グラフスキーマ評価手法の提案とユーザスタディによる整合性検証

湯川　楓祐†　　塩川　浩昭††

† 筑波大学情報学群情報科学類　〒 305–8573 茨城県つくば市天王台 1-1-1
†† 筑波大学計算科学研究センター　〒 305–8573 茨城県つくば市天王台 1-1-1
E-mail: †f.yukawa@kde.cs.tsukuba.ac.jp, ††shiokawa@cs.tsukuba.ac.jp

あらまし　プロパティグラフ（PG）は属性を持つノードとエッジによりデータを表現するグラフデータモデルであり，ソーシャルネットワークをはじめとする多様な領域で利用されている．PG はスキーマレス運用によって柔軟なデータ表現を可能とする一方で，データ構造の一貫性が保証されず，クエリ性能の低下やデータ統合の複雑化を招く可能性がある．この問題に対処するため，PG に対して構造的制約を与えるスキーマを自動抽出する研究が進められているが，抽出されたスキーマの品質を定量的に評価する方法は十分に確立されていない．そこで本研究では PG のスキーマ品質を網羅性と簡潔性の観点から定量化する評価手法を提案する．提案手法の有効性を検証するため，まず複数の実データセットに対してスキーマに意図的な誤りを挿入する実験を行い，提案手法が適切に評価値を変化させることを確認した．さらにユーザスタディを通じて人間の判断と提案手法が示す評価傾向の整合性を分析し，提案手法が人間の判断と一貫した評価傾向を示すことを明らかにした．

キーワード　グラフデータベース，プロパティグラフ，NoSQL，スキーマ，ユーザスタディ

1　はじめに

プロパティグラフ（PG）は属性付きのノードとエッジからなるデータモデルであり，グラフデータベースにおいて広く利用されている．PG ではスキーマを事前に厳密に定義せずに運用できるためデータの構造の拡張に柔軟に対応できる一方，スキーマが明示されないまま拡張が進むとデータの一貫性が低下し，クエリの最適化やデータ統合が困難となる．本稿では，実際にデータが格納された PG を**インスタンス**と呼ぶ．図 1 に，ソーシャルネットワークを模した PG インスタンスの例を示す．ノードやエッジにはラベルとプロパティが付与され，ユーザや投稿，それらの関係が表現されている．

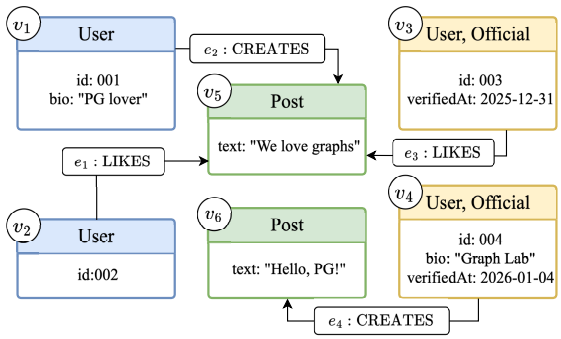


図 1: $\mathcal{G}$：SNS を模した PG の例

このような PG インスタンスに対し後付けでスキーマを導入することで，許容されるラベルやプロパティなどの制約を明示できるため，データ一貫性の検証やデータ統合の支援が期待できる．しかし，インスタンスが大規模かつ複雑な場合には人手によるスキーマ設計は困難であり，近年では PG インスタンスからスキーマを自動抽出する手法が多数提案されている [1–4]．

しかし同一の PG インスタンスに対しても，人手設計や自動抽出などにより複数の異なるスキーマ候補が得られ得る．図 2 は人手で設計されたスキーマの例であり，全体構造は直感的に理解しやすい一方，希少なプロパティ `bio` が欠落していたり，`Post` ノードの不要な区別が導入されている．一方，図 3 に示す自動抽出スキーマは属性の欠落を補完しているが，`User` や `id` が重複して定義されており，必要な定義が最小化されていない．このように PG のスキーマ品質は多面的であり，複数のスキーマ候補を客観的かつ定量的に比較することは容易ではない．

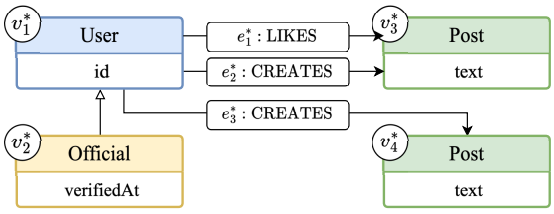


図 2: $\mathcal{S}^*$：人手で設計されたスキーマの例

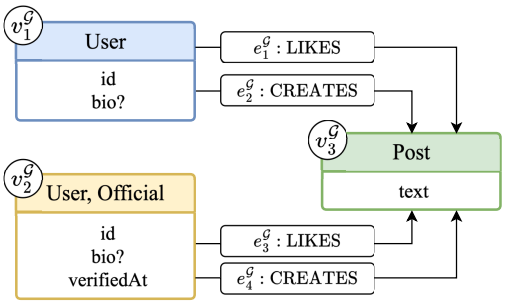


図 3: $\mathcal{S}^{\mathcal{G}}$：Xue の手法 [1] により自動抽出されたスキーマ

これまでリレーショナルデータや半構造化データを対象として，さまざまな観点からスキーマ品質を評価する研究が行われてきた [5,6]．しかし複数ラベルやプロパティの組み合わせに基づく多様な構造を許容する PG 特有の柔軟なデータモデルを考慮した定量的評価手法は十分に確立されていない．

そこで本研究では (i) インスタンスに観測される属性や制約をどの程度捉えているかを測る「網羅性」，(ii) 不要な重複や過剰な細分化を抑えた最小限の記述であるかを測る「簡潔性」，およびこれらを統合した指標である「C2 スコア」に基づく PG のスキーマ評価手法を提案する．本研究の主な貢献は以下の通りである．

1. **定量性**：網羅性と簡潔性に基づく定量的なスキーマ評価指標を定義し，それらの調和平均として統合指標 C2 スコアを導入した．
2. **適用性**：任意プロパティや継承関係などを含む複雑な PG スキーマに対しても，一貫した品質評価が可能であることを示した．
3. **スケーラビリティ**：大規模 PG インスタンスから生成されたスキーマに対しても，実用的な時間で評価可能であることを確認した．
4. **妥当性**：ユーザスタディおよび複数の実データ・人工データを用いた実験により，提案指標が人間の直感的評価と整合する傾向を持つことを示した．

2 関 連 研 究

半構造化データに対するスキーマ定義，抽出，および評価はデータ構造の理解や運用効率向上を目的として長年研究されてきた．本節ではまず既存のスキーマ定義と抽出技術を概観し，最後にスキーマ評価に関する研究動向と課題を整理する．

2.1 半構造化データのスキーマ定義

XML では DTD や XSD，JSON では JSON Schema，RDF では RDF Schema や OWL といったスキーマ言語が整備され，データ整合性の確保や相互運用性の向上を支えてきた [7–11]．

一方，PG におけるスキーマ定義は比較的新しい研究分野であり，初期には基本的なデータ構造や制約の形式化が行われ [12]，その後にはデータ検証やスキーマの拡張を扱う理論的枠組みや，一階述語論理に基づく形式的定義が提案された [13,14]．近年では GQL を見据えた PG-Schema [15] をはじめ，関数従属性や正規化手法に関する研究も進展している [16–18]．PG のスキーマの明確な定義と標準化は，クエリ高速化，データ品質の向上および異種システム間の相互運用性の確保に向けて重要な課題である [19,20]．

しかし実運用の場面ではスキーマが事前に設計されていない場合も多く，実データからの後付け設計は容易ではない．このギャップを解消するため，既存データからスキーマ情報を自動的に推定するスキーマ抽出技術が不可欠となる．

