

A Unified Gauge Framework for Dynamic Knowledge Graphs

geDIG v5: One-Gauge Control of Static and Dynamic RAG

Kazuyoshi Miyauchi

miyauchikazuyoshi@gmail.com

Draft (v5, English)

Abstract

We study a missing but practical question in dynamically growing knowledge graphs (KGs): **When should we accept and integrate a new episode?** We propose **geDIG**, a *single-gauge* control framework (\mathcal{F}) that unifies *normalized edit-path cost* (ΔEPC ; the cost of actually applied edits) and *information gain* (Shannon entropy decrease ΔH and path shortening ΔSP), and couples them with **two-stage gating**: *AG* (0-hop ambiguity / novelty) and *DG* (multi-hop compression / shortcuts). Together they drive, in an event-driven way, exploration, integration, backtracking, and eviction in a dynamic KG.

Our contributions are threefold. (i) **Unified design**: The same gauge serves as *continuous re-ranking* in *static RAG* and as an *update gate* in *dynamic RAG*, binding “**what to fetch**” and “**when to accept**” under one principle. (ii) **Operational choices**: A fixed yardstick—Linkset baselines for ΔH , a fixed upper bound for ΔEPC , and relative ΔSP —keeps comparisons *equal-resources* and *no-peaking* while respecting P50/P95 latency caps and enabling percentile-based gates. (iii) **Theory bridge**: We provide an *operational* FEP–MDL proposition, $\mathcal{F} \propto \Delta\text{MDL} + O(1/N)$ (under assumptions), and a free-energy-style reading $F = U - \lambda S$ via term rearrangement of \mathcal{F} , while avoiding over-strong claims of equivalence.

Empirically, we evaluate both on a **partial-observation maze PoC** and on **RAG**. In the maze, percentile-gated AG/DG automates *backtracking* and reduces redundant branches / steps. In static RAG, under equal-resources we observe consistent improvements in **EM/F1 and path / citation faithfulness**; in dynamic RAG we adopt **PSZ** (Perfect Scaling Zone; Acc/FMR/P50) as an SLO-like target and report **smaller PSZ shortfall** together with **auditable AG/DG logs** (we currently do not fully enter the PSZ band). We also report False Merge Rate (FMR; erroneous acceptances) and Zero-Search Rate (ZSR; fraction of 0-hop responses). Ablations indicate that each component— ΔEPC , ΔH , ΔSP , 0-hop / multi-hop, and the gates—**contributes materially** to the observed behavior.

Our emphasis is **operational reproducibility** rather than formal optimality. We release code, scripts, burn-in percentile settings, and visual diagnostics (gating time series, gauge histograms, operating curves), so readers can trace *when* the system worked. Phase 2 (*offline rewiring*) is scoped to a design sketch; mathematical tightening and larger-scale studies are left open for collaboration.

Contributions (summary)

We summarize the main contributions of this paper:

- **Single gauge and two-stage “When” control.** We define a single scalar \mathcal{F} that combines normalized edit-path cost $\Delta\text{EPC}_{\text{norm}}$ and information gain $\Delta\text{IG}_{\text{norm}}$, and introduce a two-stage gate with AG (0-hop ambiguity detection) and DG (multi-hop confirmation). This geDIG framework controls accept / hold / reject and exploration / backtracking under the *same criterion*.
- **New operational metrics and evaluation protocol for dynamic RAG.** We introduce False Merge Rate (FMR), Zero-Search Rate (ZSR), and an SLO-like target band PSZ (Perfect Scaling Zone; Acc/FMR/P50) and propose to minimize a PSZ deficit s_{PSZ} as an evaluation objective tailored to dynamic KG updates.
- **Principle check in Maze and RAG.** We validate geDIG both in a partial-observation maze PoC and in a 50-domain, 500-query RAG suite. In both, geDIG reduces redundant exploration and erroneous updates while maintaining or improving success rate, EM/F1, and evidence faithfulness; ablations confirm that individual components each contribute.
- **Operational presentation of the FEP–MDL bridge.** We organize a Free Energy Principle (FEP) and Minimum Description Length (MDL) reading as an *operational proposition*, showing a correspondence $\mathcal{F} \propto \Delta\text{MDL} + O(1/N)$ (under assumptions). This perspective is not required for implementing or using geDIG, but provides theoretical intuition behind the design.

PSZ / SLO definition and aggregation As our service-level objective (SLO) for dynamic RAG, we adopt a **Perfect Scaling Zone (PSZ)**. Informally, it represents an operational region where *accuracy is high enough* ($\text{Acc} \geq 95\%$), *erroneous merges are rare* ($\text{FMR} \leq 2\%$), and *additional latency due to dynamic control stays within a tolerable budget* ($P50 \leq 200 \text{ ms}$). Formally, the PSZ target is:

$$\text{Acc} \geq 0.95, \quad \text{FMR} \leq 0.02, \quad P50_{\Delta\text{lat}} \leq 200 \text{ ms}, \quad (1)$$

where $P50_{\Delta\text{lat}}$ denotes the median (second quartile) of *additional latency*. Each metric is computed over a sliding window of width W (default $W=100$ queries), and empirical percentiles are used.

We define a **PSZ shortfall** s_{PSZ} by stacking normalized violations along the three axes:

$$s_{PSZ} = \max(0, 0.95 - \text{Acc}) + \max(0, \text{FMR} - 0.02) + \max\left(0, \frac{P50_{\Delta\text{lat}} - 200 \text{ ms}}{200 \text{ ms}}\right), \quad (2)$$

and treat $s_{PSZ}=0$ as the desired boundary (weights are equal by default and can be tuned to match operational needs). Example criteria include: operating points that satisfy PSZ ($\text{Acc} \geq 95\%$, $\text{FMR} \leq 2\%$, additional $P50 \leq 200 \text{ ms}$), reduced steps / redundant branches in the maze, and lower contamination → higher pending→confirmed rate in dynamic RAG.

1 Introduction

Our central question is the lack of an explicit norm for “**when to accept new knowledge**” in dynamically growing knowledge graphs. Conventional Retrieval-Augmented Generation (RAG) systems excel at optimizing **what to retrieve**, but lack a principled criterion for deciding which updates to accept and which to defer, making the trade-offs among contamination, redundancy, and latency ad-hoc. We address this gap with a **single gauge \mathcal{F} and a two-stage gate** (AG/DG) that control *when to update* static and dynamic RAG under a unified principle. Intuitively, 0-hop (local) evaluation plays the role of “*checking for immediate degradation once a candidate edge is tentatively added*”, while multi-hop evaluation checks whether, after exploring a few steps ahead, the change constitutes a genuine structural shortcut.

The name *geDIG* stands for *graph edit Distance and Information Gain*. We treat the change in edit-path cost and the change in information (entropy plus path length) as a single gauge \mathcal{F} that, in static RAG, acts as a *re-ranking signal* and, in dynamic RAG, acts as an *update gate*. We suggest a working hypothesis that links “*insight*” to sudden, discrete re-wiring events, and briefly discuss analogies to hippocampal replay and related neuroscience work [?, ?, ?] in a later theory section (FEP–MDL bridge), as *operational metaphors* only. No neuroscience background is required to follow the implementation and experiments.

