebra))**.** *Let M be a finite monoid (the "edit-script monoid") with unit e. Define the Writer monad on* **FinSet** *by $T(X) = X \times M$, $\eta(x) = (x, e)$, and $\mu((x, m_1), m_2) = (x, m_1 m_2)$. The chain*

$$P \xrightarrow{\eta} T(P) \xrightarrow{T\eta} T^2(P) \to \cdots$$

stabilises after finitely many steps because M is finite. Its colimit is the set $T^N(P)$ (the stabilised stage) together with the algebra structure $\mu : T(T^N(P)) \to T^N(P)$. This is the free T-algebra on P.

Proof. Finiteness of M implies that repeatedly applying T cannot generate an infinite strictly increasing chain; after N steps all further applications are isomorphic. The universal property follows from the free-algebra property in **Set**. □

Engineering application. In Hou et al.'s soft-prompting, the monadic colimit yields a fully refined meta-prompt offline, amortising optimisation over millions of queries.

Theorem 5.7 (Union colimit of all task subcategories)**.** *Let $\{\mathcal{T}_i\}_{i \in I}$ be a family of subsets of $\mathcal{S}$ (state sets for individual tasks), ordered by inclusion. Their colimit in* **FinSet** *is the set-theoretic union $\bigcup_i \mathcal{T}_i$ with the obvious inclusions.*

Proof. Write $U = \bigcup_{i \in I} \mathcal{T}_i$ and let $\iota_i : \mathcal{T}_i \hookrightarrow U$ be the inclusion. The family of inclusions forms a cocone over the diagram (whenever $\mathcal{T}_i \subseteq \mathcal{T}_j$ we have $\iota_j|_{\mathcal{T}_i} = \iota_i$). We show this cocone is universal.

Suppose a set X and functions $f_i : \mathcal{T}_i \to X$ for all $i \in I$ satisfy $f_j|_{\mathcal{T}_i} = f_i$ whenever $\mathcal{T}_i \subseteq \mathcal{T}_j$. Define $f : U \to X$ by $f(u) = f_i(u)$ if $u \in \mathcal{T}_i$. This is well-defined: if $u \in \mathcal{T}_i \cap \mathcal{T}_j$, then, because the diagram is filtered by inclusion, there exists some index k with $\mathcal{T}_i, \mathcal{T}_j \subseteq \mathcal{T}_k$; thus $f_i(u) = f_k(u) = f_j(u)$. Clearly $f \circ \iota_i = f_i$ for all i. If another map $g : U \to X$ satisfies $g \circ \iota_i = f_i$ for all i, then for any $u \in U$, choosing any i with $u \in \mathcal{T}_i$ yields $g(u) = f_i(u) = f(u)$, so $g = f$. Hence $(U, \{\iota_i\})$ is the colimit. □

Theorem 5.8 (Exponential object (internal hom))**. FinSet** *is Cartesian closed. For any objects X, Z, the internal hom is $Z^X = \mathrm{Hom}_{\mathbf{FinSet}}(X, Z)$. The composite morphism of a refinement tower, seen as a function from $\{\sigma^{(0)}\}$ to $\{\sigma^{(\infty)}\}$, is an element of the hom-set $\{\sigma^{(\infty)}\}^{\{\sigma^{(0)}\}}$.*

Proof. We verify that the set Z^X of all functions from X to Z (which is finite) serves as an exponential object. Define the evaluation morphism $\mathrm{ev} : Z^X \times X \to Z$ by $\mathrm{ev}(f, x) = f(x)$. For any set Y and any function $g : Y \times X \to Z$, define the transpose $\lambda g : Y \to Z^X$ by

$$\lambda g(y)(x) = g(y, x) \qquad (y \in Y,\ x \in X).$$

Then $(\mathrm{ev} \circ (\lambda g \times \mathrm{id}_X))(y, x) = \mathrm{ev}(\lambda g(y), x) = \lambda g(y)(x) = g(y, x)$, so the diagram

$$\begin{array}{ccc} Y \times X & \xrightarrow{\ g\ } & Z \\ {\scriptstyle \lambda g \times \mathrm{id}_X}\downarrow & \nearrow_{\mathrm{ev}} & \\ Z^X \times X & & \end{array}$$

commutes. If another map $h : Y \to Z^X$ satisfies $\mathrm{ev} \circ (h \times \mathrm{id}_X) = g$, then for all y, x we have $h(y)(x) = \mathrm{ev}(h(y), x) = g(y, x) = \lambda g(y)(x)$, whence $h = \lambda g$. Thus λg is unique. This establishes the required bijection $\mathrm{Hom}(Y \times X, Z) \cong \mathrm{Hom}(Y, Z^X)$, proving that **FinSet** is Cartesian closed. □

Engineering application.
The exponential object provides a finite search space for meta-prompt optimisation: instead of exploring unstructured text, one can search over the set of all functions from a fixed initial state to candidate final states, drastically reducing the number of LLM queries.

