

geDIG：自由エネルギー原理に基づく知識グラフ自律更新の統一ゲージ

geDIG: A Unified Gauge for Autonomous Knowledge Graph Updates Based on the Free Energy Principle

宮内 和義^{*1}

Kazuyoshi Miyauchi

^{*1}独立研究者

Independent Researcher

Dynamic knowledge graphs need a criterion for when to accept new edges. We propose geDIG, a unified gauge that operationally bridges the Free Energy Principle and the Minimum Description Length by trading off structural edit cost and information gain: $\mathcal{F} = \Delta\text{EPC}_{\text{norm}} - \lambda(\Delta H_{\text{norm}} + \gamma\Delta\text{SP}_{\text{rel}})$. We apply geDIG to a partially observable maze task with a two-stage gate: AG (0-hop ambiguity) and DG (multi-hop shortcut). The agent reaches goals with success rates of 0.98 (15x15) and 0.75 (25x25), while 51x51 drops to 0.46 under a step cap. Time-series of g_0/g_{\min} show gate activations aligned with exploration and integration, and snapshots indicate compact graph growth. These results suggest geDIG provides a practical criterion for selective knowledge integration.

1. はじめに

動的に成長する知識グラフでは「何を取り込むか (What)」だけでなく、「いつ受け入れるか (When)」の規範が重要である。知識導入の基準はしばしば設計者のヒューリスティックに依存し、内発報酬もタスク依存に定義されることが多く、そのため運用上は When 判断が明示化されないことが多く、冗長化や誤統合につながりやすい。本研究では、構造コストと情報利得を単一スカラーで統合する geDIG を提案し、When 判断の統一ゲージを与える。言語の影響を避けるため、部分観測迷路を用いた世界モデル実験で原理検証を行う。

本研究の貢献は以下の3点である：

1. FEP[1] と MDL[2] を操作的に橋渡しする geDIG の定式化
2. 0-hop/多段 hop を使い分ける AG/DG 二段ゲートによる統合制御
3. 迷路環境における探索・統合の同時制御の実証 (15×15 ~ 51×51)

2. 提案手法：geDIG と二段ゲート

2.1 geDIG ゲージ

geDIG は以下で定義される：

$$\mathcal{F} = \Delta\text{EPC}_{\text{norm}} - \lambda(\Delta H_{\text{norm}} + \gamma\Delta\text{SP}_{\text{rel}}) \quad (1)$$

(1) 各項の意味

- $\Delta\text{EPC}_{\text{norm}}$ ：編集経路コスト（構造更新の複雑さ）
- ΔH_{norm} ：エントロピー差（曖昧さの変化）
- $\Delta\text{SP}_{\text{rel}}$ ：平均最短路の相対改善（統合利得）

小さい \mathcal{F} ほど「低コストで高利得な統合」と解釈する。

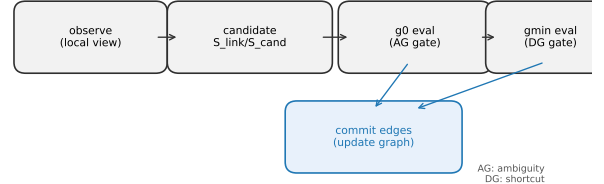


図 1: AG/DG 二段ゲートの制御フロー。0-hop で曖昧さを検知し、multi-hop で短絡効果を確認する。

2.2 AG/DG 二段ゲート

(2) **AG (0-hop)**：局所候補で g_0 を評価し、曖昧さが高い場合に探索を促す ($g_0 > \theta_{AG}$)。

(3) **DG (multi-hop)**：複数 hop で g_{\min} を評価し、ショートカット効果が高い場合に統合を確定する ($g_{\min} < \theta_{DG}$)。

図 1 に制御フローを示す。

2.3 g_0 と g_{\min} の算出

各ステップで $h = 0..H$ の候補グラフを構成し、次式で $g(h)$ を評価する：

$$g(h) = \Delta\text{GED}(h) - \lambda(\Delta H + \gamma\Delta\text{SP}(h)).$$

$h = 0$ は S_link (Top-L) を接続したグラフで評価し、 g_0 は曖昧さ検知の信号として用いる。 $h > 0$ では S_cand からショートカット寄与の大きい候補を貪欲に追加し、 $g_{\min} = \min_h g(h)$ を統合判定に用いる。編集コスト ΔGED は候補台で正規化し、情報利得 ΔH は S_link 基準で評価することで、尺度のブレを抑制した。