2.2 スキーマ抽出

スキーマ抽出は，明示的なスキーマを持たないデータからその構造的特徴を自動的に導出する技術である．XML や JSON におけるスキーマ抽出は，パス構造や属性分布を利用した手法からヒューリスティックおよび並列処理可能な手法へと発展してきた [5,7,21–23]．また RDF では多対多関係を持つグラフ構造を扱うためクラスタリングに基づくスキーマ抽出が主流となり [24,25]，この発想は一部の PG 向けスキーマ抽出手法にも引き継がれている．

PG においては，これまで類似度ベースクラスタリング手法 [1]，ルールベース手法 [2]，および階層型クラスタリング手法 [3,4] が提案されてきた．しかし，型階層の表現能力，プロパティ共起の扱い，スケーラビリティや冗長性といった点でそれぞれの手法ごとに限界があり，標準的アプローチは確立されていない．したがって，(1) スキーマ抽出手法を比較・分析するための基盤として (2) 抽出したスキーマが実運用に耐えうる品質であるかを事前に検証するための仕組みとしてスキーマの品質を客観的に判断できる評価の枠組みが必要である．

表 1: スキーマ評価手法の比較

評価項目	正解との比較	人間の主観評価	提案手法
正解スキーマの要否	× 必要	✓ 不要	✓ **不要**
定量性 / 再現性	✓ 高い	× 低い	✓ **高い**
実データへの適用	× 困難	✓ 可能	✓ **可能**

2.3 スキーマ評価

既存の半構造化データ分野では，正解スキーマとの一致度に基づく精度・再現率，スキーマの網羅性や冗長性，および実行性能などを用いた定量的な評価が行われてきた [5,22,23,26]．また，スキーマの可読性や妥当性に関する人間による主観的評価も重視されている [5,27]．

一方，PG 領域では主に正解スキーマとの一致度に基づく評価がスキーマ抽出手法の性能比較に用いられてきたが [1–3]，正解スキーマが存在しない実運用環境においてスキーマや抽出手法の品質・性能を評価する枠組みは十分に検討されていない．また，定量的な指標と人間の主観的評価との対応関係を体系的に検証した研究も報告されていない．

そこで本研究では，正解スキーマに依存せず，網羅性および簡潔性の観点から任意の PG スキーマの品質を定量的に評価する手法を提案する．さらに，ユーザスタディを通じて提案指標と人間の主観的評価との整合性を検証する．

表 1 に，本研究の評価手法と既存の評価手法との違いをまとめる．正解スキーマとの比較に基づく評価は 客観性や定量性に優れる一方で，正解スキーマを前提とするため実運用環境への適用が困難である．また，人間による主観評価は実データに対して柔軟に適用可能であるものの，評価の再現性や定量性に課題が残る．これに対し，本研究で提案する評価手法は正解スキーマを必要とせず，網羅性および簡潔性という定量指標に基づいて客観的かつ再現可能な評価を実現する点に特徴があり，実運用環境における PG スキーマ評価への適用が可能である．

表 2: 本稿で用いる主な記号

記号	説明
$\mathcal{G}$	PG インスタンス
$V(\cdot), E(\cdot)$	ノード集合，エッジ集合
$\lambda(o)$	オブジェクト o のラベル集合
$\kappa(o)$	オブジェクト o のプロパティキー集合
$\mu(o,k)$	オブジェクト o のプロパティ k の必須／任意
$\rho(o)$	オブジェクト o の親オブジェクト型の集合
$\mathbf{src}(e), \mathbf{dst}(e)$	エッジ e の始点ノード，終点ノード
$\mathbf{req}(o), \mathbf{opt}(o)$	オブジェクト o の必須／任意プロパティ集合
$\mathcal{S}^*$	評価対象となる元のスキーマ
$\mathcal{S}^+$	継承関係を展開したスキーマ（$= \mathbf{Flat}(\mathcal{S}^*)$）
$\mathcal{S}^{\mathcal{G}}$	$\mathcal{G}$ から抽出したスキーマ（$= \mathbf{Abs}(\mathcal{G})$）
$\mathbf{sim}_V(\cdot,\cdot), \mathbf{sim}_E(\cdot,\cdot)$	ノード型（エッジ型）間の類似度
$\mathbf{Cvg}_V, \mathbf{Cvg}_E$	ノード／エッジ網羅性
$\mathbf{Con}_V, \mathbf{Con}_E$	ノード／エッジ簡潔性
$\mathbf{C2}_V, \mathbf{C2}_E$	ノード／エッジ C2 スコア

3 事 前 準 備

3.1 用語の定義

本稿では，PG におけるノードおよびエッジを総称して**オブジェクト**と呼ぶ．また，オブジェクトに付与されるラベルおよびプロパティキーを総称して**属性**と呼ぶ．さらに，実際に観測されるノードとエッジの集合として与えられるグラフを**インスタンス**と呼び，インスタンスに現れ得るオブジェクトの構造的特徴を抽象化したものを**スキーマ**と呼ぶ．以降，特に断りのない限り，インスタンスとスキーマを区別して議論する．

3.2 プロパティグラフ

本研究では ISO/IEC 39075:2024 [28] に準拠し，GQL と互換性のある PG を対象とする．以下では，本稿の議論に必要な最小限の構成要素のみを定義する．

定義 3.2.1 (プロパティグラフ)**.** ラベル集合を $\mathcal{L}$，プロパティキー集合を $\mathcal{K}$ とする．プロパティグラフを $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \lambda, \kappa, \sigma)$ と定義する．ここで $\mathcal{V}$ はノードの有限集合，$\mathcal{E}$ は有向エッジの有限集合であり，$(\mathcal{V} \cap \mathcal{E}) = \varnothing$ とする．$\lambda : (\mathcal{V} \cup \mathcal{E}) \to 2^{\mathcal{L}}$ および $\kappa : (\mathcal{V} \cup \mathcal{E}) \to 2^{\mathcal{K}}$ はそれぞれ各オブジェクトに付与されたラベル集合およびプロパティキー集合を返す関数である．また $\sigma : \mathcal{E} \to (\mathcal{V} \times \mathcal{V})$ は各エッジの始点および終点ノードの順序付きタプルを返す関数である．

本稿ではエッジ $e \in \mathcal{E}$ に対して $\sigma(e) = (\mathbf{src}(e), \mathbf{dst}(e))$ と表記し，$\mathbf{src}(e)$ と $\mathbf{dst}(e)$ をそれぞれ**始点ノード**，**終点ノード**と呼ぶ．また，ノード集合を $V(\mathcal{G}) = \mathcal{V}$，エッジ集合を $E(\mathcal{G}) = \mathcal{E}$ と表記する．ここで，本稿で用する主な記号を表 2 に示す．

3.3 PG のスキーマ

PG のスキーマはインスタンスに現れ得るノードおよびエッジの構造的制約を抽象的に記述したものである．本研究では，実用的なプロパティグラフスキーマに必要な要素を体系的に整理した PG-Schema [15] に準拠したスキーマモデルを採用する．

本稿では，スキーマを構成する基本単位として**ノード型**および**エッジ型**を導入する．ノード型は特定のラベル集合とプロパティキー集合を持つノードの型を表し，エッジ型はこれらに加えて始点および終点となるノード型が定められたエッジの型を表す．以降，ノード型とエッジ型を総称して**オブジェクト型**，あるいは単に**型**と呼ぶ．

定義 3.3.1 (PG のスキーマ)**.** ラベル集合を $\mathcal{L}$，プロパティキー集合を $\mathcal{K}$ とする．PG のスキーマを $\mathcal{S} = (\mathcal{V}_T, \mathcal{E}_T, \lambda, \kappa, \sigma, \mu, \rho)$ と定義する．ここで $\mathcal{V}_T$ はノード型の有限集合，$\mathcal{E}_T$ はエッジ型の有限集合であり，$(\mathcal{V}_T \cap \mathcal{E}_T) = \varnothing$ とする．λ および κ は定義 3.2.1 と同様に，各オブジェクト型のラベル集合およびプロパティキー集合を返す関数である．また，$\mu : (\mathcal{V}_T \cup \mathcal{E}_T) \times \mathcal{K} \to \{\mathbf{True}, \mathbf{False}\}$ は各プロパティが必須か否かを返す関数であり，$\rho : (\mathcal{V}_T \cup \mathcal{E}_T) \to 2^{(\mathcal{V}_T \cup \mathcal{E}_T)}$ は定義 3.4.1 の継承関係に基づく親オブジェクト型を返す関数である．

