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-peeking* 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 $\Delta\mathrm{EPC}_{\mathrm{norm}}$ and information gain $\Delta\mathrm{IG}_{\mathrm{norm}}$, 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 $\mathcal{F} \propto \Delta\mathrm{MDL}+O(1/N)$ (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 (Acc $\geq 95\%$), erroneous merges are rare (FMR $\leq 2\%$), and additional latency due to dynamic control stays within a tolerable budget (P50 $\leq 200\,ms$).* Formally, the PSZ target is:

$$\mathrm{Acc} \;\geq 0.95, \quad \mathrm{FMR} \;\leq 0.02, \quad P50_{\Delta\mathrm{lat}} \;\leq 200\,\mathrm{ms}, \tag{1}$$

where $P50_{\Delta\mathrm{lat}}$ 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:

$$s_{\mathrm{PSZ}} \;=\; \max(0,\, 0.95-\mathrm{Acc}) \;+\; \max(0,\, \mathrm{FMR}-0.02) \;+\; \max\!\left(0,\, \tfrac{P50_{\Delta\mathrm{lat}}-200\,\mathrm{ms}}{200\,\mathrm{ms}}\right), \tag{2}$$

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 (Acc$\geq$ 95%, FMR$\leq$ 2%, additional P50$\leq$ 200 ms), reduced steps / redundant branches in the maze, and lower contamination / higher pending$\rightarrow$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 [3, 4, 8] 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 $\Delta\mathrm{EPC}_{\mathrm{norm}}$ and information gain $\Delta\mathrm{IG}_{\mathrm{norm}}{=}\Delta H_{\mathrm{norm}}{+}\gamma\,\Delta\mathrm{SP}_{\mathrm{rel}}$; 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).
- ΔEPC / ΔH / ΔSP: 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).
- ΔEPC / ΔH / ΔSP: 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

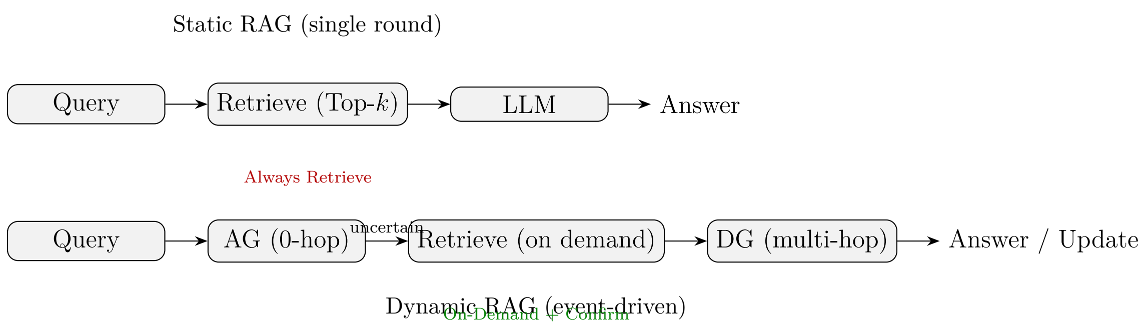

Figure 1: Static (single-round) vs dynamic (event-driven) RAG pipelines.

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 ($\Delta\mathrm{EPC}_{\mathrm{norm}} - \lambda\,\Delta\mathrm{IG}_{\mathrm{norm}}$)
g_0	0-hop evaluation (immediately after tentative wiring)
$g^{(h)}$	h-hop evaluation ($h \in \{1, \ldots, H\}$)
$g_{\min}$	$\min_{1\le h\le H} g^{(h)}$
$b(t)$	Operational gauge ($\min\{g_0, g_{\min}\}$)
θ_{AG}, θ_{DG}	Quantile-based thresholds (AG/DG)
$\Delta\mathrm{EPC}_{\mathrm{norm}}$	Normalized edit-path cost (Phase 1 operational GED)
$\Delta\mathrm{IG}_{\mathrm{norm}}$	Normalized IG difference ($\Delta H_{\mathrm{norm}}+\gamma\,\Delta\mathrm{SP}_{\mathrm{rel}}$)
$c(\cdot)$	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:

$$\Delta\mathrm{IG}_{\mathrm{norm}} \;=\; \Delta H_{\mathrm{norm}} \;+\; \gamma\,\Delta\mathrm{SP}_{\mathrm{rel}}, \qquad \gamma > 0. \tag{3}$$

Note (default parameter). In our experiments we set γ=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

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

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 $\Delta\mathrm{EPC}_{\mathrm{norm}}$, ΔH_{norm}, and $\Delta\mathrm{SP}_{\mathrm{rel}}$.

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:

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

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

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

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

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

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

Structural integration is measured as relative gain in path length:

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

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 $g_0 > \theta_{\mathrm{AG}}$, the system deepens exploration (AG). If $\min\{g_0, g_{\min}\} \le \theta_{\mathrm{DG}}$, 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 ($\theta_{\mathrm{AG}}, \theta_{\mathrm{DG}}$) 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.

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 $\to$ 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 φ : text $\to \mathbb{R}^d$	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.

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

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** $\mathrm{GED}_{\min}$. Exact computation of $\mathrm{GED}_{\min}$ is in general **NP-hard** (e.g., [6]), and jointly optimizing $\Delta\mathrm{EPC}$ and $\Delta\mathrm{IG}$ 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 $\mathcal{F} = \Delta\mathrm{EPC}_{\mathrm{norm}} - \lambda\Delta\mathrm{IG}_{\mathrm{norm}}$ 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 13.2 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 [3, 4], 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.