3. 実験：部分観測迷路

3.1 エピソード設計

迷路は DFS 生成 (maze_type=dfs) とし、開始位置は左上寄り (1,1)、目標は右下寄り (H-2, W-2) に固定した。各エピソードはゴール到達または最大ステップ到達で終了する。エピソード中、クエリノード $Q = (r, c, \text{query})$ をハブとするグラフを保持し、ステップごとに以下を行う：

連絡先: 宮内 和義（独立研究者）、所在地：日本、E-mail: miyauchikazuyoshi@gmail.com

表 1: 8 次元ベクトルの各成分

次元	意味
0,1	位置 ($r/H, c/W$)
2,3	方向差分 ($\Delta x, \Delta y$)
4	通行可否フラグ
5	訪問回数 ($\log(1+v)$)
6	成功フラグ
7	ゴールフラグ

エピソード処理フロー（概略）

1. クエリ Q 生成と候補集合作成
2. S.link/S.cand 構築と g_0 評価
3. multi-hop 評価で g_{\min} 算出
4. DG 成立時にエッジをコミット
5. 行動選択・移動・訪問回数更新

図 2: クエリハブ型エピソードの処理フロー（概略）

(1) **候補生成**: 現在位置から 4 方向の観測候補と、過去のメモリ由来候補を生成する。

(2) **geDIG 評価**: S.link をベースに 0-hop の g_0 を評価し、必要に応じて multi-hop で g_{\min} を評価する。

(3) **グラフ更新**: Q と方向ノードの接続、移動後の Q 間接続を追加し、訪問回数やアンカー情報を同期する。

この設計により、探索に伴う記憶更新を単一のエピソード内で一貫して扱える。方向ノードは (r, c, dir) として定義され、クエリハブ直下に接続される。また、行動後に生成される次クエリとの接続を保持することで、時系列の遷移構造を明示する。

3.2 クエリと類似度の設計

クエリ Q と候補ノードは 8 次元ベクトルで表現する。クエリベクトルは位置情報のみを持つ:

$$\mathbf{v}_q = [r/H, c/W, 0, 0, 1, 0, 0, 0].$$

候補ベクトルは位置に加え、方向差分、通行可否、訪問回数、成功/ゴール属性を持つ:

$$\mathbf{v}_c = [r/H, c/W, \Delta x, \Delta y, s, \log(1+v), \mathcal{K}_{\text{succ}}, \mathcal{K}_{\text{goal}}].$$

重み付き距離

$$d(\mathbf{v}_q, \mathbf{v}_c) = \|W \odot (\mathbf{v}_q - \mathbf{v}_c)\|_2, \quad W = (1, 1, 0, 0, 3, 2, 0, 0)$$

を用い、類似度を $\exp(-d/T)$ ($T = 0.1$) で定義する。候補集合 S.cand は $d \leq \theta_{\text{cand}}$, S.link は $d \leq \theta_{\text{link}}$ を満たすものとして構成し、上限 k_{\max} で打ち切る。メモリ候補は重み付き L2 距離による半径プリフィルタで絞り込み、計算量を制御した。

表 1 に各次元の意味を示す。

3.3 候補集合と行動選択

二段しきい値により S.cand と S.link を形成し、S.link は S.cand の部分集合に丸め込む。S.link が空の場合は、観測候補の上位を強制候補として用い、探索の停滞を避ける。行動選択は観測候補の類似度に基づき行い、通行不可の候補は除外する。この設計により、局所探索の安全性を維持しつつメモリ統合を制御できる。

3.4 実験設定

迷路サイズは 15×15 , 25×25 , 51×51 を用いた。エージェントは局所視野のみを観測し、クエリノードをハブとするグラ

表 2: 主要パラメータ

パラメータ	値
λ	1.0
γ	1.0
θ_{AG}	0.4
θ_{DG}	0.0
max_hops	10
Top-L	1
action_policy	softmax ($T = 0.1$)

表 3: 迷路サイズ別の結果（平均）

サイズ	成功率	平均ステップ	平均エッジ数
15×15	0.98	92.8	69.0
25×25	0.75	315.6	284.5
51×51	0.46	793.2	1541.0