本稿では，オブジェクト型 o の必須プロパティ集合および任意プロパティ集合を $\mathbf{req}(o) = \{k \in \mathcal{K} \mid \mu(o,k) = \mathbf{True}\}, \mathbf{opt}(o) = \{k \in \mathcal{K} \mid \mu(o,k) = \mathbf{False}\}$ と表記する．また任意プロパティを図示する際は，図 3 に示すようにプロパティ名の末尾に「?」を付与して表記する．

なお PG-Schema では定義 3.3.1 に加えてプロパティの型やエッジ多重度などの制約も定義されているが，本稿では議論を簡潔にするためにこれらの制約を定義から省略する．これらの制約は，4.3 節で述べる手順に軽微な拡張を施すことで本稿の評価手法に組み込むことが可能である．

3.4 オブジェクト型の継承

PG のスキーマでは複数の型が共通のラベルやプロパティを持つことが多く，これらを個別に定義するとスキーマが冗長になり保守性が低下する．この問題を避けるため，PG-Schema [15] では種々の制約を再利用する仕組みとして型の継承が導入されている．本稿でもこの考え方に基づき，オブジェクト型の継承関係を明示的に定義する．

定義 3.4.1 (オブジェクト型の継承関係)**.** オブジェクト型の集合 $\mathcal{V}_T \cup \mathcal{E}_T$ 上の二項関係 $\prec \subseteq (\mathcal{V}_T \times \mathcal{V}_T) \cup (\mathcal{E}_T \times \mathcal{E}_T)$ を **継承関係**と呼ぶ．$o_1 \prec o_2$ のとき，o_1 を **子オブジェクト型**，o_2 を **親オブジェクト型**と呼ぶ．

また継承関係 $\prec$ は非巡回であり，多重継承を許容する．加えて継承関係は推移律を満たす．すなわち，$o_1 \prec o_2$ かつ $o_2 \prec o_3$ が成り立つとき，o_1 は o_3 を継承するものと解釈する．

子オブジェクト型 o_1 が親オブジェクト型 o_2 を継承する場合，o_1 は o_2 によって課されるすべての構造的制約を満たすことを意味する．ただし例外的に，o_1 と o_2 の間で矛盾する制約が定義されている場合，o_2 の制約は o_1 に継承されない．例えば，親ノード型 o_2 があるプロパティ k を必須と定義し，子ノード型 o_1 が同じプロパティ k を任意と定義している場合，o_1 は k を任意プロパティとして扱う．

本稿では，継承関係を図示する際には図 2 および図 3 に示すように，先端が $\triangleright$ の矢印で表す．

4 提案手法

4.1 基本アイデア

本研究では，PG スキーマの品質をインスタンスに観測されるデータ構造をどの程度表現できているかという**網羅性**と，不要な冗長性を避け，人間にとって理解・保守しやすい形で記述されているかという**簡潔性**の 2 つの観点で捉える．この考え方に基づき，網羅性は「インスタンスに現れる構造がスキーマによってどの程度説明されているか」を評価する指標として定義し，簡潔性は「インスタンス構造の説明に本質的に寄与しない冗長な要素がどの程度抑制されているか」を評価する指標として定義する．最終的にこれら二つの指標を調和平均として統合することで，一方に偏ったスキーマを過大評価しない単一の品質指標である C2 スコアを導入する．

4.2 提案手法の全体像

本研究の目的は，インスタンス $\mathcal{G}$ に対するスキーマ $\mathcal{S}^*$ の品質を定量的に評価することである．そのためには，インスタンスに現れるデータ構造とスキーマが表現する構造とを同一の比較単位で対応付けた上で，網羅性と簡潔性をそれぞれ評価する必要がある．提案手法は，この比較単位の不一致を解消するための前処理と網羅性・簡潔性を算出する評価ステップから構成される．提案手法の全体像を図 4 に示す．

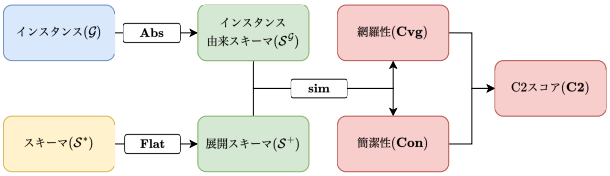


図 4: 提案手法の全体像

インスタンス $\mathcal{G}$ は個々のオブジェクトの集合として与えられる一方，スキーマはオブジェクト型の集合として与えられるため，両者の表現粒度は本質的に異なり，直接比較することができない．そこで本研究ではインスタンスに観測される要素を構造的特徴に基づいて抽象化し，**インスタンス由来スキーマ** $\mathcal{S}^{\mathcal{G}} = \mathbf{Abs}(\mathcal{G})$ を構成する．この操作により，インスタンス側をオブジェクト型の集合として扱うことが可能となる．

しかし，比較対象となるスキーマ側が継承関係を暗黙に含んだままである場合，依然として比較粒度は一致しない．PG のスキーマでは，属性や制約が親オブジェクト型に定義され子オブジェクト型には暗黙的に継承されることが多く，表層的な型定義のみを比較するとスキーマが本来表現可能な構造を過小評価してしまうおそれがある．そこで本研究では，継承によって暗黙的に与えられる属性や制約を各オブジェクト型に明示的に反映することで，インスタンス由来スキーマとの比較粒度を一致させる操作を導入する．この操作を**継承関係の展開**と呼び，この操作を施したスキーマを $\mathcal{S}^+ = \mathbf{Flat}(\mathcal{S}^*)$ として導入する．

以上の前処理により，インスタンス側を $\mathcal{S}^{\mathcal{G}}$，スキーマ側を $\mathcal{S}^+$ として両者を同一の比較単位で扱うことが可能となる．この上で網羅性および簡潔性の評価，および両者を統合した C2 スコアの算出を行う．以降では，これら各ステップについて順に形式的定義と評価方法を述べる．

4.3 インスタンス由来スキーマの生成

本研究ではインスタンス $\mathcal{G}$ に現れるノードおよびエッジをその構造的特徴に基づいて分類し，インスタンス由来スキーマ $\mathcal{S}^{\mathcal{G}}$ を構成する．この抽象化操作を $\mathcal{S}^{\mathcal{G}} = \mathbf{Abs}(\mathcal{G})$ と定義する．

$\mathbf{Abs}(\cdot)$ はノードについてはラベルが完全に一致するもの同士を，エッジについては（エッジラベル，始点ノードラベル，終点ノードラベル）のタプルが一致するもの同士をそれぞれ部分集合に分割し，各部分集合から対応するノード型およびエッジ型を生成する操作である．各オブジェクト型に付随するプロパティについては，部分集合内のすべてのオブジェクトに共通して出現するものを必須，そうでないものを任意として区別する．

なおこの操作は決定的であり，同一のインスタンス $\mathcal{G}$ に対して $\mathbf{Abs}(\mathcal{G})$ は常に一意に定まる．なお，$\mathbf{Abs}(\cdot)$ の具体的なアルゴリズムは付録 1.1 に示す．

4.4 継承関係の展開

本研究では，継承によって暗黙的に与えられる属性や制約を各オブジェクト型に明示的に反映するための操作として継承関係の展開を行う．以降，評価対象となる元のスキーマを $\mathcal{S}^*$，継承によって暗黙的に含まれる情報を各オブジェクト型に明示化したスキーマを $\mathcal{S}^+$ と表記し，この操作を $\mathcal{S}^+ = \mathbf{Flat}(\mathcal{S}^*)$ と定義する．$\mathbf{Flat}(\cdot)$ は各オブジェクト型について，そのすべての祖先型に定義された属性およびエッジの場合は端点制約を推移的にコピーし，当該オブジェクト型に明示的に付与する操作である．またエッジ型については，当該エッジ型の始点（終点）型として指定されたノード型についてそのすべての子孫型を列挙し，列挙された始点・終点型の組に対して対応するエッジ型を生成することで，暗黙的に含まれる端点制約を明示化する．