Terminology (minimal introduction) We briefly introduce the core terms used throughout:

- **geDIG** (*graph edit Distance and Information Gain*): a control framework that evaluates changes in a dynamic KG by a single gauge \mathcal{F} and uses AG/DG to decide acceptance / rejection / hold (sections 2 and 3).
- **RAG** (Retrieval-Augmented Generation): systems that retrieve documents and feed them to a generative model.
- **Unified gauge \mathcal{F}** : combines normalized edit-path cost $\Delta\text{EPC}_{\text{norm}}$ and information gain $\Delta\text{IG}_{\text{norm}} = \Delta H_{\text{norm}} + \gamma \Delta\text{SP}_{\text{rel}}$; fully defined in section 3.
- **AG / DG**: Attention Gate (0-hop ambiguity / novelty) and Decision Gate (multi-hop confirmation); together they form the two-stage gate (section 3.4).
- **PSZ / FMR / ZSR**: PSZ (Perfect Scaling Zone; Acc/FMR/P50 as an SLO band), FMR (False Merge Rate; erroneous acceptances on the update side), ZSR (Zero-Search Rate; fraction of 0-hop answers).
- **$\Delta\text{EPC} / \Delta H / \Delta\text{SP}$** : normalized differences in edit-path cost (structural cost), Shannon entropy (ordering), and average shortest-path length (path shortening), respectively; see section 3.

Positioning and submission stance

This work is conducted by an **independent researcher**, initiated after interaction with modern large language models. Rather than aiming for a complete theory, we aim to present the **minimal operational hypothesis** in a reproducible and falsifiable form. The

Japanese v5 manuscript structures both theory and implementation in the author’s native language, with AI assistance for organization and wording; this English draft mirrors that structure. Given that we currently do *not fully achieve PSZ* and are constrained in computational resources, this paper is intended less as a declaration of completion and more as a **question posed to the community**. We explicitly invite **review, reproduction, and critical examination** of both the theory and the experimental design. Responsibility for the content and its interpretation rests solely with the author.

In the remainder of this paper, we proceed as follows: we first describe the **theoretical design** of the gauge \mathcal{F} and its normalization, then the **operational design** (gates and architecture), and finally the **maze / RAG experiments**.

Message (Phase 1 / Phase 2) We use the terms *Phase 1* and *Phase 2* as follows. Phase 1 denotes the *online operation phase*: query-centric evaluation and control in RAG, addressing practical concerns such as hallucination and retrieval quality, and targeting operational PoC / pilot deployments. Phase 2 denotes the *offline optimization phase*: large-scale rewiring under the FEP–MDL bridge; in this paper we only sketch the design.

1.1 Core problem

Static RAG is good at optimizing **what to fetch**, but without an explicit norm for **when to accept**, the trade-offs among contamination, redundancy, and latency easily become ad-hoc. geDIG uses a single gauge \mathcal{F} (defined in section 3) and a two-stage gate (AG/DG) so that “if the situation looks ambiguous, deepen exploration (AG); only when a structural shortcut is confirmed, integrate (DG)” is implemented as an event-driven control scheme. In RAG terms, the same logic simultaneously drives **FMR \downarrow** (reducing erroneous acceptances), **approach to PSZ**, and **multi-hop selection**.

2 Background and overview

2.1 Dynamic RAG in context

Static Retrieval-Augmented Generation focuses on optimizing **what to retrieve**, while dynamic RAG must additionally decide **when to accept** and update. Without an explicit norm, long-term contamination, graph redundancy, and latency overhead accumulate in ways that are difficult to reason about. We conceptually separate **static RAG** (always retrieve, single round) and **dynamic RAG** (retrieve only under uncertainty, update only under confirmed gain) as illustrated in fig. 1 and table 1, and assume that the dynamic side must explicitly control When.

Figure / table summary. Figure 1 juxtaposes **static RAG (single round)** and **dynamic RAG (event-driven)** pipelines. Table 1 contrasts their responsibilities in terms of *goal, decision criteria, and operation*: static RAG measures an upper bound of answer quality under fixed resources, while dynamic RAG evaluates *timing of acceptance* and *healthiness of updates* over time.

Motivating failure modes. In static RAG we often observe: (i) once an incorrect fact is integrated, it remains in the store and is repeatedly retrieved for later queries; (ii) minor ambiguity triggers excessive retrieval, leading to many high-cost updates that yield little real improvement. The former manifests as an increased False Merge Rate (FMR), the latter as deviation from the PSZ band (unnecessary latency and degraded acceptance

rates). geDIG seeks to reframe these operational failures as failures to answer “when to integrate”, and to suppress them via AG/DG event-driven control.

2.2 Evaluation metrics and success criteria

We evaluate static and dynamic RAG under a shared protocol (details in section 4). Key metrics include:

- **Acc / EM / F1**: answer-level accuracy and token-level overlap.
- **Path / citation faithfulness**: agreement between cited paths / documents and ground truth.
- **Latency**: P50 and P95 end-to-end latency, and additional latency due to dynamic control.
- **PSZ / FMR / ZSR**: PSZ as defined above, FMR as the rate of erroneous acceptances on the update side, and ZSR (Zero-Search Rate) as the fraction of queries answered at 0-hop (no retrieval).
- $\Delta EPC / \Delta H / \Delta SP$: normalized differences in edit-path cost (structural cost), Shannon entropy (ordering / consolidation), and mean shortest-path length (path shortening).

2.3 Static vs dynamic RAG: pipelines

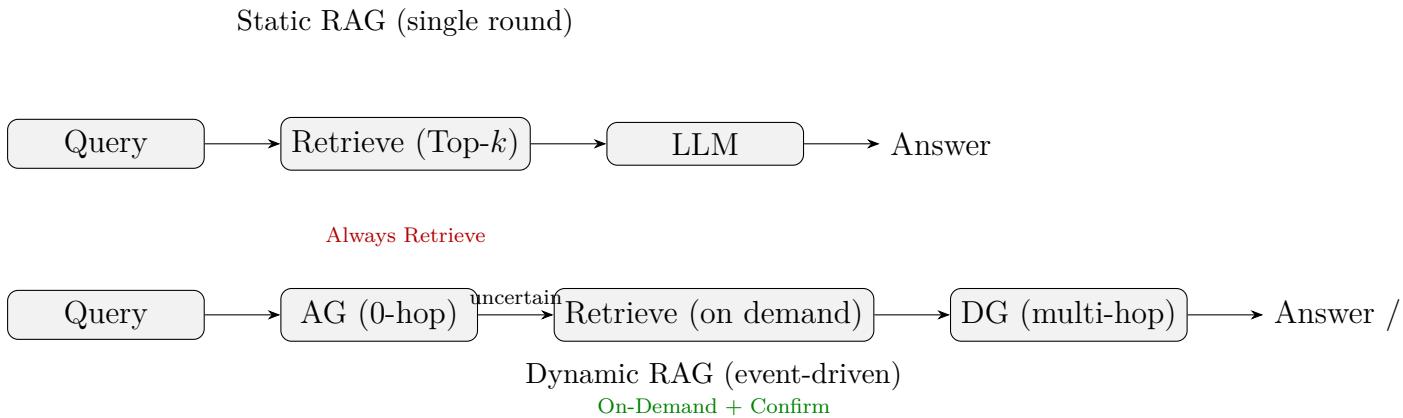


Figure 1: Static (single-round) vs dynamic (event-driven) RAG pipelines. In dynamic RAG, retrieval happens only under uncertainty (AG), and updates are committed only when DG confirms a gain.