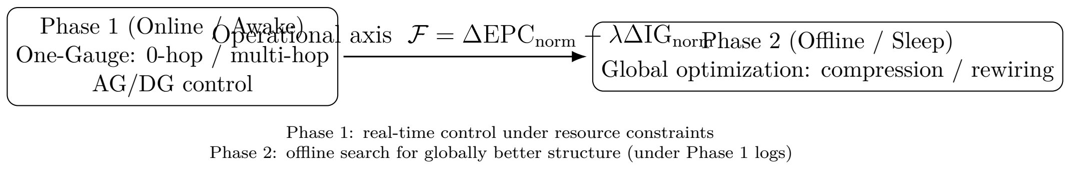
Phase 1 (Online / Awake)
One-Gauge: 0-hop / multi-hop
AG/DG control
Operational axis $\mathcal{F} = \Delta\mathrm{EPC}_{\mathrm{norm}} - \lambda\Delta\mathrm{IG}_{\mathrm{norm}}$
Phase 2 (Offline / Sleep)
Global optimization: compression / rewiring
Phase 1: real-time control under resource constraints
Phase 2: offline search for globally better structure (under Phase 1 logs)

Figure 2: Two-phase architecture (awake / sleep) as a design metaphor.

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 $\mathcal{F} = \Delta\mathrm{EPC}_{\mathrm{norm}} - \lambda\Delta\mathrm{IG}_{\mathrm{norm}}$ 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 $\Delta\mathrm{EPC}_{\mathrm{norm}}$ and $\Delta\mathrm{IG}_{\mathrm{norm}}$, forcing per-domain retuning of gate thresholds and percentiles. While $\Delta\mathrm{SP}_{\mathrm{rel}}$ 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 $\Delta\mathrm{EPC}_{\mathrm{norm}}$ and $\Delta\mathrm{IG}_{\mathrm{norm}}$. 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 [9], 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 [9]. 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

Summary (results-first)

The goal of this PoC is to quantify **exploration efficiency in unknown mazes** under partial observability. In representative settings (15/25/50×15 mazes), geDIG maintains high success rates (roughly 95–100% on easier configurations) while reducing the number of steps by about 20–30% compared to baselines, and pruning roughly 95% of candidate edges. Classical planners such as Dijkstra and A* serve as *ideal upper bounds* for reference (Regret, SPL, frontier order), not as direct competitors.

For a 25×25 maze (L3-only, max steps = 250), typical numbers include:

- **Exploration ratio** (unique / total): ≈ 0.26 (average over seeds).
- **Revisit ratio** (steps / unique): ≈ 1.3.
- **Average backtrack length**: around 1.0 step (as an approximation of dead-end run length).
- **Dead-end detection delay**: near 0 (effectively immediate).
- **Success rate**: high on 15×15 and 25×25 with generous budgets; lower on harder 25×25 configurations with tight budgets.
- **Reference proximity**: Regret and SPL remain close to Dijkstra/A* bounds, and the frontier ranking induced by $-\mathcal{F}$ correlates well with Dijkstra priority.

Design-wise, we verify that: (i) **Unified-gauge control** improves combined metrics (success, steps, compression) relative to EPC-only / IG-only controls; (ii) **Two-stage gating** behaves as intended (division of labor between 0-hop and multi-hop, stable AG/DG rates and DG/AG ratio); and (iii) **Query-centric multi-hop evaluation** with small $H \in \{1, 2, 3\}$ balances shortcut detection and computational cost. Together, these observations support the view that g_0 *detects stagnation early*, and *DG steers the agent back to promising branches*, thereby reducing unnecessary wandering.

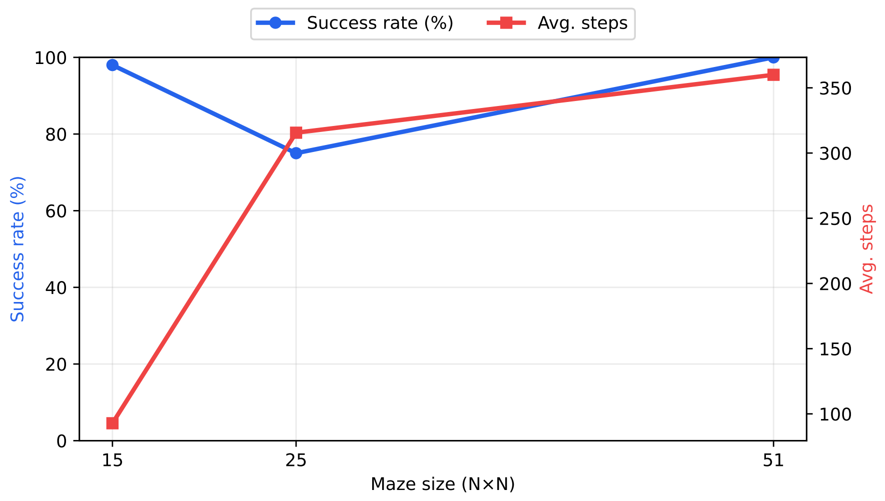

Figure 3: Maze scaling (L3; max steps 250/500/1000). Success rate (left axis) and average steps (right axis) as a function of maze size. For 15×15 (max steps 250) and 25×25 (max steps 500), geDIG maintains high success while keeping steps low; for 51×51, larger budgets (max steps 1000) recover success.