なお本研究では定義 3.4.1 に示すように継承関係が有向非巡回であることを前提としているため，$\mathbf{Flat}(\cdot)$ は同一の $\mathcal{S}^*$ に対して常に一意な $\mathcal{S}^+$ を与える．また $\mathbf{Flat}(\cdot)$ の形式的定義および具体的なアルゴリズムは付録 1.2 に示す．

4.5 オブジェクト型の類似度

オブジェクト型の類似度は (i) ラベル集合の一致度，(ii) プロパティ集合の一致度，および (iii) エッジ型の場合は端点ノード型の一致度に基づき定義する．

定義 4.5.1 (ノード型の類似度)**.** ノード型 $v^{\mathcal{G}} \in \mathcal{S}^{\mathcal{G}}$ と $v^+ \in \mathcal{S}^+$ の類似度を以下のように定義する．

$$\mathbf{sim}_V(v^{\mathcal{G}}, v^+) = \begin{cases} 0 & \lambda(v^{\mathcal{G}}) \cap \lambda(v^+) = \varnothing, \\ D_{\mathrm{attr}}(v^{\mathcal{G}}, v^+) & \text{otherwise.} \end{cases}$$

$$D_{\mathrm{attr}}(v^{\mathcal{G}}, v^+) = \alpha\, D_\lambda(v^{\mathcal{G}}, v^+) + (1-\alpha)\, D_\kappa(v^{\mathcal{G}}, v^+)$$

ただし $\alpha \in [0,1]$ はラベル一致度とプロパティ一致度の重みを調整するパラメータである．また D_λ および D_κ は Dice 係数に基づく一致度であり，以下のように定義する．

$D_\lambda(v^{\mathcal{G}}, v^+) = D(\lambda(v^{\mathcal{G}}), \lambda(v^+))$

$D_\kappa(v^{\mathcal{G}}, v^+) = \dfrac{D(\mathbf{req}(v^{\mathcal{G}}), \mathbf{req}(v^+)) + D(\mathbf{opt}(v^{\mathcal{G}}), \mathbf{opt}(v^+))}{2}$

ただし $D(A,B) = \dfrac{2|A \cap B|}{|A| + |B|}$である．

定義 4.5.1 では型同士のラベル集合が一致しない場合に類似度を 0 とすることで，意味的に明らかに異なる型が偶然に共通するプロパティキーを持つことによる誤った高類似度評価を防止している．

定義 4.5.2 (エッジ型の類似度)**.** エッジ型 $e^{\mathcal{G}} \in \mathcal{S}^{\mathcal{G}}$ と $e^+ \in \mathcal{S}^+$ の類似度を以下のように定義する．

$$\mathbf{sim}_E(e^{\mathcal{G}}, e^+) = \begin{cases} 0, & \lambda(e^{\mathcal{G}}) \cap \lambda(e^+) = \varnothing, \\ D_{\mathrm{edge}}(e^{\mathcal{G}}, e^+), & \text{otherwise.} \end{cases}$$

$$D_{\mathrm{edge}}(e^{\mathcal{G}}, e^+) = \beta\, D_{\mathrm{attr}}(e^{\mathcal{G}}, e^+) + (1-\beta)\, D_\sigma(e^{\mathcal{G}}, e^+)$$

ただし $\beta \in [0,1]$ はエッジ型の属性の一致度と端点ノード型の一致度の重みを調整するパラメータである．また D_σ は端点ノード型の一致度であり，以下のように定義する．

$$D_\sigma(e^{\mathcal{G}}, e^+) = \frac{\mathbf{sim}_V(\mathbf{src}(e^{\mathcal{G}}), \mathbf{src}(e^+)) + \mathbf{sim}_V(\mathbf{dst}(e^{\mathcal{G}}), \mathbf{dst}(e^+))}{2}$$

4.6 網羅性（Coverage）

網羅性は，インスタンスに観測される各オブジェクト型に対しスキーマ中で最も類似する型との一致度を用いることで，スキーマがインスタンス構造をどの程度説明できているかを評価する指標である．インスタンス由来の型を十分に表現できる型がスキーマに存在する場合には高い類似度が得られ，対応する型がスキーマに存在しない場合には類似度は低い値となる．このような対応関係に基づく類似度を集約することで，スキーマがインスタンス構造をどの程度網羅しているかを定量化する．

定義 4.6.1 (網羅性)**.** 評価対象のスキーマを $\mathcal{S}^*$，インスタンスを $\mathcal{G}$ とする．ノード網羅性およびエッジ網羅性を

$$\mathbf{Cvg}_V(\mathcal{G}, \mathcal{S}^*) = \frac{1}{|V(\mathcal{S}^{\mathcal{G}})|} \sum_{v^{\mathcal{G}} \in V(\mathcal{S}^{\mathcal{G}})} \max_{v^+ \in V(\mathcal{S}^+)} \mathbf{sim}_V(v^{\mathcal{G}}, v^+)$$

$$\mathbf{Cvg}_E(\mathcal{G}, \mathcal{S}^*) = \frac{1}{|E(\mathcal{S}^{\mathcal{G}})|} \sum_{e^{\mathcal{G}} \in E(\mathcal{S}^{\mathcal{G}})} \max_{e^+ \in E(\mathcal{S}^+)} \mathbf{sim}_E(e^{\mathcal{G}}, e^+)$$

と定義する．ただし，$\mathcal{S}^{\mathcal{G}} = \mathbf{Abs}(\mathcal{G})$，$\mathcal{S}^+ = \mathbf{Flat}(\mathcal{S}^*)$ である．

4.7 簡潔性（Concision）

簡潔性は，インスタンス構造の説明に本質的に寄与しない冗長なオブジェクト型がスキーマにどの程度含まれているかを測る指標である．簡潔性の計算では，各スキーマオブジェクト型を 1 つずつ除去し，その除去が網羅性に与える影響に基づいて冗長性を判定する．除去しても網羅性の低下が十分に小さい場合，当該オブジェクト型はインスタンス構造の説明に本質的には寄与しておらず冗長であるとみなす．このような冗長オブジェクト型の割合に基づいて簡潔性を定義する．

まず，「網羅性の低下が十分に小さい」ことを定量化するため寄与下限を定義する．

定義 4.7.1 (寄与下限)**.** 評価対象のスキーマを $\mathcal{S}^*$，インスタンスを $\mathcal{G}$ とし，$\mathcal{S}^+ = \mathbf{Flat}(\mathcal{S}^*)$ とする．ノード網羅性およびエッジ網羅性に対する寄与下限を

$$\epsilon_V = \frac{\mathbf{Cvg}_V(\mathcal{G}, \mathcal{S}^*)}{|V(\mathcal{S}^+)|} \cdot \gamma,$$

$$\epsilon_E = \frac{\mathbf{Cvg}_E(\mathcal{G}, \mathcal{S}^*)}{|E(\mathcal{S}^+)|} \cdot \gamma$$

と定義する．ただし $\gamma \in (0,1]$ は冗長性判定の厳しさを調整するパラメータである．

次に，定義 4.7.1 で定義した寄与下限を用いて冗長オブジェクト型を定義する．

定義 4.7.2 (冗長オブジェクト型)**.** オブジェクト型 o を評価対象スキーマ $\mathcal{S}^*$ から除去したときのノード網羅性およびエッジ網羅性の低下量を以下のように定義する．

$$\Delta_V(o) = \mathbf{Cvg}_V(\mathcal{G}, \mathcal{S}^*) - \mathbf{Cvg}_V(\mathcal{G}, \mathcal{S}^* \setminus \{o\})$$

$$\Delta_E(o) = \mathbf{Cvg}_E(\mathcal{G}, \mathcal{S}^*) - \mathbf{Cvg}_E(\mathcal{G}, \mathcal{S}^* \setminus \{o\})$$