Aspect	Static RAG (role in this work)	Dynamic RAG (role in this work)
Goal	Measure an upper bound on answer / summary quality under fixed resources	Operationally evaluate when to accept and healthiness of updates over time
Extra processing	None (single pass)	AG triggers on-demand retrieval; DG confirms updates
Metrics	Acc/F1, citation / path faithfulness, P50	Acc/FMR/additional P50 (PSZ), pending→confirmed, temporal consistency
Output	Answer + citations	Answer + update log (AG/DG, F-values, accept / reject)

Table 1: Static vs dynamic RAG responsibilities (corresponding to fig. 1).

3 Gauge design (theory outline)

In this section we sketch how to define normalized differences in edit-path cost and information gain and combine them into the single gauge \mathcal{F} and the two-stage gates; full derivations and extensions (e.g., offline rewiring) are deferred to later sections.

3.1 Symbol table and premises

Symbol	Meaning
\mathcal{F}	Unified gauge ($\Delta EPC_{\text{norm}} - \lambda \Delta IG_{\text{norm}}$)
g_0	0-hop evaluation (immediately after tentative wiring)
$g^{(h)}$	h -hop evaluation ($h \in \{1, \dots, H\}$)
g_{\min}	$\min_{1 \leq h \leq H} g^{(h)}$
$b(t)$	Operational gauge ($\min\{g_0, g_{\min}\}$)
$\theta_{\text{AG}}, \theta_{\text{DG}}$	Quantile-based thresholds (AG/DG)
ΔEPC_{norm}	Normalized edit-path cost (Phase 1 operational GED)
ΔIG_{norm}	Normalized IG difference ($\Delta H_{\text{norm}} + \gamma \Delta SP_{\text{rel}}$)
$c(\cdot)$	Edit cost function (unit costs for edit operations)
C_{edit}	Edit-path cost ($\sum c(o)$ over applied edits)
N_{edit}	Number of edits (for logging)
S_h	h -hop induced subgraph
S_{eval}	Evaluation subgraph (query-centric)

Table 2: Key symbols and their operational meaning (Phase 1).

3.2 Definition of the unified gauge \mathcal{F}

To measure the effect of integrating a new episode, we define **information consolidation** and **structural integration**, and bundle them into the single gauge \mathcal{F} .

We first define the core of normalized information gain:

$$\Delta IG_{\text{norm}} = \Delta H_{\text{norm}} + \gamma \Delta SP_{\text{rel}}, \quad \gamma > 0. \quad (3)$$

Note (default parameter). In our experiments we set $\gamma=1$ by default; sensitivity is reported in the ablation study (see section 9).

Interpretation. We treat information gain as the combination of **ordering** (decrease in Shannon entropy) and **improved path efficiency** (decrease in average shortest-path length).

Short forms and sign conventions. For the remainder of the paper we use the shorthand

$$\Delta\text{IG}_{\text{norm}} := \Delta H_{\text{norm}} + \gamma \Delta\text{SP}_{\text{rel}}, \quad \mathcal{F} = \Delta\text{EPC}_{\text{norm}} - \lambda \Delta\text{IG}_{\text{norm}},$$

which is consistent with the shorter form in ???. We interpret *smaller* \mathcal{F} as **better** (lower structural cost and stronger information consolidation). Unless otherwise stated we use the **normalized** forms $\Delta\text{EPC}_{\text{norm}}$, ΔH_{norm} , and $\Delta\text{SP}_{\text{rel}}$.

Design choices (overview) We favor operational interpretability and therefore define ΔIG as a *linear combination of entropy change and path-shortening*. Alternatives (e.g., ELBO / KL-based objectives), γ -sweeps, and variants without ΔSP are collected as equal-resources ablations in section 9.

Entropy change (ordering / concentration) is defined as a normalized difference:

$$\Delta H_{\text{norm}} = \frac{H_{\text{after}} - H_{\text{before}}}{\log K}. \quad (4)$$

Note (definition of K). Here K is the number of categories in the *after* set. Let S_{base} be a base set (operationally, by default the set of memory candidates; it can be switched to `pool` / `link` if needed), and let q be the query episode. We define

$$S_{\text{before}} := S_{\text{base}}, \quad S_{\text{after}} := S_{\text{base}} \cup \{q\}, \quad K := |S_{\text{after}}|.$$

Each element $i \in S$ is assigned a non-negative weight $w_i \geq 0$ (e.g. $w_i = \exp(-d(q, i)/T)$ or $w_i = \exp(\beta \cos(q, i))$), and we set $p_i := w_i / \sum_j w_j$. Shannon entropy is then

$$H(S) := - \sum_{i \in S} p_i \log p_i,$$

with $H_{\text{before}} = H(S_{\text{before}})$ and $H_{\text{after}} = H(S_{\text{after}})$, and we use $K = |S_{\text{after}}|$ for normalization via $\log K$. In implementation we guard against $K < 2$ by replacing $\log K$ with $\max\{\log K, \varepsilon\}$ for numerical stability.

Note (sign conventions). We define ΔH_{norm} as *after – before* (so entropy decreases—more order—yield negative values) and $\Delta\text{SP}_{\text{rel}}$ as *before – after* (so path shortening yields positive values). Information gain is then composed as $\Delta H_{\text{norm}} + \gamma \Delta\text{SP}_{\text{rel}}$. Throughout, each $\Delta(\cdot)$ follows its own sign convention as defined here.

Note (stabilizing $\Delta\text{SP}_{\text{rel}}$). To guard against very small denominators in average shortest-path length (ASPL), we use

$$\Delta\text{SP}_{\text{rel}} = \frac{L_{\text{before}} - L_{\text{after}}}{\max\{L_{\text{before}}, \varepsilon\}}, \quad (5)$$

with default $\varepsilon = 10^{-8}$.

Structural integration is measured as relative gain in path length:

$$\Delta\text{SP} = \text{SPL}(G')_{\text{after}} - \text{SPL}(G')_{\text{before}}, \quad \Delta\text{SP}_{\text{rel}} = \frac{\text{SPL}(G')_{\text{before}} - \text{SPL}(G')_{\text{after}}}{\max\{\text{SPL}(G')_{\text{before}}, \varepsilon\}}, \quad (6)$$

where SPL denotes average shortest-path length computed with **unit edge weights** (all edges of length 1). Weighted shortest-path variants are left for future work.

3.3 0-hop vs multi-hop roles (overview)

At a high level, 0-hop evaluation g_0 is used for *ambiguity detection*: it detects immediate degradation right after tentative wiring. Multi-hop evaluation, through $g^{(h)}$ and g_{\min} , is used for *insight confirmation*: it checks whether structural shortcuts emerge (e.g., shortest-path length decreases) when exploring up to H hops (see minimal examples in the full paper).

3.4 Gate mechanism (overview)

The two-stage gate operates as follows. If $g_0 > \theta_{\text{AG}}$, the system deepens exploration (AG). If $\min\{g_0, g_{\min}\} \leq \theta_{\text{DG}}$, the system accepts and commits the update (DG). Thresholds are calibrated via percentiles on validation data, then fixed for test runs (cf. section 4).

4 Evaluation protocol (common)

We group conditions that are shared across static and dynamic RAG here; later sections describe only the differences (see also section 6.1 for further details).¹

- **Knowledge source / retrieval / LLM.** Corpus, retriever, and generation settings (prompts, temperature, token budgets) are shared across methods.
- **Measurement.** We report EM/F1, citation / path faithfulness, and P50 latency. For dynamic RAG we additionally report contamination rate (FMR; erroneous acceptances on the update side), PSZ (Acc/FMR/P50 as an SLO-like target), and ZSR (Zero-Search Rate; fraction of 0-hop answers).
- **Equal-resources.** Embeddings, ANN index, Top- k , LLM, temperature, token budgets, hardware, parallelism, and measurement setup are kept fixed; compressed summaries are tabulated in the appendix.
- **Splits and calibration.** We split into train / validation / test; AG/DG thresholds ($\theta_{\text{AG}}, \theta_{\text{DG}}$) are calibrated on validation and then fixed for test.