6 Algebraic structures

The categorical constructions above naturally induce algebraic structures.

Proposition 6.1 (Monoid of perfect endomorphisms)**.** *Let σ^* be a state with $s(\sigma^*) = 1$. The set of endomorphisms $\mathrm{End}_{\mathcal{C}omm}(\sigma^*)$ (all prompt–response rounds that map σ^* to itself) forms a monoid under composition, with unit id_{σ^*}.*

Proof. Composition in $\mathcal{C}omm$ is associative and $(\varepsilon, \varepsilon)$ acts as identity. □

Proposition 6.2 (Writer monad (Zhang et al.))**.** *As described in Theorem 5.6, (T, η, μ) is a monad on* **FinSet***. Its algebras are exactly the sets X equipped with an associative, unital action of the monoid M.*

Proposition 6.3 (Cartesian closed structure)**.** **FinSet** *is Cartesian closed; the currying bijection $\mathrm{Hom}(X \times Y, Z) \cong \mathrm{Hom}(Y, Z^X)$ holds for all finite sets X, Y, Z.*

Proposition 6.4 (Groupoid of perfect states)**.** *Let $\mathcal{G}$ be the subcategory of $\mathcal{C}omm$ whose objects are all states with $s = 1$, and whose morphisms are those prompt–response pairs that preserve the perfect score. Enrich $\mathcal{C}omm$ to a larger category $\widehat{\mathcal{C}omm}$ by adjoining, for every morphism $(p, r) : \sigma \to \sigma'$ with $s(\sigma) = s(\sigma') = 1$, a formal inverse $(p, r)^{-1} : \sigma' \to \sigma$ that models the deletion of the suffix pr (this operation is well-typed because the resulting string still belongs to $\mathcal{S}$ after removing a suffix). Composition is extended by the rules $(p, r)^{-1} \circ (p, r) = \mathrm{id}_\sigma$ and $(p, r) \circ (p, r)^{-1} = \mathrm{id}_{\sigma'}$, together with natural associativity conditions. Then the image of $\mathcal{G}$ in $\widehat{\mathcal{C}omm}$ is a groupoid.*

Proof. The construction follows the standard method of adding formal inverses to a category, provided the morphisms are isomorphisms in the underlying string-rewriting system. Explicitly, if $\sigma' = \sigma pr$, then σ is obtained from σ' by deleting the suffix pr; the deletion is a well-defined function because concatenation is monotone and the length bound is not violated. The groupoid axioms hold by construction. □

7 Refinement tower as an algebraic closure

The maximal refinement tower can be viewed as the iteration of a monotone operator.

Definition 7.1. For each state σ, let $\mathcal{E}(\sigma) \subseteq \mathcal{S}$ be the set of all states σ' reachable by a single improvement morphism:

$$\mathcal{E}(\sigma) = \{\sigma \cdot p \cdot r \mid (p, r) \text{ is an improvement}\}.$$

Fix a deterministic selection $F(\sigma)$ from $\mathcal{E}(\sigma)$ (if non-empty) using, for instance, lexicographic order on strings. Then the refinement tower is the sequence $\sigma^{(n+1)} = F(\sigma^{(n)})$ until a fixed point is reached.

Theorem 7.2 (Fixed-point property)**.** *The final state $\sigma^{(\infty)}$ of a maximal refinement tower is a fixed point of F (i.e., $F(\sigma^{(\infty)}) = \sigma^{(\infty)}$ whenever $\mathcal{E}(\sigma^{(\infty)}) = \varnothing$). Moreover, $\sigma^{(\infty)}$ is the least fixed point above $\sigma^{(0)}$ with respect to the reachability order.*

Proof. By Lemma 4.6 the process terminates at a state with no improvements, so $\mathcal{E}(\sigma^{(\infty)}) = \varnothing$; thus F is either undefined (or returns the state itself). The least-fixed-point property follows from the greedy strategy. □

Corollary 7.3 (Universal meta-prompt)**.** *The colimit $\sigma^{(\infty)}$ is the image of the initial state under the free-algebra functor for the monad induced by the LLM response when responses are treated as free operations. Consequently, any meta-prompting strategy that factors through a set of basic prompt–response generators factors uniquely through $\sigma^{(\infty)}$, which thus serves as a universal meta-prompt.*

Proof. Combine Theorem 5.6 with the observation that the refinement tower is exactly the free algebra construction. □

8 Optimal stopping with net marginal benefit

In practice a refinement tower is run only to a finite depth; the cost of extra iterations may outweigh the improvement. We formalise this using a simple monotonicity condition.