フ (query-hub) を逐次構築する。主なパラメータは $\lambda = 1.0$, $\gamma = 1.0$, max_hops=10, $\theta_{AG} = 0.4$, $\theta_{DG} = 0.0$ とした。最大ステップは 15×15 で 250, 25×25 で 500, 51×51 で 1500 とした。シード数は 15×15 で 100, 25×25 で 60, 51×51 で 13 である。

評価指標は成功率（ゴール到達率）、平均ステップ数、平均エッジ数である。

3.5 評価指標とパラメータ

成功率はゴール到達エピソードの割合、平均ステップは到達までの行動数、平均エッジ数はエピソード終了時のグラフ規模を示す。これらに加えて、 g_0/g_{\min} の時系列を記録し、AG/DG の切り替えが探索と統合に対応しているかを確認した。表 2 に主要パラメータを示す。

3.6 結果

表 3 に各サイズの結果を示す。 15×15 と 25×25 では高い成功率を維持しつつエッジ数が迷路全域に拡張せず、必要領域に限定された構造化が見られた。 51×51 ではステップ上限の影響が大きく、探索途中で打ち切られるケースが増え成功率が低下した。

図 3 は 25×25 (seed=31) の g_0/g_{\min} の推移であり、探索フェーズと統合フェーズの切り替えが視覚的に確認できる。図 4 は同設定のグラフ成長スナップショットで、必要な領域に限定された構造化が見られる。

3.7 ログと再現性

各エピソードは summary と steps を出力し、成功率・平均ステップ・平均エッジ数に加えて、 g_0/g_{\min} 系列や候補数の統計を記録する。図 3 は steps の時系列から生成したものである。また迷路レイアウトと開始/目標位置はスナップショットとして保存し、同一条件での再実行が可能である。必要に応じて SQLite に差分レジャーを保存し、グラフ成長の再構成を行った。

4. 実装詳細

4.1 候補生成の内訳

候補は観測由来とメモリ由来に分けて生成する。観測候補は現在位置の 4 近傍から構成し、メモリ候補は過去の方ノードを参照する。メモリ候補は重み付き L2 距離に基づく半径プリフィルタで間引き、候補数の上限を制御した。

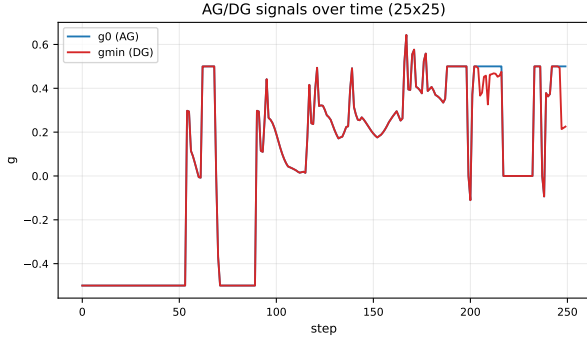


図 3: AG/DG の信号推移 (25×25, seed=31)。

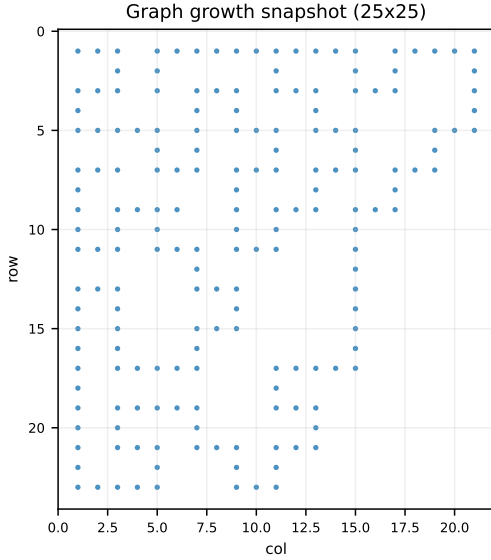


図 4: グラフ成長スナップショット (25×25)。

4.2 候補集合とコミット規則

S.link は 0-hop の評価用リンク集合であり、Top-L をベースに仮配線して g_0 を算出する。S.cand は multi-hop の探索用候補集合であり、 ΔSP 寄与が大きい枝を貪欲に追加する。DG が成立した場合のみ、候補集合から限られた本数をコミットすることで、過剰統合を抑制する。