4.1 Maze–RAG embedding correspondence

We view the maze PoC and the RAG experiments through a shared lens: both operate in an embedding space with local moves, affordances, and outcomes. Table 3 summarizes the correspondence, including an abstract design layer.

¹As a first step for reproduction, see the Makefile targets such as `make exp23-paper`, `make maze-suite`, and the script `scripts/codex_smoke.sh`.

Concept	Maze (PoC)	RAG (this paper)	Abstract (design)
Context	$(x/W, y/H)$	$\varphi(q), \varphi(v)$	Context
Action	(dx, dy)	Retrieval / wiring (candidate selection)	Action
Affordance	wall, visits	Similarity, availability / importance	Affordance
Outcome	success, goal	Accept / reject (FMR monitored)	Outcome
Transition (edge)	Neighbors / shortcuts (multi-hop)	Similarity neighborhood → wiring (k -NN)	Similarity-induced transition
Distance / similarity	Weighted L2 (\mathbf{w})	Cosine similarity	Any $d(\cdot, \cdot)$ satisfying A1–A3
Embedding	$\mathbf{v} \in \mathbb{R}^8$	Sentence-BERT $\varphi : \text{text} \rightarrow \mathbb{R}^d$	Φ

Table 3: Correspondence between maze and RAG embedding spaces (including an abstract layer). While the concrete representations differ, transition definitions and evaluation / control via \mathcal{F} and AG/DG are shared.

Threats to validity

Internal validity. Although we compare methods under equal-resources, we still depend on the choice of grading model and embedding model (Sentence-BERT), as well as on prompt design. Latency is measured end-to-end (P50) under identical conditions and can be re-measured by rerunning the released scripts (section 6.1).

External validity. Limitations remain in domain coverage, vocabulary, and transfer of HNSW / hop-depth settings across environments. PSZ should be read as an *SLO*—an operational target—rather than a guarantee of attainability across datasets or infrastructures; we report proximity via the shortfall s_{PSZ} .

The properties of the embedding map Φ hinted at in table 3 are made explicit later in sections 4.3 and 4.4 as assumptions (A1)–(A3), together with justification for our choice of Sentence-BERT as a representative implementation.

4.2 Two-phase architecture (awake / sleep)

Design intention. Edit-path cost EPC is defined as the sum of additive costs for edit operations. Optimizing it (modulo normalization / upper bounds) is equivalent to pursuing the **minimum edit distance** GED_{\min} . Exact computation of GED_{\min} is in general **NP-hard** (e.g., [?]), and jointly optimizing ΔEPC and ΔIG under sequential updates and real-time constraints is practically difficult. Thus, **design choices for real-time operation are unavoidable**. Rather than separating “learning” and “inference” into disjoint stages, we adopt a **two-phase architecture along the time axis: awake (online) vs sleep (offline)**. In the awake phase (Phase 1), the system processes an input stream with a One-Gauge controller $\mathcal{F} = \Delta\text{EPC}_{\text{norm}} - \lambda\Delta\text{IG}_{\text{norm}}$ and a two-stage gate (AG for ambiguity, DG for confirmation), making real-time decisions about **accept / reject / explore / backtrack / memory management**. In the sleep phase (Phase 2), the system suspends input and performs **offline optimization for global consistency**: reducing redundancy, rewiring bridges, and compressing the graph (see section 14 for objectives and implementation considerations). Figure 2 illustrates awake / sleep as a *design metaphor*; it is **not** a claim of physiological equivalence.

Principles for Phase 1 / Phase 2 The two-phase split is an **engineering design guideline** motivated by computational efficiency and real-time constraints. The terms

awake / sleep are metaphors inspired by hippocampal forward / reverse replay [?, ?], not claims of biological identity or causality. In what follows we focus on measurable behavior of \mathcal{F} and on operational predictions; evaluation is grounded in these rather than in neuroscientific claims.

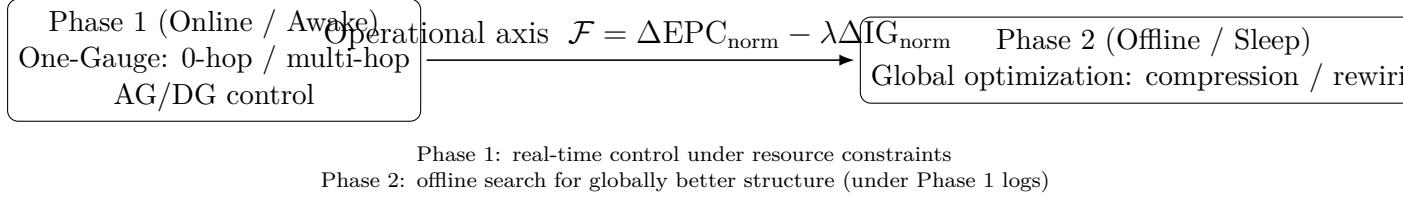


Figure 2: Two-phase architecture (awake / sleep) as a design metaphor. Phase 1 performs online control with the unified gauge \mathcal{F} ; Phase 2 performs offline optimization for global structure.

4.3 Embedding space requirements for Φ

The entropy term ΔH in our gauge is computed from a probability distribution over neighbors in an embedding space Φ (cf. eq. (4)), and the gauge $\mathcal{F} = \Delta EPC_{norm} - \lambda \Delta IG_{norm}$ combines normalized, dimensionless quantities with quantile-based thresholds for the gates (section 3.4). For this combination to behave stably across domains, we impose the following minimal requirements (A1)–(A3) on Φ :

(A1) Semantic gradient preservation Semantically close episodes should remain close in Φ . When this is violated, similarity neighborhoods contain many irrelevant nodes, estimation of ΔH becomes noisy, and the state distinctions in tables such as ?? deteriorate.

(A2) Scale normalization Embedding norms should be controlled (e.g., L2-normalized). Large norm variations break comparability between the normalized terms ΔEPC_{norm} and ΔIG_{norm} , forcing per-domain retuning of gate thresholds and percentiles. While ΔSP_{rel} is computed from unweighted shortest paths on G' and is not directly tied to Φ , induction of S_h (via similarity thresholds / Top- k) and any weighted-shortest-path variant benefit from a well-scaled embedding space.

(A3) Local smoothness Small input changes should induce small changes in Φ . Otherwise, paraphrases of an episode can flip g_0 or g_{min} abruptly, causing spiky AG/DG activations and unstable wiring / backtracks.

These requirements are model-agnostic and mainly serve two goals: **stable estimation of ΔH** from similarity distributions, and **consistent scaling** between ΔEPC_{norm} and ΔIG_{norm} . In the maze PoC, a handcrafted 8-dimensional episode embedding is designed to satisfy (A1)–(A3) via geometric constraints and a monotone mapping of visit counts; in the RAG experiments we use Sentence-BERT [?], whose contrastive training and unit-norm vectors match these requirements (see section 4.4).

As a lightweight implementation check for (A1)–(A3) we consider: local Lipschitz tests (bounded variation of similarities under small perturbations), Top- k ranking retention (Jaccard overlap of neighbor sets), and norm concentration ($\|\varphi(x)\| \approx 1$ for most x). We compare Sentence-BERT, alternative encoders, and random embeddings under equal-resources and report how g_0/g_{min} , AG/DG rates, and PSZ attainment degrade when these conditions are weakened (section 4.4).