In this chapter we treat the partial-observation maze as a small, fully controllable world-model and ask whether a dynamic KG can, under a minimal configuration (8-dimensional state vectors, partial observability, incremental wiring), use the **single gauge** $\mathcal{F}$ and **two-stage gate** (AG/DG) to simultaneously govern:

1. selection among new episodes (accept / hold / reject), and
2. exploitation of the episodic memory graph (explore / backtrack / reuse).

The key design elements are:

1. **Unified-gauge control**: a single scalar $\mathcal{F} = \Delta\mathrm{EPC}_{\mathrm{norm}} - \lambda\Delta\mathrm{IG}_{\mathrm{norm}}$ that drives event-based decisions over candidate edges and exploration branches.
2. **Two-stage gate (AG/DG)**: 0-hop (novelty / ambiguity) detects stagnation; multi-hop evaluation confirms shortcuts and triggers backtracking / integration.
3. **Query-centric multi-hop evaluation**: k-hop subgraphs centered on the current episode, providing scalable EPC/IG evaluation with local computations.

5.1 Environment and episodes

We use grid mazes of sizes 15×15, 25×25, and 50×50 with partial observations: the agent sees only a local 3×3 window around its current position. The underlying maze layout (walls, goal position) is unknown; the agent must discover it by moving. States, actions, and outcomes are embedded into an 8-dimensional episode vector (one of the simplest nontrivial encodings that satisfies the embedding requirements in section 4.3). An episodic graph G_t is built online: visited positions become nodes, transitions become edges, and episode vectors are stored as node or edge attributes.

At each time step t the agent:

- observes a local patch around the current position p_t,

- constructs an episode vector $v_t \in \mathbb{R}^8$ (encoding normalized position, wall configuration, visit counts, etc.),
- tentatively integrates v_t into the episodic graph,
- evaluates candidate connections and updates via the gauge $\mathcal{F}$ and gates AG/DG.

The exact coordinate encoding and weight vector are domain-specific to the maze, but the evaluation-and-gating logic is domain-agnostic.

5.2 Gauge-based control in the maze

We use the same gauge as in the RAG setting:

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

Here $\Delta\mathrm{EPC}_{\mathrm{norm}}$ is the normalized edit-path cost of the episode edits actually applied in the local graph, and $\Delta\mathrm{IG}_{\mathrm{norm}}$ is the normalized information gain (entropy decrease plus path shortening) on a query-centric induced subgraph. We interpret negative $\mathcal{F}$ as beneficial (strong consolidation with modest structural cost).

At runtime we separate:

- **0-hop evaluation** $g_0(t)$: local structural and entropy change immediately after tentative wiring; high g_0 indicates ambiguous or unhelpful edits.
- **Multi-hop evaluation** $g^{(h)}(t)$**,** $g_{\min}$: evaluation over h-hop induced subgraphs, detecting structural shortcuts (shorter paths, better connectivity).

AG is triggered when $g_0 > \theta_{\mathrm{AG}}$ (stagnation / ambiguity), and DG is triggered when $\min\{g_0, g_{\min}\} \leq \theta_{\mathrm{DG}}$ (confirmed gain). Percentile-based thresholds $\theta_{\mathrm{AG}}, \theta_{\mathrm{DG}}$ are calibrated on burn-in trajectories.

5.3 Experimental design

We use grid mazes with random wall patterns and a single start/goal pair. The agent starts at a fixed or random start cell and aims to reach the goal within a maximum number of steps (step budget scaled with maze size). The PoC uses simple policies on top of the gauge:

- actions are sampled from a softmax over distances to known episodes (greedy toward promising frontier states),
- backtracking is handled via BFS to the last branching point when DG confirms a shortcut,
- eviction (if enabled) uses low-confidence and old episodes as eviction candidates.

We do not perform policy learning or value-function optimization: the goal is to test *operational consistency* of recall and integration under the gauge, not to optimize a general RL policy.

5.4 Metrics and success criteria

Primary metrics focus on exploration efficiency:

- **Exploration ratio**: unique visited cells / total steps.
- **Revisit ratio**: steps / unique cells.
- **Average backtrack length**: average length of backtrack segments (AG→DG).
- **Dead-end detection delay**: steps between entering a dead-end and AG firing.
- **Success rate**: fraction of episodes reaching the goal within the step budget.

Secondary metrics quantify path quality: Regret = steps $-L^*$ (difference to shortest path), SPL = $L^*/\max\{L^*, \text{steps}\}$ (closer to 1 is better). Diagnostics 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).

5.5 Baselines and ablations

Under identical partial-observation and wiring conditions we compare: Greedy Novelty (least-visited-first), ε-greedy (e.g., 90% unvisited / 10% random), UCB1-like novelty scores, partially observed A* (A* on the known region plus exploration toward the unknown boundary), and diagnostic oracles (Dijkstra/A* with full map). We also run ablations of geDIG: EPC-only (sending $\lambda \to \infty$), IG-only (ignoring EPC), no AG/DG (always accept / always reject), and 0-hop-only (H=0).

Overall, geDIG improves exploration efficiency and path quality relative to these baselines while keeping AG/DG rates in a controlled range.

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 $(\theta_{\mathrm{AG}}, \theta_{\mathrm{DG}})$ 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 EM≈ 0.00, PER≈ 0.17, whereas geDIG G1 reaches EM≈ 0.25, PER≈ 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.