このとき，$\Delta_V(o) < \epsilon_V$ かつ $\Delta_E(o) < \epsilon_E$ を満たすオブジェクト型 o を**冗長オブジェクト型**と定義する．さらに冗長オブジェクト型の集合を以下のように定義する．

$$R_V(\mathcal{G}, \mathcal{S}^*) = \{\, v^* \in V(\mathcal{S}^*) \mid \Delta_V(v^*) < \epsilon_V \wedge \Delta_E(v^*) < \epsilon_E \,\}$$

$$R_E(\mathcal{G}, \mathcal{S}^*) = \{\, e^* \in E(\mathcal{S}^*) \mid \Delta_V(e^*) < \epsilon_V \wedge \Delta_E(e^*) < \epsilon_E \,\}$$

ここで，オブジェクト型の除去操作は o がノード型である場合には当該ノード型とそれを端点として参照するエッジ型を同時に除去し，o がエッジ型である場合には当該エッジ型のみを除去する．このため，ノード型を除去した場合には対応するエッジ型も失われることでエッジ網羅性が低下し得る．一方，エッジ型を除去した場合でもそれが継承関係を表すエッジ型であるときには属性や制約の継承が行われなくなるため，展開後スキーマにおけるノード型の属性集合が変化し，ノード網羅性が低下し得る．したがって，冗長オブジェクト型の判定ではノード型・エッジ型の別に関わらず Δ_V と Δ_E の両方を考慮する．

最後に，冗長オブジェクト型の割合に基づいて簡潔性を定義する．

定義 4.7.3 (簡潔性)**.** 評価対象のスキーマを $\mathcal{S}^*$，インスタンスを $\mathcal{G}$ とする．ノード簡潔性およびエッジ簡潔性を以下のように定義する．

$$\mathbf{Con}_V(\mathcal{G}, \mathcal{S}^*) = 1 - \frac{|R_V(\mathcal{G}, \mathcal{S}^*)|}{|V(\mathcal{S}^*)|}$$

$$\mathbf{Con}_E(\mathcal{G}, \mathcal{S}^*) = 1 - \frac{|R_E(\mathcal{G}, \mathcal{S}^*)|}{|E(\mathcal{S}^*)|}$$

4.8 C2 スコア

C2 スコアは網羅性と簡潔性を単一の指標として統合するための指標であり，いずれか一方が低い場合にスコアが低下するように両者の調和平均として定義する．

定義 4.8.1 (C2 スコア). 評価対象のスキーマを $\mathcal{S}^*$，インスタンスを $\mathcal{G}$ とする．ノード C2 スコアおよびエッジ C2 スコアを以下のように定義する．

$$\mathbf{C2}_V(\mathcal{G},\mathcal{S}^*) = \frac{2\cdot \mathbf{Cvg}_V(\mathcal{G},\mathcal{S}^*)\cdot \mathbf{Con}_V(\mathcal{G},\mathcal{S}^*)}{\mathbf{Cvg}_V(\mathcal{G},\mathcal{S}^*)+\mathbf{Con}_V(\mathcal{G},\mathcal{S}^*)},$$

$$\mathbf{C2}_E(\mathcal{G},\mathcal{S}^*) = \frac{2\cdot \mathbf{Cvg}_E(\mathcal{G},\mathcal{S}^*)\cdot \mathbf{Con}_E(\mathcal{G},\mathcal{S}^*)}{\mathbf{Cvg}_E(\mathcal{G},\mathcal{S}^*)+\mathbf{Con}_E(\mathcal{G},\mathcal{S}^*)}$$

計算量 提案手法全体の計算量は $O\big(|V(\mathcal{G})|+|E(\mathcal{G})|+|V(\mathcal{S}^{\mathcal{G}})||V(\mathcal{S}^+)|+|E(\mathcal{S}^{\mathcal{G}})||E(\mathcal{S}^+)|\big)$ である．アルゴリズムの擬似コードと簡単な計算量解析は付録に示す．

4.9 提案指標の計算例

図 1 に示す PG インスタンスを $\mathcal{G}$，図 2 に示すスキーマを評価対象スキーマ $\mathcal{S}^*$ とし，提案指標の計算例を示す．以下ではノードの網羅性・簡潔性・C2 スコアについての計算過程を示す．パラメータは $\alpha=0.5$，$\beta=0.5$，$\gamma=0.15$ とする．

(1) インスタンス由来スキーマの生成 インスタンス $\mathcal{G}$ に観測される構造を抽象化し，$\mathcal{S}^{\mathcal{G}}=\mathbf{Abs}(\mathcal{G})$ を構成する．得られたインスタンス由来スキーマを図 3 に示す [1]．

本例では，インスタンス $\mathcal{G}$ のノードはラベル集合に基づいて $\{v_1,v_2\}$，$\{v_3,v_4\}$，$\{v_5,v_6\}$ の 3 つの部分集合に分割され，それぞれからノード型 $v_1^{\mathcal{G}},v_2^{\mathcal{G}},v_3^{\mathcal{G}}$ が生成される．エッジについても同様に（始点ラベル集合，エッジラベル，終点ラベル集合）のタプルに基づいて分割され，4 種類のエッジ型が生成される．

(2) 継承関係の展開 評価対象スキーマ $\mathcal{S}^*$ に対して継承関係を展開し，$\mathcal{S}^+=\mathbf{Flat}(\mathcal{S}^*)$ を構成する．$\mathcal{S}^*$ を展開したスキーマを図 5 に示す．この操作により，親型から暗黙的に継承されていたラベルやプロパティおよび端点制約が明示化される．

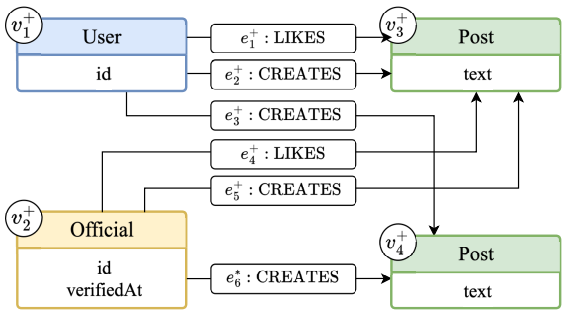


図 5: 展開後スキーマ $\mathcal{S}^+$

(3) 類似度の計算 定義 4.5.1 に基づき，(1) のインスタンス由来スキーマ $\mathcal{S}^{\mathcal{G}}$ の各ノード型と (2) の展開後スキーマ $\mathcal{S}^+$ の各ノード型との間で類似度を計算する．表 3 にノード型の類似度行列を示す．

(4) ノード網羅性の計算 定義 4.6.1 および表 3 に基づき，ノード網羅性は以下のように計算される．

1：1 節では図 3 は Xue らの手法によって抽出されたスキーマとして紹介しているが，本節ではインスタンス $\mathcal{G}$ に対して **Abs** を適用して得られるインスタンス由来スキーマとして用いている．本稿では両者が一致しているため同一の図を用いているが，一般には両者が一致するとは限らない．

表 3: ノード型の類似度行列

	v_1^+	v_2^+	v_3^+	v_4^+
$v_1^{\mathcal{G}}$	**0.75**	0.5	0.0	0.0
$v_2^{\mathcal{G}}$	0.5	**0.75**	0.0	0.0
$v_3^{\mathcal{G}}$	0.0	0.0	**1.0**	**1.0**

表 4: ノード型の冗長性評価

除去したノード型	Δ_V	Δ_E	$\Delta_V<\epsilon_V$	$\Delta_E<\epsilon_E$	冗長ノード型
v_1^*	0.33	0.93	False	False	False
v_2^*	0.08	0.03	False	False	False
v_3^*	0.00	0.47	True	False	False
v_4^*	0.00	0.00	True	True	True

$$\mathbf{Cvg}_V(\mathcal{G},\mathcal{S}^*)=\frac{1}{3}(0.75+0.75+1.0)=\boxed{0.83}$$