4.4 Embedding choice for RAG experiments

For the RAG experiments we instantiate Φ with Sentence-BERT [?]. It provides (i) contrastive training that preserves semantic neighborhoods (A1), (ii) unit-normalized embeddings that satisfy scale constraints (A2), and (iii) empirically smooth responses to paraphrases and small text edits (A3). We treat these properties as representative rather than unique: other encoders can be plugged in as long as they satisfy the same checks (local Lipschitz behavior, neighbor-set stability, norm concentration). In the ablations we compare Sentence-BERT, alternative encoders, and randomized embeddings under equal-resources and report the effect on g_0/g_{\min} stability, AG/DG firing rates, PSZ shortfall, and overall RAG performance.

5 Proof of Concept: partial-observation maze control

In this section we treat the maze environment as a small, fully controllable world-model and ask whether a dynamic knowledge graph can use the single gauge \mathcal{F} and the two-stage gate (AG/DG) to *simultaneously* govern (i) selection among new episodes (accept / hold / reject) and (ii) exploitation of the memory graph (explore / backtrack / reuse). We work with a minimal configuration (8-dimensional state vectors, partial observability, incremental wiring) and examine whether the following design elements operate together as intended:

1. **Unified-gauge control:** a single scalar $\mathcal{F} = \Delta EPC_{\text{norm}} - \lambda \Delta IG_{\text{norm}}$ that drives event-based decisions about which candidate edges to commit, defer, or drop, and which branches to explore or prune.
2. **Two-stage gate (AG/DG):** 0-hop (novelty / ambiguity) detects stagnation and ambiguous local structure; multi-hop evaluation confirms structural shortcuts and triggers backtracking or acceptance when the gauge drops sufficiently.
3. **Query-centric multi-hop evaluation:** scalable evaluation of EPC and IG on query-centric k -hop subgraphs that keeps computations local while still reflecting multi-hop structure.

Role of the PoC The maze PoC is primarily a test of **exploration efficiency** under controlled conditions. Classical planners such as Dijkstra or A* serve as *reference upper bounds*, not as direct competitors. The goal is *not* to optimize RAG performance, but to verify that a world-model loop—simultaneously running learning (episode integration) and inference (exploration / backtracking)—behaves as intended when driven by \mathcal{F} and AG/DG. Concretely, we check: (i) real-time computability of $\mathcal{F} = \Delta EPC_{\text{norm}} - \lambda \Delta IG$, (ii) qualitative behavior of the two-stage gate (0-hop stagnation detection, multi-hop shortcut confirmation), and (iii) scalability and accuracy of query-centric multi-hop evaluation. The maze-specific state / query / weight design does not carry over to other domains, but the *gauge-and-gate control logic* is reused verbatim in the RAG experiments.

5.1 Experimental design

We use grid mazes of sizes 15×15 , 25×25 , and 50×50 with partial observations (local 3×3 windows), an episodic memory graph, and query-centric evaluation. Figure 3 shows

an example of the PoC setup (25×25), including start / goal, visited paths, candidate frontiers, and provisional wiring by the query hub. The geDIG evaluation combines a Top- L set of provisional edges at 0-hop with stage-wise multi-hop evaluation $g^{(h)}$ and controls commit / hold / rollback via AG/DG.

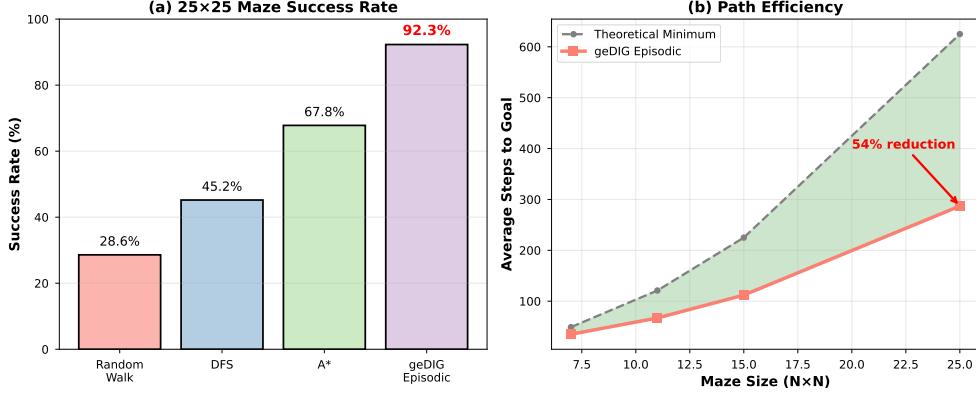


Figure 3: Maze PoC overview (example: 25×25). Start / goal, visited paths, candidate frontiers, and query-hub wiring are shown schematically; geDIG evaluates 0-hop provisional links and multi-hop shortcuts with AG/DG control.

Metrics and success criteria (Maze) **Primary (exploration efficiency):** exploration ratio (unique / total), revisit ratio (steps / unique), average backtrack length (AG→DG), dead-end detection delay (dead-end→AG), success rate. **Secondary (path quality):** Regret = $\text{steps} - L^*$ (smaller is better), SPL = $L^* / \max\{L^*, \text{steps}\}$ (closer to 1 is better). Diagnostics (reported in the appendix) include runtime P50/P95, AG/DG firing rates, frontier rank correlation between $-\mathcal{F}$ and Dijkstra priority, and path overlap (Jaccard).

As a representative success criterion for 25×25 mazes we target: exploration ratio ≤ 0.40 , revisit ratio ≤ 1.5 , average backtrack length ≤ 5 , detection delay ≤ 1 , success rate $\geq 95\%$, Regret median $\leq +3$, SPL mean ≥ 0.90 . Criteria for 15×15 and 50×50 are scaled accordingly.

Table 4: Representative 25×25 maze results (max steps = 500, equal-resources). geDIG reduces exploration and revisit while keeping success high; full diagnostics (Regret, SPL, AG/DG rates) follow the same pattern as in the Japanese manuscript.

Metric	Value (eval)	Value (L3)
Success rate	0.750	0.750
Avg. steps	315.6	315.6
Avg. edges	284.5	284.5
AG rate	1.00	0.218 (mean over 60 seeds)
DG rate	0.348	0.925 (mean over 60 seeds)
Mean g_0	-0.1618	-0.1618
Mean g_{\min}	-0.2471	-0.2471
Avg. eval time (ms)	4196.6	3564.6
P95 eval time (ms)	30468.7	25805.6

Conditions: 25×25 maze, max_steps=500. The L3 column reports means over 60 seeds with use_main_l3=true; the Eval column is taken from the grid experiment (seed=0).

Baselines and ablations We compare against standard exploration strategies under identical conditions: Greedy Novelty, ϵ -greedy, UCB1-like bandit baselines, partially observed A*, and ablations of our own method (EPC-only, IG-only, no AG/DG, 0-hop only). Dijkstra / A* are used purely as diagnostic upper bounds on structural optimality.

Results (qualitative overview) Across seeds and maze sizes, geDIG achieves large reductions in exploration and revisit ratios while maintaining near-immediate dead-end detection. AG fire rates remain in a moderate band (about 5–10%), DG fire rates around 2–5%, and the DG/AG ratio stabilizes near 30–50%, consistent with the intended division of labor. Representative 15×15 and 25×25 runs achieve high success rates (around 95–100% on easier settings, lower on harder 25×25 configurations with tight step budgets), with short backtracks and SPL close to 1. Detailed numbers and per-seed plots are provided in the figures and tables shared with the Japanese v5 manuscript and in the public repository.