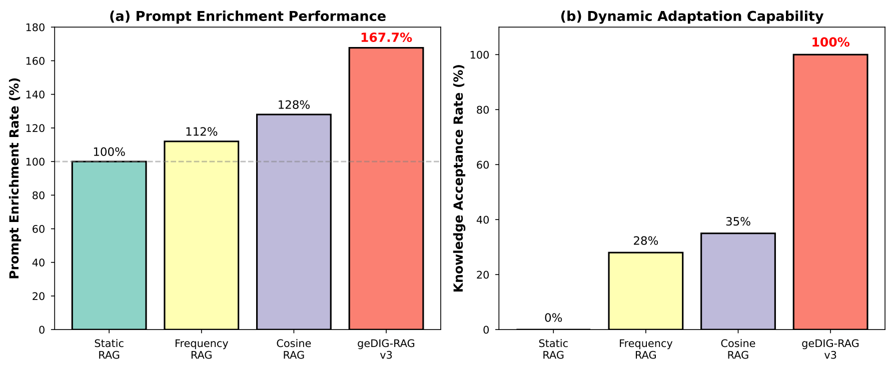

Figure 4: Representative static RAG performance under equal-resources (lite 500-query suite). geDIG-soft improves EM / PER and citation faithfulness over the strongest static baseline while keeping latency comparable.

Table 5: Static vs dynamic RAG (lite 500-query run; equal-resources). PSZ shortfall s_{PSZ} summarizes how far each method is from the PSZ band (Acc≥ 0.95, FMR≤ 0.02, P50$\leq 200\,\text{ms}$).

Method	PER	Acc	ZSR	FMR	P50 (ms)	P95 (ms)	s_{PSZ}
static_rag	0.172	0.000	1.000	1.000	160.0	160.0	1.930
frequency	0.172	0.000	1.000	1.000	160.0	160.0	1.930
cosine_topk	0.172	0.000	1.000	1.000	160.0	160.0	1.930
gedig_ag_dg	0.421	0.374	1.000	0.626	240.0	240.0	1.222

Table 6: 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 [7], DyG-RAG [1], KEDKG [2], 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

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

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 7 summarizes the desired directions on each metric axis; we follow the same conventions throughout.

Table 7: 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 $\leq 200\,\text{ms}$)
Contamination rate p_{contam}	Fraction of updates later rolled back	$\downarrow$
Promotion rate	Pending$\to$confirmed transitions	$\uparrow$
Maze steps	Exploration steps in the PoC	$\downarrow$
Shortcut contribution $\Delta\text{SP}_{\text{rel}}$	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$\to$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.

7.1 Setup (events, updates, and time)

We follow a Dynamic GRAG-style setting with incremental updates and temporal consistency:

- **Data stream.** Time-stamped documents (e.g., news, specs, change logs) arrive as episodes with timestamps t and optional durations Δt.
- **Updates.** Each episode is tentatively integrated: a local subgraph is built, merged into the main KG via incremental wiring (high-gain edges strengthened, low-gain edges weakened or pruned), and logged with its gauge trajectory.
- **Retrieval.** Time-consistent retrieval follows DyG-RAG / temporal-RAG practices: queries are matched against time-aware neighborhoods, and geDIG-soft applies continuous weakening via $\sigma(\tau\mathcal{F})$ on top of the base retriever.

- **Generation.** We use temporally aware prompts (Temporal CoT-style) that encourage the model to surface time-resolved evidence paths and to always return citations.
- **Sleep phase (offline).** In Phase 2, we replay logs of $\mathcal{F}$ and AG/DG decisions over batches to further down-weight persistently low-$\mathcal{F}$ regions and to promote consistently high-gain chains; this is sketched as an outlook rather than fully implemented.

7.2 Dynamic metrics and success criteria

Beyond EM/F1 and faithfulness, dynamic runs track:

- **Acceptance rate (Acc).** Fraction of updates (pending/confirmed) that are ultimately judged correct; PSZ targets Acc$\geq$ 95%.
- **False Merge Rate (FMR).** Among accepted episodes, the fraction later judged erroneous merges; PSZ targets FMR$\leq$ 2%.
- **Additional P50 latency.** Median extra latency relative to static RAG; PSZ targets $\mathrm{P50}_{\Delta\mathrm{lat}} \leq 200\,\mathrm{ms}$.
- **PSZ shortfall s_{PSZ}.** Aggregated deficit from the PSZ band as defined in eq. (2).
- **ZSR (Zero-Search Rate).** Fraction of queries answered at 0-hop (no AG firing), indicating how often the current KG suffices without additional retrieval.
- **Temporal Consistency and update lag.** Stability of answers and evidence as the KG evolves, and delay from ingest to availability.
- **AG/DG firing rates.** Frequency of AG and DG events over the query stream, used as diagnostics for gating behavior.

As a practical success criterion we aim either to enter the PSZ band or to *consistently reduce* s_{PSZ} while keeping answer quality and temporal consistency acceptable.

7.3 Time-series analysis and operating curves

Following the evaluation protocol in sections 4 and 6.1, we analyze AG/DG behavior over query indices and inspect operating curves:

- **AG/DG time-series.** We plot AG and DG firing rates as functions of query index; under scale-up, AG tends to increase (more ambiguity in larger KGs) while DG remains in a moderate band (roughly a few percent), consistent with the intended division of labor.
- **Operating curves.** By sweeping $(\theta_{\mathrm{AG}}, \theta_{\mathrm{DG}})$ we obtain 3D operating curves over (Acc, FMR, additional P50). Regions with low s_{PSZ} correspond to operating points that are close to the PSZ band, even if full attainment is not yet realized.