(5) ノード簡潔性の計算 まず，定義 4.7.1 に基づき寄与下限を計算するとそれぞれ $\epsilon_V=0.031$，$\epsilon_E=0.023$ となる．次に，各ノード型を $\mathcal{S}^*$ から除去した場合の網羅性低下量 Δ_V,Δ_E を計算し，冗長性を判定する．表 4 に各ノード型の判定結果を示す．表より，ノード型 v_4^* のみがノード網羅性およびエッジ網羅性のいずれに対しても寄与が小さく，冗長な型と判定される．したがって定義 4.7.3 に基づき，ノード簡潔性は以下のように計算される．

$$\mathbf{Con}_V(\mathcal{G},\mathcal{S}^*)=1-\frac{1}{4}=\boxed{0.75}$$

(5) ノード C2 スコアの計算 定義 4.8.1 に基づき，ノード C2 スコアは以下のように計算される．

$$\mathbf{C2}_V(\mathcal{G},\mathcal{S}^*)=\frac{2\cdot 0.83\cdot 0.75}{0.83+0.75}=\boxed{0.79}$$

5 評価実験

5.1 評価実験の概要

本研究では，提案指標がプロパティグラフのスキーマ品質評価指標として実運用に耐えうるかを検証するため，以下に示す 4 つのリサーチクエスチョン（RQ）を設定し，それぞれに対応する評価実験を行う．

RQ1：大規模グラフに対しても実行可能であるか 大規模なプロパティグラフに対しても，提案指標が現実的な計算時間で実行可能であるかを検証する．複数の実データセットを用い，ノード数およびエッジ数の増加に伴う評価時間のスケーリング挙動を測定する．

RQ2：スキーマに含まれる誤りに応じて評価値が変化するか スキーマに欠損や冗長を段階的に導入した場合に，誤りの種類および量に応じて評価値が適切に変化するかを検証する．

RQ3：異なるスキーマ間の構造的差異を区別できるか 同一データセットに対して複数の既存スキーマ抽出手法を適用し，生成されたスキーマ間の構造的差異が提案指標によって区別可能かを検証する．

表 5: データセット一覧

データセット	カテゴリ	ノード数	エッジ数	ノードラベル数	エッジラベル数	ノードプロパティ数	エッジプロパティ数	正解スキーマ	継承	任意プロパティ	複数ラベル	実データ／人工データ
FinDKG [29]	知識グラフ	13,645	223,983	12	15	2	1	–	–	–	–	実データ
LDBC-SNB [30]	ソーシャルグラフ	30,191	44,742	14	15	16	3	✓	✓	✓	✓	人工データ
NeuPrint MB6 [31]	生物学	486,267	961,571	5	3	32	3	✓	✓	✓	✓	実データ
IT-Management [32]	IT	83,847	181,995	17	12	17	0	–	✓	✓	✓	人工データ
Northwind [33]	物流	1,035	3,139	5	4	38	5	✓	–	–	–	人工データ
Spotify [34]	音楽	64,458	155,430	4	4	16	0	✓	–	–	–	実データ
Steam(this work)	ゲーム	229,372	438,264	28	9	20	4	✓	✓	✓	✓	実データ
TPC-H [35]	物流	78,805	205,150	7	7	30	3	✓	–	–	–	人工データ
Twitter [36]	ソーシャルグラフ	43,325	56,403	6	8	14	0	–	✓	✓	–	実データ
WordNet [37]	知識グラフ	689,929	1,013,075	12	31	35	0	–	✓	✓	✓	実データ

RQ4：人間による品質判断と評価結果が整合するか ユーザスタディを通じて，人間によるスキーマ品質の主観的な順位付けと提案指標による評価結果との整合性を検証する．

実験環境 本実験は，Apple M1 Ultra（20 コア）および 128 GB メモリを搭載した Mac Studio 上で実行した．OS には macOS 15.5 を使用し，グラフデータベースは Neo4j Enterprise 2025.11.2 を用いた．アルゴリズムは Python 3.12.4 で実装し，すべての実験アルゴリズムはシングルスレッドで実行した．

データセット 本実験では，表 5 に示す 10 種類のデータセットを用いた．本実験では実世界データと人工データの双方を使用し，また継承関係，任意プロパティ，マルチラベルといった PG 特有の構造を含むデータセットとそれらを含まないデータセットの双方を用いることで，データ構造の多様性を広くカバーしている．なお，Steam データセットは本研究のために著者らが独自に収集・構築したものである．

5.2 RQ1：大規模グラフに対する実行可能性

概要 本実験では，入力グラフの規模を段階的に拡大した場合の実行時間を測定し，提案指標が大規模なプロパティグラフに対しても実用的な時間で評価を実行可能であるかを検証する．提案手法の計算量は $O\big(|V(\mathcal{G})|+|E(\mathcal{G})|+|V(\mathcal{S}^{\mathcal{G}})||V(\mathcal{S}^{+})|+|E(\mathcal{S}^{\mathcal{G}})||E(\mathcal{S}^{+})|\big)$ である．一般にスキーマサイズはインスタンスサイズに比べて十分小さいため，実行時間は主として $|V(\mathcal{G})|$ および $|E(\mathcal{G})|$ の走査コストに支配され，入力規模に対して概ね線形に増加すると期待される．

データセットは，スキーマ構造を固定したままインスタンスの規模のみを制御可能な LDBC-SNB データセットを用いる．インスタンス規模を調整する変数 scale factor を段階的に変更して異なる規模のデータセットを生成し，各データセットに対して Lbath らのスキーマ抽出手法 [2] により得られたスキーマを入力として提案手法によるスキーマ評価を実行した．実行時間にはデータ読み込みおよびスキーマ評価計算に要する時間を含み，スキーマ抽出に要する時間は含まない．各条件について 10 回実行し平均実行時間を記録した．

実験結果 表 6 に各 scale factor におけるノード数，エッジ数，および平均実行時間を示す．表より，提案手法は入力規模に対してほぼ線形にスケールし，数千万〜2 億エッジ規模のグラフに対してもスキーマ品質評価を現実的な計算時間で実行可能であることが示された．またこの結果は理論的な計算量解析とも整合している．

表 6: スキーマ評価の実行時間

Scale Factor	ノード数	エッジ数	実行時間 [秒]
0.1	0.4M	2M	9
0.3	1M	6M	15
1.0	4M	21M	29
3.0	11M	63M	118
10.0	34M	200M	354

5.3 RQ2：スキーマ誤りに対する評価値の変化

概要 本実験では，提案指標がスキーマに含まれる誤りの種類および程度に応じた直感的に妥当な評価値の変化を示すかを検証する．本実験では正解スキーマに対して人為的に誤りを挿入し，誤り数の増加に伴う評価値の挙動を観測する．

NeuPrint MB6 および Steam データセットに付属する正解スキーマを起点とし，(a) オブジェクト型の削除，(b) ラベルの削除，(c) プロパティの削除，(d) オブジェクト型の重複追加，(e) エッジ方向の反転，(f) (a) – (e) からのランダム選択の計 6 種類の誤り挿入操作を行う．ノード型に対しては (a) – (d) および (f)，エッジ型に対しては (a) および (c) – (f) を適用した．なお Neo4j はラベルなしエッジを許容しないため，(b) はエッジに対して適用していない．誤り数は 0 から 5 までの 6 段階とし，各（誤り種別，誤り数）の組に対して 100 個のスキーマを生成・評価し，評価値の平均を算出した．