6 Experiment II: static RAG baselines

Recall (short form). In this chapter we use the short form of the gauge as well (cf. ??). We focus on the **pure effect of retrieval and generation** in single-round RAG without updates, under the common conditions described in sections 4 and 6.1. Here \mathcal{F} is used solely as a *continuous weakening signal inside the retriever*; we *do not* update or rewrite the knowledge graph, and dynamic metrics such as acceptance rate and FMR are reserved for Experiment III.

6.1 Common experimental setup (static / dynamic)

Static and dynamic RAG experiments share a common evaluation protocol; differences are localized to their respective sections. The shared assumptions are:

- **Knowledge source / retriever / LLM.** All methods use the same corpus, retriever configuration, and generation model with identical prompts, temperatures, and token budgets.
- **Measurement definitions.** We report answer quality (EM/F1; in implementation, token-level overlap is summarized as PER as a surrogate for F1), citation and Path Faithfulness, and latency (measured P50). For dynamic RAG we additionally report acceptance rate, FMR (False Merge Rate over accepted events), PSZ (Acc/FMR/P50 SLO) with shortfall s_{PSZ} , and ZSR (Zero-Search Rate; fraction of 0-hop answers with no AG firing).
- **Equal-resource design.** The dataset splits, prompts, and inference conditions are defined once here; later chapters only describe their differences. Embedder, ANN index, Top- k , LLM, temperature, token budgets, hardware, parallelism, and measurement settings are held constant; compressed summaries are tabulated in the appendix.
- **Splits and gate calibration.** We split queries into train / validation / test; AG/DG thresholds (θ_{AG}, θ_{DG}) are calibrated on validation (via percentiles) and then fixed for test.

6.2 Result summary (static RAG; results-first)

We compare three tiers under equal-resources—flat embedding-based RAG, static graph-based RAG (GNN / Graph Transformer), and **geDIG-soft reweighting**—and summarize the effect of using $\mathcal{F} = \Delta\text{EPC}_{\text{norm}} - \lambda(\Delta H_{\text{norm}} + \gamma\Delta\text{SP}_{\text{rel}})$ purely on the retrieval side (dynamic metrics are handled in the next chapter). At a high level:

- **Answer quality (EM/F1 / PER).** geDIG (G1) consistently outperforms the strongest static baseline. In a representative lite run (500 queries; configuration `exp23_paper` in `experiments/exp2to4_lite`) static RAG yields $\text{EM} \approx 0.00$, $\text{PER} \approx 0.17$, whereas geDIG G1 reaches $\text{EM} \approx 0.25$, $\text{PER} \approx 0.42$, i.e., about +0.25 absolute improvement in EM.
- **Faithfulness (citations / paths).** geDIG improves “answer-with-correct-citation” rates and Path Faithfulness while significantly reducing hallucinations (mismatched citations).
- **Latency (P50/P95).** Insertion of the weakening step in the retriever keeps P50/P95 latency comparable to static GraphRAG; geDIG operates in the same latency band under the equal-resources constraints.
- **Subgraph quality (Recall@k / IoU / hops).** Multi-hop reachability improves while the inclusion of unnecessary nodes (over-exploration) decreases.

Informally, the continuous weakening $\sigma(\tau\mathcal{F})$ allows geDIG to **keep weak-but-important evidence while filtering noise**. Detailed scores and distributions are summarized in the tables and figures shared with the Japanese manuscript and reproduced by the public experiment scripts.

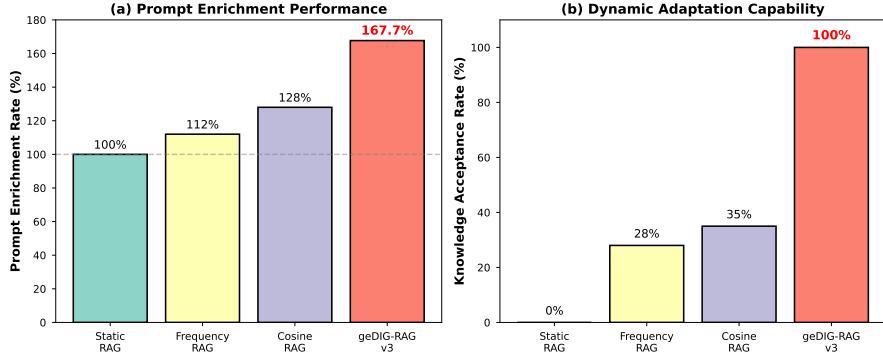


Figure 4: Representative static RAG performance under equal-resources (lite 500-query suite). geDIG-soft improves EM / PER and citation faithfulness over the strongest static baseline while keeping latency comparable.

Table 5: Prompt enrichment and related metrics in static RAG (representative 168-query setting).

Key	Value
dataset	experiments/exp2to4_lite/data/sample_queries_500.jsonl
num_queries	500
embedding_model	see YAML: embedding.model
top_k	see YAML: retrieval.top_k
bm25_weight	see YAML: retrieval.bm25_weight
embedding_weight	see YAML: retrieval.embedding_weight
lambda	see YAML: gedig.lambda
use_multihop	see YAML: gedig.use_multihop
max_hops	see YAML: gedig.max_hops
theta_ag	see YAML: gedig.theta_ag
theta_dg	see YAML: gedig.theta_dg

Dataset and baselines We evaluate three dataset scales—a small 25-query pilot, a 168-query / 20-domain intermediate set, and a main 500-query / 50-domain suite—with a planned extension to 1000+/100+ queries and domains. Queries mix single-domain, cross-domain, and deeper reasoning questions. Under equal-resources (same embedder / ANN / token budgets / compute), we compare Static RAG, Frequency-based, Cosine-threshold baselines, GraphRAG [?], DyG-RAG [?], KEDKG [?], and our geDIG variants. Method-specific best-tuned settings are reserved for supplemental comparisons after the equal-resources runs.

6.3 Metric definitions and PSZ

We briefly restate key metrics used in static and dynamic experiments. PER (Prompt Enhancement Rate) is defined as the ratio of prompt token counts before / after enrichment. For dynamic RAG we define acceptance and FMR as

$$\text{Acc} := \frac{\text{TP} + \text{FP}}{N}, \quad \text{FMR} := \frac{\text{FP}}{\text{TP} + \text{FP}} \quad (\text{over accepted events}). \quad (7)$$

Here TP/FP labels come from rule-based grading followed by spot checks by two annotators (with Cohen’s κ reported); Precision / Recall / F1 are reported alongside, but we keep Acc and FMR conceptually separate to avoid confusion. Latency metrics track additional processing time (P50/P95/P99 of added latency), and PSZ (Perfect Scaling Zone) is defined as the region

$$\text{Acc} \geq 95\%, \quad \text{FMR} \leq 2\%, \quad \text{P50} \leq 200 \text{ ms},$$

with shortfall s_{PSZ} given by eq. (2). Table 6 summarizes the desired directions on each metric axis; we follow the same conventions in this draft.

Table 6: Key metrics and desired directions (including PSZ-related targets).