These analyses, together with the logs of ΔEPC, ΔH, ΔSP, and $\mathcal{F}$, give an operational view of *when* the system chose to explore, backtrack, or commit, and how AG/DG behavior changes as the KG grows.

Time-series and logs We log ΔEPC, ΔH, ΔSP, and $\mathcal{F}$ together with acceptance decisions (pending→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.

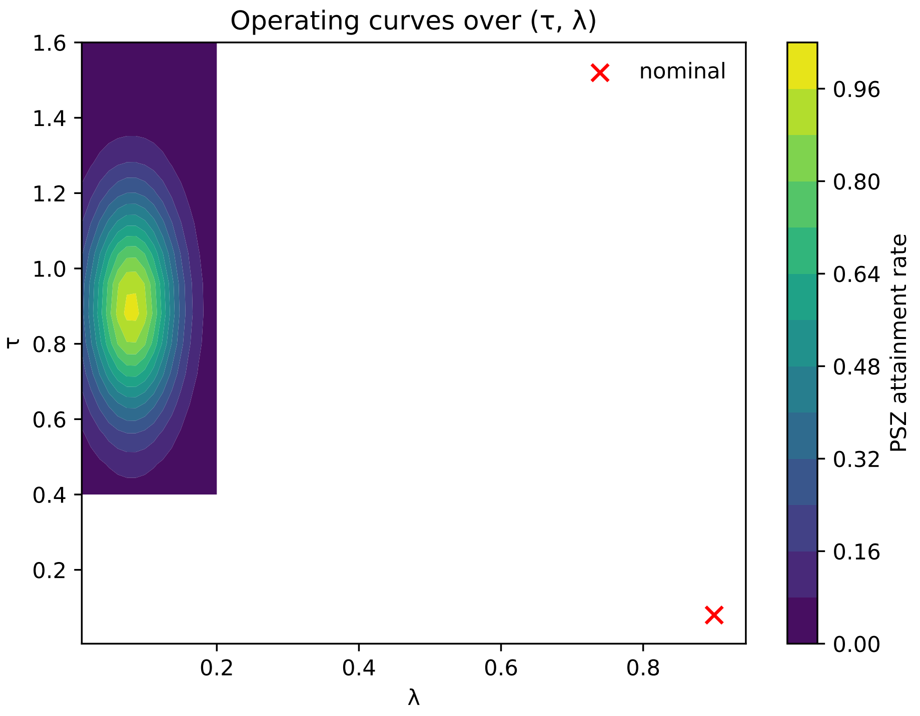

Figure 5: Example operating curves for dynamic RAG (lite 500-query run). Sweeping $(\theta_{\mathrm{AG}}, \theta_{\mathrm{DG}})$ traces trade-offs among Acc, FMR, and additional latency; PSZ-compatible regions correspond to low PSZ shortfall s_{PSZ}.

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 $\Delta s = s_{\mathrm{support}} - s_{\mathrm{random}} > 0$ 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.

Table 8: Alignment summary (lite 500-query run). Support vs random subgraphs show a positive Δs with large effect size and statistically significant sign tests.

Metric	Value	Note
$s_{support}$	0.0190	mean
s_{random}	-0.0019	mean
s_{topk}	0.0190	mean
$s_{AG\text{-}pick}$	0.0190	mean
$\Delta s_{support-random}$	0.0209	sign-test $p = 5.24e - 09$
Cohen's $d_{support-random}$	1.019	95% CI [0.0149, 0.0267]

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.