実験結果 図 6〜図 9 に NeuPrint MB6 および Steam データセットにおける誤り数および誤り種別ごとのスキーマ評価結果を示す．図より，いずれのデータセットにおいても正解スキーマ（誤り数 0）が最も高いスコアを示し，誤り数の増加に伴ってスコアが単調に低下する傾向があることが分かる．したがって，提案指標はスキーマの誤り数に応じて直感的に妥当な形で評価値を低下させることが確認された．

また誤りの種類によってスコア低下の度合いが異なることも確認できる．例えばノードプロパティの欠損に対しては比較的緩やかなスコアの低下が見られる一方，ノードラベルの欠損に対しては急激なスコア低下が生じている．一般的にラベルの欠損はプロパティの欠損に比べてオブジェクト型の識別力を大きく損なうため，提案手法はスキーマに含まれる誤りの性質に応

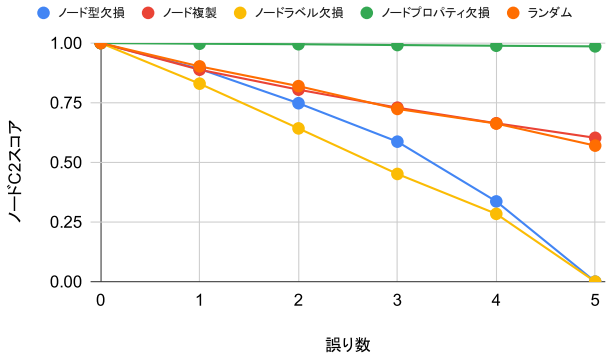


図 6: ノード C2 スコア (MB6)

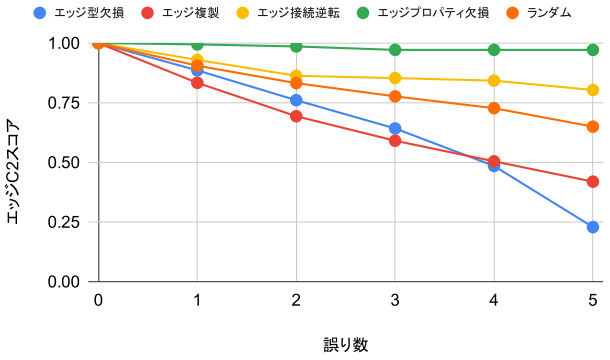


図 7: エッジ C2 スコア (MB6)

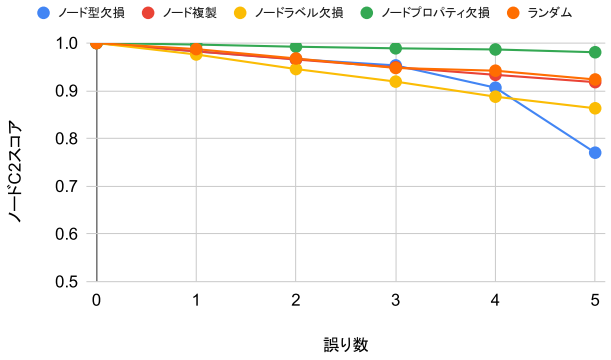


図 8: ノード C2 スコア (Steam)

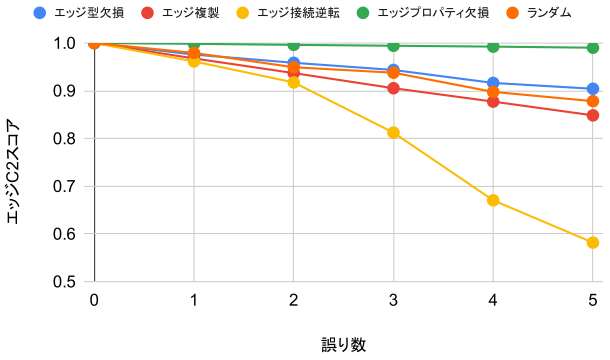


図 9: エッジ C2 スコア (Steam)

じて適切に評価値を変化させていることが示唆される．

さらにオブジェクト型やラベルの欠損では主として網羅性が低下するのに対し，冗長なオブジェクト型の追加では簡潔性が大きく低下する傾向が観測された．これらの結果は，提案指標が欠陥の種類に応じて網羅性と簡潔性を適切に区別して評価できていることを示している．

以上より，提案手法は正しいスキーマに対して高い評価を与え，誤ったスキーマに対しては誤り数および誤りの性質に応じて直感的に妥当な形で各評価値を低下させることが確認された．

5.4 スキーマ抽出手法間比較実験

概要 本実験では異なるスキーマ抽出手法によって得られたスキーマ間の評価値の差が，スキーマの構造的特徴として人間にとって説明可能な形で現れるかを検証する．なお本実験は抽出手法の優劣を決定することを目的とするものではない．

本実験では表 5 に示す全てのデータセットに対して類似度ベース手法 [1]（以下，手法 1），ルールベース手法 [2]（手法 2），および階層型クラスタリング手法 [3]（手法 3）の 3 種類のスキーマ抽出手法を適用し，得られたスキーマを評価した．

実験結果 各データセットおよび各手法で抽出したスキーマの評価結果を表 7 に示す．表より，手法 3 による抽出スキーマは他の手法と比較して全体的に網羅性がやや低い傾向が確認できる．これは，手法 3 がインスタンス全体を走査せずサンプリングに基づいてスキーマを抽出するため，希少なラベルやプロパティがスキーマに反映されにくいことに起因すると考えられる．例えば WordNet データセットでは，手法 1 および手法 2 が 12 種類のノードラベルを抽出しているのに対し手法 3 は 10 種類にとどまっており，このことがノード網羅性の低下の一因として評価値に反映されていた．

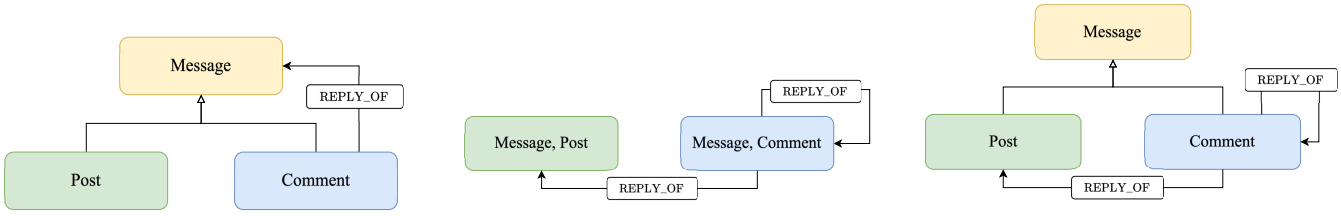
一方，手法 1 および手法 2 による抽出スキーマは高い網羅性とノード簡潔性を示す一方で，エッジ簡潔性が相対的に低くなる傾向が観測された．この傾向も各手法の設計上の特性と整合している．例えば手法 2 では継承関係を抽出するものの，親オブジェクト型へのエッジの集約を行わないため意味的に同一のエッジが子ノード型の組ごとに分散して定義されやすい．また手法 1 は継承関係を扱わないため親型による関係の集約が行われず，結果としてエッジ型の冗長性が生じやすい．図 10 は LDBC-SNB データセットにおける正解スキーマと抽出スキーマの一部を示したものである（ただしプロパティは省略している）．正解スキーマでは `Message` に対して集約的に定義されている `REPLY_OF` エッジが，手法 1 および手法 2 による抽出スキーマでは `Post` や `Comment` ごとに分散して定義されており，これがエッジ簡潔性の低下として評価値に反映されていることが分かる．以上より，提案指標はスキーマ抽出手法の設計上の特性やそれに起因するスキーマ構造の違いを評価値として反映できることが確認された．