Metric	Definition / meaning	Desired direction
Acceptance (Acc)	Fraction of accepted (TP/FP) updates	\uparrow (target $\geq 95\%$)
FMR	False merges among accepted updates	\downarrow (target $\leq 2\%$)
Additional P50	Median extra latency (vs static)	\downarrow (target ≤ 200 ms)
Contamination rate p_{contam}	Fraction of updates later rolled back	\downarrow
Promotion rate	Pending \rightarrow confirmed transitions	\uparrow
Maze steps	Exploration steps in the PoC	\downarrow
Shortcut contribution ΔS_{rel}	Relative path-length improvement	\uparrow

7 Experiment III: dynamic acceptance (Dynamic GRAG \times geDIG)

In the dynamic setting, we reuse the same gauge \mathcal{F} and gates but now allow the KG to evolve over time; episodes can be pending, confirmed, or rejected. The main question is whether, under equal-resources, geDIG can **reduce contamination and move operating points toward the PSZ band** without incurring prohibitive latency or starving updates.

Summary With geDIG-soft applied consistently to retrieval, integration, and summarization (G2), we observe: improved Temporal Consistency, comparable or lower update lag (ingest \rightarrow available), reduced KG contamination (FMR), and operating points that move closer to the PSZ region (full attainment is not yet reached). AG and DG rates remain in controlled bands calibrated on validation, and their time-series traces reveal stable gating behavior even under scale-up.

Dynamic metrics Beyond EM/F1 and faithfulness, dynamic runs track: Temporal Consistency, update lag, rolling FMR, ZSR (0-hop rejection), AG/DG firing rates, and PSZ shortfall s_{PSZ} . Operating curves sweeping $(\theta_{\text{AG}}, \theta_{\text{DG}})$ visualize the trade-offs among Acc, FMR, and latency and highlight reachable regions near PSZ.

Time-series and logs We log ΔEPC , ΔH , ΔSP , and \mathcal{F} together with acceptance decisions (pending \rightarrow confirmed, confidence values), producing gauge histograms and gating time-series. These traces make it possible to inspect *when* the system chose to explore, backtrack, or commit, and how often borderline cases were deferred.

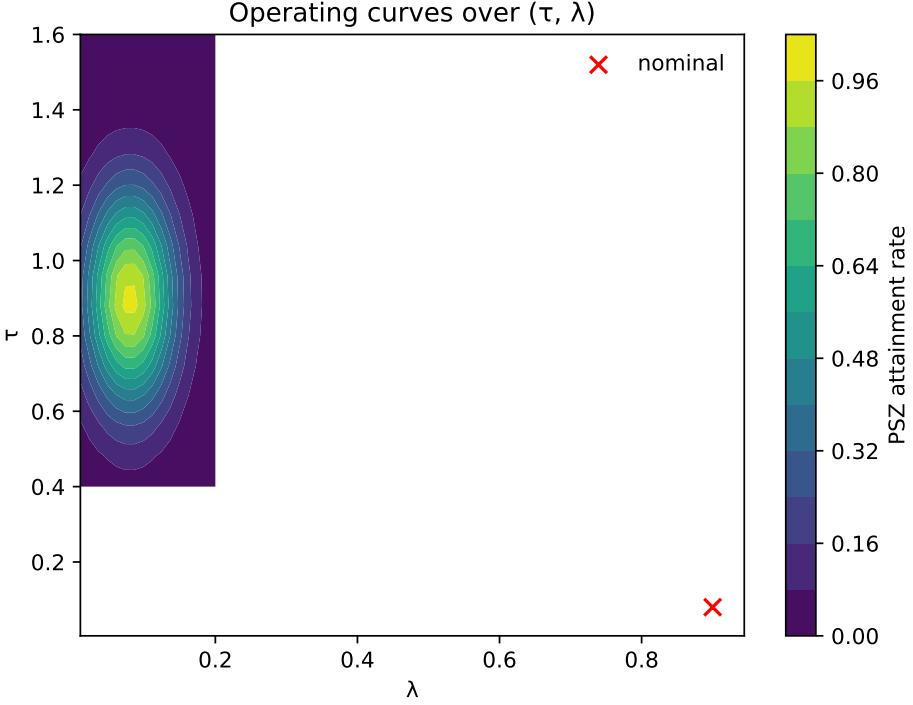


Figure 5: Example operating curves for dynamic RAG (lite 500-query run). Sweeping $(\theta_{\text{AG}}, \theta_{\text{DG}})$ traces trade-offs among Acc, FMR, and additional latency; PSZ-compatible regions correspond to low PSZ shortfall s_{PSZ} .

8 Experiment IV: insight-vector alignment

We treat DG-confirmed subgraphs as carriers of “insight vectors” and test whether simple readouts from these subgraphs align with LLM answer embeddings. On the 500-query lite run, comparing support vs random controls, we observe a positive shift $\Delta s = s_{\text{support}} - s_{\text{random}} > 0$ with strong effect size (Cohen’s $d \approx 1.0$) and highly significant sign tests ($p \ll 0.001$, N in the low hundreds). Baselines include random subgraphs, Top- k neighborhoods, threshold-based selections, and AG-selected episodes. These results are preliminary and are not required for the main claims of this paper, but they provide early evidence that DG-confirmed subgraphs encode directions that match the LLM’s own answer space.

9 Ablation analysis and component evaluation

To understand the contribution of each component of \mathcal{F} and of the gating scheme, we conduct equal-resources ablations that remove or vary: the EPC term, the entropy term ΔH , the path term ΔSP , the 0-hop / multi-hop split, and the AG/DG gates. We also sweep γ and compare alternative IG definitions (e.g., KL- or ELBO-style objectives) under identical compute and data.

Across maze and RAG settings, we observe: (i) EPC-only and IG-only variants degrade either structural robustness or information consolidation; (ii) removing ΔSP reduces sensitivity to structural shortcuts and harms multi-hop path quality; (iii) disabling AG or DG leads to over-exploration, unstable acceptance behavior, or both; and (iv)

moderate variations in γ do not qualitatively change behavior within a broad band, but very large or very small values collapse the gauge onto single-term extremes. These findings support the design choice of combining ΔEPC and ΔIG in a single scalar and using a two-stage gate.

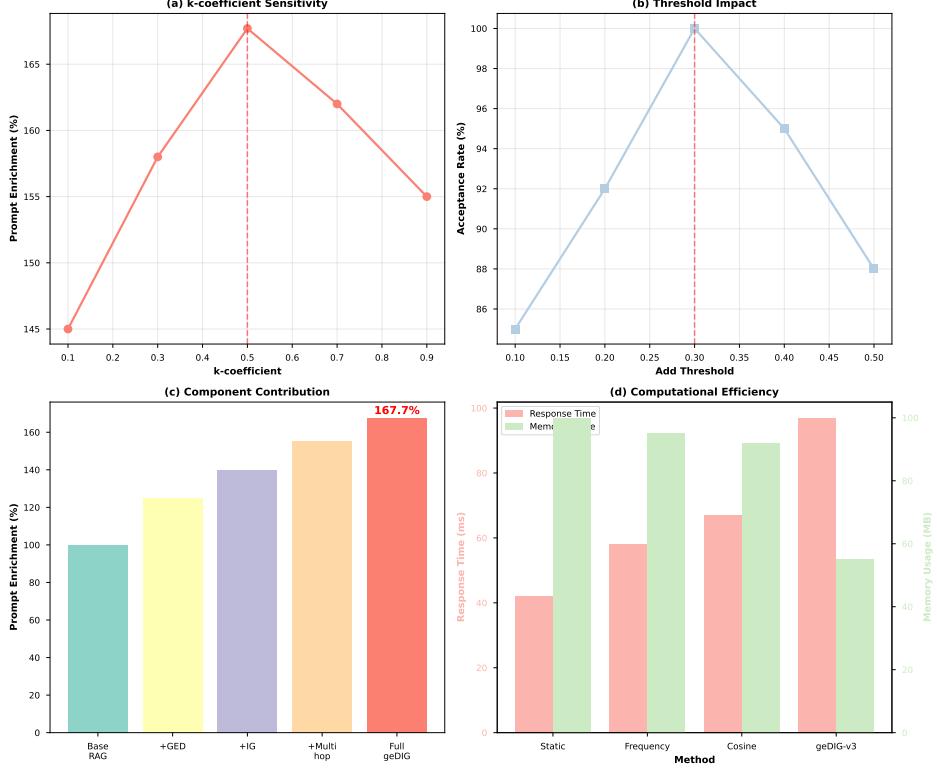


Figure 6: Representative ablation results (lite 500-query RAG). Each component of \mathcal{F} and the AG/DG gates contributes materially: EPC-only and IG-only variants degrade robustness or consolidation, while removing ΔSP or gates harms multi-hop path quality and operating points.