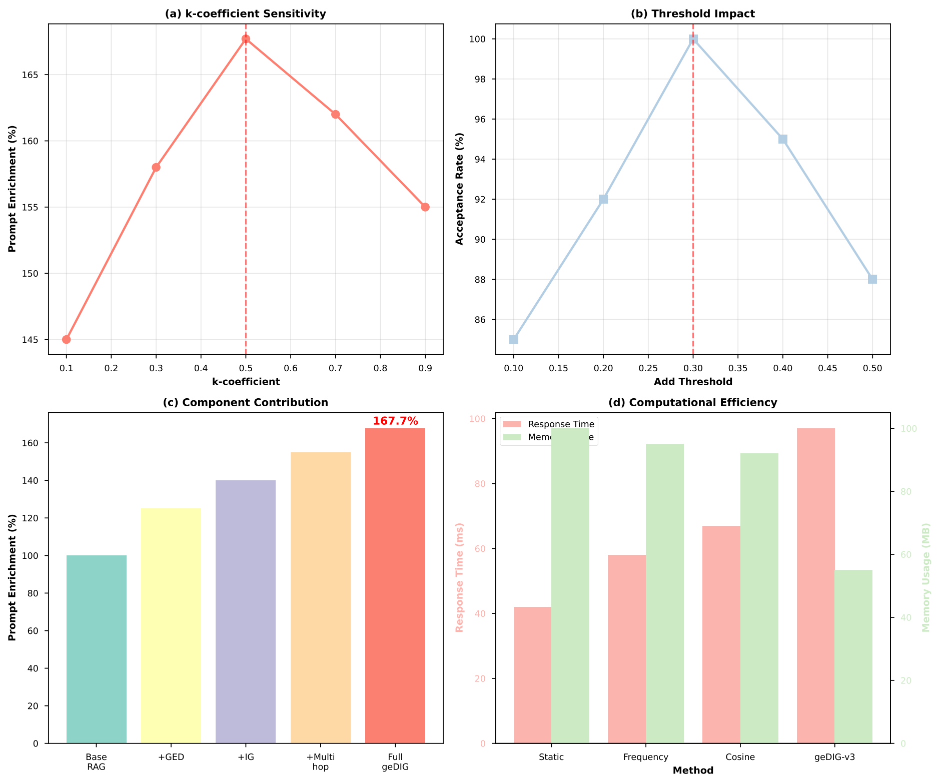

Figure 6: Representative ablation results (lite 500-query RAG). Each component of $\mathcal{F}$ and the AG/DG gates contributes materially: EPC-only and IG-only variants degrade robustness or consolidation, while removing ΔSP or gates harms multi-hop path quality and operating points.

Table 9: Ablation summary (lite 500-query dynamic RAG). Variants that remove EPC, IG, ΔSP, or gates show degraded acceptance profiles or latency compared to the base configuration.

variant	per_mean	acceptance	fmr	lat_p50	lat_p95
base	0.4207	0.374	0.626	240.0	240.0
epc_only	0.4207	0.374	0.626	240.0	240.0
hop0_only	0.4207	0.374	0.626	240.0	240.0
ig_emphasis	0.4207	0.374	0.626	240.0	240.0

10 FEP–MDL bridge (operational proposition)

Scope. This section provides an *operational* 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. We treat the Free Energy Principle (FEP) and Minimum Description Length (MDL) as sources of design intuition rather than as frameworks to which we claim formal equivalence.

Background: FEP and MDL FEP (in the sense of Friston and others [5]) frames adaptive behavior as minimization of a variational free-energy bound on surprise; MDL frames learning as minimization of the sum of model and data code lengths. In both views, there is a trade-off between *complexity* (model / structure cost) and *accuracy* (fit to data). Our gauge $\mathcal{F} = \Delta\mathrm{EPC}_{\mathrm{norm}} - \lambda\Delta\mathrm{IG}_{\mathrm{norm}}$ can be read as an operational instantiation of this trade-off at the level of a dynamic knowledge graph, where:

- $\Delta\mathrm{EPC}_{\mathrm{norm}}$ plays the role of a structural complexity term (edit-path cost of the applied updates),
- $\Delta\mathrm{IG}_{\mathrm{norm}} = \Delta H_{\mathrm{norm}} + \gamma\Delta\mathrm{SP}_{\mathrm{rel}}$ plays the role of a data/accuracy improvement term (entropy decrease plus path shortening),
- λ acts as an information temperature-like coefficient that balances the two.

We do *not* claim that $\mathcal{F}$ is a variational bound in the FEP sense or that it equals a particular MDL objective; instead, we use these analogies to reason about design choices and to interpret experimental observations.

Operational correspondence: definition We call a relation an *operational correspondence* if it satisfies three conditions:

1. it is proportional rather than exact (e.g., up to a positive scale factor),
2. residual terms are bounded, e.g., $O(1/N)$ under explicit assumptions, and
3. it yields concrete, testable predictions (e.g., about ablations or metric behavior).

In our case we consider a correspondence of the form

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

under assumptions about normalization, bounded horizons, decomposable edits, stable entropy estimation, and embedding conditions as in section 4.3. The proportionality constant is absorbed into λ (interpreted as a scale-anchoring, temperature-like parameter).

Reading guide This section is best read as a proof sketch tying together:

- the **definition** of the gauge $\mathcal{F} = \Delta\mathrm{EPC}_{\mathrm{norm}} - \lambda\Delta\mathrm{IG}_{\mathrm{norm}}$ (section 3),
- the **duality intuition** between 0-hop (error / ambiguity; FEP-side) and multi-hop (compression / shortcuts; MDL-side),
- the **gate mechanism** (AG/DG) as an engineering realization of this split (section 3.4),
- the **embedding assumptions** (A1–A3) in section 4.3.

The main question is: under what assumptions can we interpret $\mathcal{F}$ as proportional to a change in an MDL-style objective, and what does this imply for using a single control signal to govern both structure and information?

Implications If the correspondence $\mathcal{F} \propto \Delta\text{MDL}+O(1/N)$ holds under these assumptions, it suggests:

- **Justification for a single control signal.** It is reasonable to use one scalar $\mathcal{F}$ to jointly control structure edits (EPC) and information consolidation (ΔH, ΔSP) rather than separate, potentially conflicting signals.
- **Interpretation of λ.** The coefficient λ can be read as an information temperature, analogous to the ratio between data and model code-length scales in MDL.
- **Orthogonality and double-counting.** By keeping ΔEPC on the structural side and ΔIG on the information side, we avoid double-counting structural effects in the information term, which improves interpretability.
- **Alignment with ablations.** Ablations in section 9 (removing ΔH or ΔSP) show behavior consistent with the idea that these terms contribute to an MDL-like compression objective.

At the same time, this is explicitly an *operational* proposition: we do not claim that $\mathcal{F}$ is the unique or correct MDL/FEP objective, only that it behaves in a way that is consistent with these principles under the given assumptions.

Assumptions (B1–B4) and their intent To reason about $\mathcal{F}$ as proportional to ΔMDL we rely on the following assumptions, adapted from the Japanese manuscript:

(B1) Local boundedness Edit costs are bounded by $C_{\max}$ and the multi-hop horizon H is finite; this anchors normalization and prevents unbounded growth of $\Delta\text{EPC}_{\text{norm}}$.