4.3 短絡評価の計算

ΔSP はクエリ周辺の部分グラフ上で評価し、計算量を抑えるために候補対をサンプリングする。さらに最短路距離のキャッシュを用いて、エピソード内の再計算を削減した。この設計により、迷路サイズが大きい場合でも評価を継続できる。

4.4 正規化と安定化

geDIG の各項はスケールが異なるため、候補台での正規化を行い、ステップ間での比較可能性を確保する。特に ΔGED は候補集合の規模に依存しやすいため、候補数に基づく正規化を採用した。また ΔH は S.link を基準に評価し、multi-hop の評価では ΔSP の寄与に焦点を当てる。

4.5 時系列構造の保持

クエリノード間の接続を保持し、エピソードの遷移構造をグラフに明示する。これにより、探索が進むにつれてクエリ列が連結され、短絡が有効に働く局面を可視化できる。図 4 で

は、この時系列構造が必要領域に集中して成長する様子が確認できる。

5. 位置づけと応用可能性

迷路における経路計画は通常「最短経路の探索」を目的とするが、本研究の焦点は「記憶として何を統合するか」にある。そのため、A*などの経路最適化とは目的関数が異なり、短絡が有用であれば統合を促進するが、無益な統合は抑制するという設計になっている。

クエリハブ構造は、動的知識グラフの最小構成要素として解釈できる。クエリノードは「問い」、方向ノードは「候補知識」に対応し、geDIG は統合すべきエッジの選別基準として機能する。この観点から、迷路実験は知識統合の原理検証として位置づけられる。

応用先としては、検索結果を逐次統合する RAG 構成が想定される。RAG では候補文書の取り込み基準が実装依存になりがちだが、geDIG を介することで「統合のタイミング」を明示化できる可能性がある。

6. 考察

迷路は言語ノイズを排した純粋な検証環境であり、geDIG の原理検証に適している。 g_0/g_{\min} の時系列は探索 (AG) と統合 (DG) の切り替えを明確に示し、グラフスナップショットは選択的な構造化が起きていることを示す。一方で、51×51 における成功率低下はステップ上限と計算量制約の影響が大きい。AG のみ/DG のみのアブレーションや、オフライン再配線 (Phase 2) の導入は今後の課題である。

さらに、候補集合の構成が探索効率に大きく影響するため、 θ_{cand} と θ_{link} の感度解析が必要である。大規模迷路では multi-hop 評価の計算量が支配的になりやすく、探索予算と評価精度のトレードオフを明示的に扱う設計が求められる。

内発報酬は探索を促す有効な枠組みだが、報酬設計の自由度が高く、タスクごとの恣意性残りやすい。本研究の geDIG は報酬そのものではなく、記憶統合の判定基準として働くため、探索の動機付けと知識統合を分離して扱える点に利点がある。今後は報酬設計と geDIG ゲージの役割分担を明示し、適用領域ごとのスケール特性を検証する必要がある。

7. 制約と拡張

本実験では最大ステップを固定しており、大規模迷路では探索予算不足が支配的となる。特に multi-hop 評価は計算負荷が高く、より長い探索を許容するためには計算資源の拡張や評価の近似が必要である。

また、 θ_{AG} と θ_{DG} は固定値であり、環境規模や探索の進行に応じた適応が行われていない。動的しきい値 (分位更新や履歴ベースの調整) を導入することで、探索初期と収束期で異なる挙動を実現できる可能性がある。

特徴ベクトルは迷路の構造に合わせた簡易設計であり、より複雑な知識グラフに適用する際には表現能力の拡張が必要となる。その際、geDIG の役割は「評価軸の統一」に留め、特徴抽出や候補生成はタスク側に委ねる構成が望ましい。

最後に、本稿では AG のみ/DG のみのアブレーションを実施していないため、二段ゲートの寄与分解は今後の検証課題である。

8. 結論

geDIG は構造コストと情報利得のトレードオフを統合する単一ゲージとして、動的知識グラフの When 判断を支援する。部分観測迷路での実験により、探索と統合の同時制御が可能であることを示した。今後は RAG[3] など検索補助型知識統合への応用を進める。

参考文献

- [1] Friston, K.: The free-energy principle: a unified brain theory?, Nature Reviews Neuroscience, Vol.11, pp.127–138 (2010).
- [2] Grünwald, P. D.: The Minimum Description Length Principle, MIT Press (2007).
- [3] Lewis, P., et al.: Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks, NeurIPS (2020).