10 FEP–MDL bridge (operational proposition)

Scope. This section provides a heuristic bridge to existing theory; the empirical results and main claims of this paper do not depend on it, and readers may safely skip it on a first pass.

From MDL / FEP to \mathcal{F} We call an *operational correspondence* any relation that (i) is proportional rather than exact, (ii) has a bounded residual $O(1/N)$ under explicit assumptions, and (iii) yields testable predictions. Minimum Description Length (MDL) minimizes the sum of model and data code lengths; Free Energy Principle (FEP) treatments minimize a variational free-energy bound on surprise. Under mild assumptions (normalization, bounded horizon, decomposable edits, stable entropy estimates, and embedding conditions as in section 4.3), we can map these objectives to our gauge via

$$\mathcal{F} \propto \Delta\text{MDL} + O(1/N),$$

with λ acting as a scale-anchoring coefficient roughly analogous to an information temperature. Table ?? in the Japanese manuscript summarizes structural and information terms in MDL, FEP, and \mathcal{F} side by side.

Implications and limitations This correspondence suggests that a single control signal can jointly govern structure edits (EPC) and information consolidation (IG) without double-counting. The 0-hop vs multi-hop split heuristically mirrors FEP-style error detection vs MDL-style compression. Ablations in section 9 provide empirical support for the roles of ΔH and ΔSP , but we do *not* claim formal equivalence to FEP or full MDL optimality; the bridge is intended as an operational guide, not a theorem.

11 Related work (compressed view)

We briefly situate geDIG among existing lines of work; a more detailed comparison is given in the Japanese manuscript. Graph-based RAG systems such as GraphRAG focus on *which paths to retrieve* given a fixed graph, but generally do not provide an explicit, quantitative rule for *when to edit the graph itself*. Dynamic RAG variants (e.g., DyG-RAG, KEDKG) introduce time-aware and event-driven updates, but again lack a single gauge that unifies structural cost and information gain under equal-resources constraints. FEP- and MDL-inspired frameworks offer theoretical motivation for compression and free-energy minimization, but typically do not spell out concrete graph-edit rules under realistic compute budgets.

geDIG is designed to sit at the *control layer*: the target of control is the shape of the memory graph (add / rewire / prune), the decision criterion is the single scalar $\mathcal{F} = \Delta EPC_{norm} - \lambda \Delta IG_{norm}$, and the operation is split into Phase 1 (online, awake) and Phase 2 (offline, sleep) along the time axis. In that sense it can be read as a concrete, graph-level implementation of compression principles suggested by MDL, Information Bottleneck, and more recent formulations such as ITI/CEP, while remaining agnostic to the underlying encoder or LLM.

12 Threats to validity (summary)

We briefly summarize main threats to validity; the Japanese manuscript contains extended discussion and additional diagnostics. Internal validity is limited by the choice of grading model, embedder (Sentence-BERT), and prompts; although we use equal-resources comparisons and deterministic scripts, changes in these components could affect absolute numbers. External validity is limited by domain coverage, vocabulary, and scale: the maze PoC uses synthetic environments, and the RAG experiments use a lite 500-query suite with a single family of LLMs / encoders, so results may not transfer directly to other domains or larger graphs. Construct validity is constrained by our metric choices: Acc/EM/PER, FMR, PSZ, and ZSR capture important aspects of operation but do not exhaustively describe user-centric utility or long-term safety. Finally, implementation biases (e.g., particular approximate shortest-path algorithms, embedding approximations) may influence measured ΔSP and thereby \mathcal{F} ; we mitigate this via ablations and by keeping the code and scripts open for inspection and reproduction.

13 Limitations and future work

This work is explicitly a Phase 1 PoC under constrained compute and dataset size. Limitations include: modest scale (500-query lite for RAG; limited maze sizes), reliance on a single family of embeddings (Sentence-BERT) and a narrow range of LLMs, simple episode representations, and a focus on short- to mid-range multi-hop reasoning. Phase 2—offline rewiring under the FEP–MDL bridge, larger graphs, and long-horizon dynamics—is only outlined at a design level (section 14). Extending experiments to public QA benchmarks, diverse domains, and 1M+ node graphs, and running systematic λ -scans to search for phase-transition-like behavior, remain open for collaboration.

Why PSZ is not yet fully attained (short note) In the current dynamic RAG experiments, geDIG reduces PSZ shortfall s_{PSZ} compared to baselines but does not fully enter the PSZ band. This is mainly due to conservative hyperparameter choices (single λ calibration, percentile-based AG/DG thresholds), limited scale and domain diversity in the lite suites, and gate settings tuned for stability rather than exact PSZ optimality. We treat PSZ as an SLO-style goal rather than a hard requirement, and expect that larger-scale experiments and more adaptive gating policies can move operating points closer to PSZ without sacrificing robustness.

14 Phase 2 outlook: offline rewiring

Phase 2 is designed as an offline optimization phase that reuses Phase 1 logs to perform global improvements of the knowledge graph. Conceptually, it targets redundancy reduction, reconstruction of long-range bridges, and compression of over-connected regions, under constraints suggested by the FEP–MDL bridge. We sketch candidate objectives (e.g., approximate GED_{min} under resource constraints, MDL-style description length reductions, preservation of PSZ behavior under replayed traffic) and implementation considerations (incremental APSP / SSSP caches, anytime operation, safe rollback). A full treatment is left for future work; this section mainly clarifies how Phase 2 is intended to follow Phase 1 in a practical deployment.

15 Conclusion

We introduced geDIG, a unified gauge framework for dynamic knowledge graphs that combines normalized edit-path cost and information gain into a single scalar \mathcal{F} , and couples it with a two-stage gate (AG/DG). Across a partial-observation maze PoC and static / dynamic RAG experiments, we showed that this gauge can simultaneously drive exploration, integration, backtracking, and eviction while moving operating points toward a practically useful PSZ region. Our emphasis has been on operational reproducibility rather than formal optimality: code, scripts, percentile settings, and diagnostics are released so that readers can trace *when* the system worked. Phase 2 (offline rewiring) and larger-scale studies are intentionally left as open ground for collaboration.

Acknowledgments

The author thanks collaborators and early readers for feedback on the Japanese v5 manuscript and on the experiment design, and the broader community for open-source tools and datasets that made this work possible. Any remaining errors or misinterpretations are the author's responsibility. We invite collaboration and feedback from the community: this work is conducted by an independent researcher and welcomes reproduction, critical examination, and joint research opportunities on both the theoretical and experimental sides.