(B2) Edit decomposability Substitution operations can be upper-bounded by delete+insert pairs, so GED can be approximated as a sum of additive edit costs.

(B3) Entropy-estimation stability The variance of local entropy-difference estimates scales as $\sigma_H^2 = O(1/|\mathcal{N}|)$ for neighborhood size $|\mathcal{N}|$; we use estimators with uniform convergence.

(B4) Normalization stability The normalization bases $C_{\max}(G)$ and $\log K(G)$ vary slowly over time so that their fluctuations can be absorbed into a scalar factor (and treated as $O(1/N)$ residuals).

These assumptions correspond to the embedding requirements (A1)–(A3) in section 4.3: semantic gradient preservation (A1) supports stable probability distributions for ΔH, scale normalization (A2) stabilizes $C_{\max}$ and $\log K$, and local smoothness (A3) keeps edits and neighborhood changes well behaved.

Table 10: Correspondence between assumptions (B1–B4) and embedding requirements (A1–A3), summarized at a high level.

Assumption	Purpose	Supported by
B1: Local boundedness	Fixes normalization base and keeps costs finite	A2 (scale normalization), A3 (local smoothness)
B2: Edit decomposability	Enables additive GED approximation	Implementation of edit model (substitution $\approx$ delete+insert)
B3: Entropy estimation	Controls variance, ensures convergence	A1 (semantic gradients) for stable probabilistic neighborhoods
B4: Normalization stability	Absorbs drift in $C_{\max}$, $\log K$	A2 (unit norms), A3 (smooth changes)

Proposition (summary) and proof sketch Under assumptions (B1)–(B4) and the normalization conventions in section 3, we can summarize the main proposition as:

$$\mathcal{F} \;=\; \Delta\mathrm{EPC}_{\mathrm{norm}} - \lambda\Delta\mathrm{IG}_{\mathrm{norm}} \;\propto\; \Delta\mathrm{MDL} + O(1/N).$$

MDL decomposes into a model term $L(M)$ and a data term $L(D \mid M)$. Using (B2) and (B1), the model term can be upper-bounded as $c_{\mathrm{ged}}\Delta\mathrm{EPC}_{\mathrm{norm}} + O(1/N)$ for some constant c_{ged}. Using (B3) and (B4), the data term can be approximated as $-c_{\mathrm{ig}}\Delta\mathrm{IG}_{\mathrm{norm}} + O(1/N)$ with c_{ig} capturing the scale of entropy / path contributions. Choosing $\lambda \approx c_{\mathrm{ig}}/c_{\mathrm{ged}}$ aligns scales so that $\mathcal{F}$ becomes proportional to ΔMDL up to an $O(1/N)$ residual. This is not a formal theorem but an operational statement: it justifies reading $\mathcal{F}$ as a proxy for MDL-style change under the stated assumptions, and it matches the behavior observed in ablations.

Thermodynamic reading (metaphor) Following the Japanese manuscript, we can also offer a free-energy-style reading as a metaphor. We define

$$U := \Delta\mathrm{EPC}_{\mathrm{norm}} - \lambda\gamma\,\Delta\mathrm{SP}_{\mathrm{rel}}, \quad S := \Delta H_{\mathrm{norm}}, \quad F := U - \lambda S, \tag{8}$$

so that F is isomorphic to $\mathcal{F}$ by rearrangement. Here U is read as a structural energy term (edit cost minus path-efficiency gains), S as an entropy term (ordering), and F as a free-energy-like objective. This mapping is purely heuristic and is not required for using geDIG; it simply offers an additional lens for interpreting λ as kT and for thinking about phase-transition-style behavior (e.g., under λ-scans).

Open theoretical questions A full, rigorous treatment of the MDL correspondence and of the thermodynamic analogy is beyond the scope of this paper. Open questions include generalizing assumptions (e.g., to directed or labeled graphs), tightening bounds, connecting to information thermodynamics, and deriving sharper predictions about phase transitions in graph structure as λ varies. We present this section as an invitation for theoretical collaboration rather than as a completed theory.

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 $\mathcal{F} = \Delta\mathrm{EPC}_{\mathrm{norm}} - \lambda\Delta\mathrm{IG}_{\mathrm{norm}}$, 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.

Table 11: High-level comparison with related approaches (performance / control / theory layers).

Method	Dynamic KG	Structure detection	Single gauge	Joint learn/use	Insight events	Theory basis
GraphRAG	✓	△	×	×	×	–
DyG-RAG	✓	△	×	△	×	–
KEDKG	✓	×	×	△	×	–
FEP / Active Inf.	×	×	✓	✓	△	FEP
MDL / IB	×	△	✓	×	△	MDL
geDIG	✓	✓	✓	✓	✓	FEP–MDL (operational)

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 13.2). 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.

13.1 Why PSZ is not yet fully attained

Although geDIG reduces PSZ shortfall s_{PSZ} compared to baselines, our current dynamic experiments do not fully enter the PSZ band. We see three main contributing factors.

First, **hyperparameter tuning is deliberately conservative**. We calibrate λ once on a lite development suite and then keep it fixed across runs, and we use simple percentile rules for $(\theta_{\text{AG}}, \theta_{\text{DG}})$ to avoid overfitting. This avoids overly optimistic operating points at the cost of leaving some room for improvement in Acc/FMR trade-offs. Systematic λ- and γ-sweeps under stricter train/val/test splits are left for future work.