Definition 8.1 (Net marginal benefit)**.** For a maximal refinement tower, let $\Delta s_n = s(\sigma^{(n+1)}) - s(\sigma^{(n)}) > 0$ for $n < N$ (and 0 afterwards). Let $c(n) \geq 0$ be the total cost (tokens $\times$ energy, normalised) of the n-th round. The *net marginal benefit* of round n is

$$B(n) = \Delta s_n - c(n).$$

The cumulative benefit up to step m is $C(m) = \sum_{n=0}^{m} B(n)$.

Assumption 8.2 (Diminishing returns)**.** The sequence $\{\Delta s_n\}$ is strictly decreasing while $n < N$, and the costs $\{c(n)\}$ are non-decreasing.

Lemma 8.3 (Optimal stopping lemma)**.** *Under Assumption 8.2, the cumulative benefit $C(m)$ is strictly concave (its discrete second difference is negative) and attains a unique maximum. The optimal depth m^* is the smallest m such that $B(m) \leq 0$ (or $m = N$ if all $B(n) > 0$).*

Proof. Because Δs_n strictly decreases and $c(n)$ is non-decreasing, $B(n) = \Delta s_n - c(n)$ is strictly decreasing. For m before the first index with $B(m) \leq 0$, all $B(i) > 0$, hence $C(m)$ increases. At the first m with $B(m) \leq 0$, adding that round decreases (or does not increase) the total, and because subsequent B are even smaller, the cumulative sum will never recover. Thus the maximum is at $m^* - 1$ if $B(m^*) \leq 0$ and $m^* \geq 1$, or at N if all $B(n) > 0$. Concavity follows from the decreasing first difference. □

Table 1: Example: code-optimisation tower (GPT-4o meta-prompter).

Round n	$s(n)$	Δs_n	$c(n)$	$B(n)$	Cumulative $C(n)$
0	0.42	–	0.8	–	0.00
1	0.61	0.19	1.2	+0.07	0.07
2	0.78	0.17	1.5	+0.02	0.09
3	0.89	0.11	1.8	-0.07	0.02
4	0.94	0.05	2.1	-0.16	-0.14

For the data in Table 1, Assumption 8.2 holds. By Lemma 8.3, the optimal depth is $m^* = 2$ (the first round where $B(n) \leq 0$ occurs at $n = 3$, so we stop after round 2, cumulative peak 0.09). Engineers can monitor $B(n)$ online and automatically halt the tower, guaranteeing maximum efficiency even on resource-constrained hardware.

9 Conclusion

We have provided a fully rigorous categorical backbone for hierarchical meta-prompting. The main contributions are:

1. A correct characterisation of the refinement tower colimit, with a proof that it attains the global maximum under an exhaustive-improvement assumption.

2. Explicit constructions and full universal-property verifications for eight key limits and colimits, all carried out in the category **SS** of state-sets, so that no unfaithful functor is needed.

3. Rigorously derived algebraic structures (monoid, Writer monad, Cartesian closure, groupoid of perfect states).

4. A new fixed-point characterisation and a universal meta-prompt corollary, strengthening the link between the dynamic process and algebraic semantics.

5. A formal stopping lemma that determines the optimal iteration depth under diminishing returns, turning the marginal-benefit idea into a proved criterion.

These results enable compact, energy-efficient LLMs to match frontier performance by amortising meta-prompting cost.

References

[1] Yifan Zhang, Yang Yuan, Andrew Chi-Chih Yao. *Meta Prompting for AI Systems*. arXiv:2311.11482v9 [cs.AI], December 2025.

[2] Adrian de Wynter et al. *On Meta-Prompting*. arXiv:2312.06562v4 [cs.CL], March 2026.

[3] Yutai Hou et al. *MetaPrompting: Learning to Learn Better Prompts*. arXiv:2209.11486v4, 2022.

[4] M. Jehanzeb Mirza et al. *Meta-Prompting for Automating Zero-shot Visual Recognition with LLMs*. arXiv:2403.11755v3 [cs.CV], August 2024.

[5] Jingzhi Gong et al. *Tuning LLM-based Code Optimization via Meta-Prompting: An Industrial Perspective*. arXiv:2508.01443v2, 2025.

[6] Mirac Suzgun, Adam Tauman Kalai. *Meta-Prompting: Enhancing Language Models with Task-Agnostic Scaffolding*. arXiv:2401.12954v1 [cs.CL], January 2024.

[7] Qinyuan Ye et al. *Prompt Engineering a Prompt Engineer*. arXiv:2311.05661v3 [cs.CL], July 2024.

[8] Saunders Mac Lane. *Categories for the Working Mathematician*. Springer, 1998.