Second, **experimental scale and diversity are limited**. The dynamic RAG experiments use a 500-query lite suite with a single family of encoders and LLMs; under such constraints, PSZ is a relatively strict target. Larger and more diverse query sets, and more aggressive exploration of DG thresholds under equal-resources constraints, may identify regions that enter PSZ more fully.

Third, **gate thresholds and time-series behavior are tuned for stability rather than PSZ optimality**. In our current design, AG/DG rates are kept in moderate bands to avoid oscillatory or brittle behavior; this sometimes leaves FMR slightly above the PSZ target or P50 slightly above 200 ms. We view PSZ as an SLO-style goal rather than a hard requirement; future work will explore adaptive thresholding and more nuanced per-query-type gating policies to move operating points closer to the PSZ band without sacrificing robustness.

13.2 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 $\mathrm{GED}_{\mathrm{min}}$ 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.

Edge feature expansions and Phase 2 use In Phase 2 we envision using not only scalar edge weights but a richer *edge feature vector* to support multi-faceted optimization. For each edge $e \in E$ we attach a feature vector

$$\mathbf{f}(e) \in \mathbb{R}^d$$

that summarizes how the edge behaves under queries and updates. Examples include:

- **Co-occurrence / frequency features**: co-occurrence counts, per-query joint activation frequencies, temporal co-activation patterns (edges that tend to be active in the same time windows).
- **Embedding / geometric features**: similarity between node embeddings, contribution to $\Delta\mathrm{SP}_{\mathrm{rel}}$ when the edge is added or removed, changes in local clustering coefficients.
- **Model / attention features**: LLM attention weights or gradient-based saliency, number of times the edge was actually cited, frequency of appearance in chains-of-thought.
- **Operational log features**: AG/DG passage counts, histories of pending$\rightarrow$confirmed / rollback, contributions to FMR, flags for long-unused edges, etc.

In Phase 1 we primarily *log* $\mathbf{f}(e)$ while using $\mathcal{F}$ and AG/DG for real-time decisions. Phase 2 then uses $\mathbf{f}(e)$ as input to:

- **Edge scoring**: learn a simple scorer $s(e) = \mathbf{w}^\top \mathbf{f}(e)$ (or a shallow model) from acceptance / rejection labels and FMR logs, and rank edges as "keep", "prune", or "rewire candidate".
- **Multi-objective optimization**: augment the Phase 2 objective with terms that capture patterns in $\mathbf{f}(e)$, such as "frequently used but contamination-prone edges" or "rarely used but structurally important bridge edges".
- **Edge-type-specific policies**: design differentiated policies, e.g. keep temporally consistent edges more aggressively than pure co-occurrence edges, or treat edges with stable high attention weights as bridge candidates with lower effective cost.

Thus Phase 1 performs local edits in real time based on $\mathcal{F}$ and AG/DG, while Phase 2 uses multi-channel edge features $\mathbf{f}(e)$ to perform *global compression, rewiring, and weight updates.* Both phases share the same gauge $\mathcal{F}$ and logs (AG/DG histories), but operate at different time scales and with different objectives (online operation vs offline optimization).

Roadmap and collaboration Looking ahead, we see several concrete directions. First, we plan to **scale Phase 1** to larger and more diverse query suites (5k–10k+ queries across many domains) and systematically explore λ, γ, and gate thresholds under stricter train/validation/test splits. Second, we aim to **operationalize Phase 2** by implementing offline rewiring pipelines that replay logs at scale, tune edge-feature-based objectives, and stress-test PSZ behavior under long-running workloads. Third, we would like to **add transformer-based validation experiments**, including alternative retrievers / LLMs and controlled perturbation studies, to probe how robust the gauge and gates are to model changes. Finally, we intend to **extend geDIG to other domains** (e.g., software engineering corpora, scientific literature, enterprise knowledge graphs) and to more varied forms of "insight" events. These directions are intentionally left open for collaboration: we welcome joint work and co-authorship on both theoretical tightening (FEP–MDL, optimality) and large-scale empirical validation.

14 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.

References

[1] Authors. Dynamic graph-based retrieval-augmented generation. arXiv preprint, 2024. Preprint; DOI/URL to be updated.

[2] Authors. Knowledge editing with dynamic knowledge graphs. arXiv preprint, 2024. Preprint; DOI/URL to be updated.

[3] Gy"orgy Buzsáki. *Rhythms of the Brain.* Oxford University Press, 2011.

[4] Matthew F Carr, Shantanu P Jadhav, and Loren M Frank. Hippocampal replay in the awake state: a neural substrate of spatial memory. *Nature Neuroscience*, 14(2):147–153, 2011.

[5] Karl Friston. The free-energy principle: a unified brain theory? *Nature Reviews Neuroscience*, 11(2):127–138, 2010.

[6] Xinbo Gao, Bing Xiao, Dacheng Tao, and Xuelong Li. A survey of graph edit distance. *Pattern Analysis and applications*, 13(1):113–129, 2010.

[7] Microsoft Research Team. Graphrag: Retrieval-augmented generation via knowledge graphs. arXiv preprint, 2024. Whitepaper/technical report; DOI/URL to be updated.

[8] Brad E Pfeiffer and David J Foster. Hippocampal place-cell sequences depict future paths to remembered goals. *Nature*, 497:74–79, 2013.

[9] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of EMNLP-IJCNLP*, 2019.