5.5 ユーザスタディ

概要 本ユーザスタディの目的は (1) 人間がスキーマ品質を評価する際に重視する欠陥の種類とその相対的重要度を明らかにす

表 7: 網羅性，簡潔性，C2 スコアの評価結果

データセット	手法 1						手法 2						手法 3					
	網羅性		簡潔性		C2 スコア		網羅性		簡潔性		C2 スコア		網羅性		簡潔性		C2 スコア	
	V	E	V	E	V	E	V	E	V	E	V	E	V	E	V	E	V	E
FindKG	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.88	1.00	1.00	1.00	0.93
LDBC-SNB	1.00	1.00	1.00	0.44	1.00	0.61	1.00	1.00	1.00	0.56	1.00	0.72	0.98	0.96	0.77	0.45	0.86	0.61
NeuPrint MB6	1.00	0.88	1.00	0.33	1.00	0.48	0.98	0.99	1.00	0.40	0.99	0.57	0.78	0.93	0.50	0.35	0.61	0.51
IT-Management	1.00	1.00	1.00	0.41	1.00	0.58	1.00	1.00	0.96	0.48	0.98	0.65	0.94	0.89	0.76	0.36	0.84	0.52
Northwind	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.97	1.00	1.00	1.00	0.98
Spotify	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Steam	1.00	1.00	1.00	0.04	1.00	0.08	1.00	1.00	0.97	0.19	0.98	0.31	0.99	0.93	0.77	0.02	0.87	0.03
TPC-H	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.98	1.00	1.00	1.00	0.99
Twitter	1.00	1.00	1.00	0.64	1.00	0.78	0.94	0.96	1.00	0.42	0.97	0.58	0.93	0.80	0.42	0.23	0.58	0.35
WordNet	0.93	0.88	1.00	0.31	0.97	0.46	0.86	0.93	0.91	0.34	0.88	0.50	0.71	0.84	0.58	0.09	0.64	0.16



(a) 正解スキーマの一部　(b) 手法 1 による抽出スキーマの一部　(c) 手法 2 による抽出スキーマの一部

図 10: LDBC データセットにおける正解スキーマと抽出スキーマの比較（プロパティは省略）

ること，および (2) 提案手法による評価結果が人間の集合的判断とどの程度一致するかを定量的に検証することである．

ユーザスタディでは，12 種類の PG インスタンスそれぞれに対し正解スキーマ 1 件と，意図的に欠陥を含めた誤りスキーマ 3 件の計 4 件を用意した．誤りスキーマは，(i) 型・ラベル・プロパティの欠損（**構造の欠損**），(ii) エッジ接続誤りや継承関係の逆転（**意味論的破綻**），(iii) 継承構造の過剰展開や型複製（**冗長な構造**），(iv) インスタンスに存在しない要素の導入（**不要な構造**）のいずれか，または複数を含むよう設計した．

評価者は情報系分野の大学生・大学院生・教員からなる 30 名である．各設問について，明確な判断基準を与えず 4 件のスキーマを直感的に良いと思う順に 1 位から 4 位まで順位付けさせた．なお同順位は認めなかった．

ゴールデンランキング 各設問の順位結果は Borda Count により集約し，ゴールデンランキングを構築した．さらに，評価者間の順位分布が一様分布から有意に逸脱しているかをカイ二乗検定により検証した結果，すべての設問で $p < 0.001$ が確認され，統計的に有意な合意が存在することを確認した．

5.5.1 人間のスキーマ評価軸の分析

本実験では明示的な評価基準を与えずに得られた順位結果から，人間がどの欠陥を相対的に重大と捉えるかを分析する．各誤りスキーマには，意味論的破綻，構造の欠損，不要な構造，冗長な構造のいずれか 1 つの代表的欠陥タグを付与した．なおスキーマが誤りを複数含む場合は，意味論的破綻 >構造の欠損 >不要な構造 >冗長な構造 の優先順でタグを付与した．

同一設問内の誤りスキーマ同士の順位関係に基づき，ある評価者が欠陥タグ t を持つスキーマを他の欠陥より重大と判断した確率 $p(t > \textbf{others})$ を算出し，全評価者・全設問について集約した．結果を表 8 および図 11 に示す．

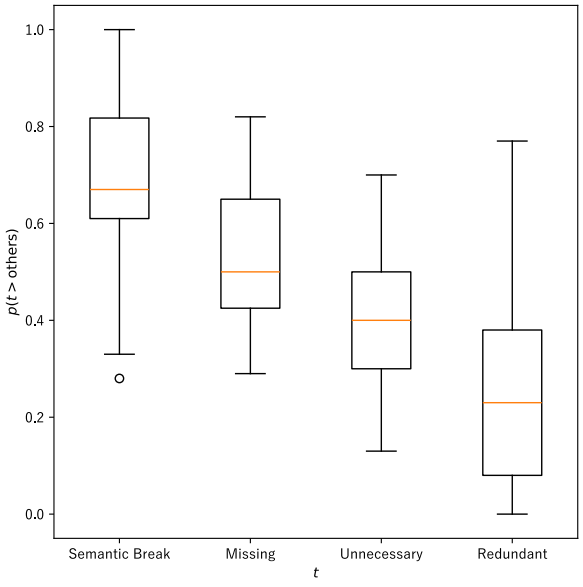


図 11: 欠陥タグごとの $p(t > \textbf{others})$ の分布

表 8: 欠陥タイプごとの相対的重大度の比較

Name	p(意味破綻 > その他)	p(欠損 > その他)	p(不要 > その他)	p(冗長 > その他)
評価者平均	0.68	0.54	0.42	0.25
ゴールデンランキング	0.83	0.59	0.40	0.00
提案手法	0.83	0.06	0.30	0.77

表 8 より，人間のスキーマ評価では，ゴールデンランキングおよび評価者平均の双方において意味論的破綻が最も重大と評価された．次いで構造の欠損，不要な構造，冗長な構造の順に重大と評価される一貫した傾向が確認された．特に意味論的破綻は他の欠陥タイプと比較して著しく高い割合で重大と判断されており，スキーマの正当性を評価する上で最も支配的な評価軸であることが示唆される．

一方で図 11 に示すように，いずれの欠陥タイプについても評価者間には一定のばらつきが存在することが分かる．その中でも冗長な構造はゴールデンランキングでは重大と判断されない一方で，評価者間の判断の分散が最も大きい欠陥タイプであることが確認された．この結果は，特定の欠陥タイプに限らずスキーマ品質の評価という行為そのものが評価者ごとに異なる判断基準を含み，一定の属人性を内在していることを示唆している．

このような属人性はスキーマの比較や選択を行う際の再現性を低下させる要因となる．したがって，人間の主観的判断に依存せず一貫した基準でスキーマを比較可能とする定量的かつ自動的な評価指標を提供することの重要性が，本ユーザスタディの結果からも裏付けられたと言える．

5.5.2 提案手法とゴールデンランキングの比較

本実験では提案手法によるスキーマ順位とゴールデンランキングとの一致度を評価する．提案手法による順位は，各スキーマ案に対して算出したノードおよびエッジの C2 スコアの平均に基づき決定した．

まず Kendall の順位相関係数 τ を用いて各設問における提案手法の順位とゴールデンランキングとの一致度を測定した．その結果，12 設問中 11 設問で $\tau > 0$，1 設問で $\tau = 0$ が得られ，平均 $\tau = 0.523$（95% 信頼区間は $[0.39, 0.67]$）であった．また，最上位スキーマの一致率は 100% であった．これらの結果は，提案手法が人間の集合的判断と同方向の順位付けを一貫して再現していることを示している．

また表 8 を参照して欠陥タイプ別に評価傾向を比較すると，提案手法と人間評価はいずれも意味論的破綻を最も重大な欠陥として評価する点で一致していた．一方で，構造の欠損と冗長な構造に対する評価の重み付けには明確な差が見られた．具体的には，人間評価では構造の欠損に対してより大きなペナルティが与えられる傾向があるのに対し，提案手法では冗長な構造が相対的に重大と評価される傾向が確認された．

この評価傾向の差は，冗長性判定の強弱を決定するハイパパラメータ γ の値を小さくしたり，C2 スコアを算出する際に網羅性と簡潔性の重み付けを調整することである程度制御可能である．一方で，前節で示した通り人間のスキーマ評価自体にも評価者間のばらつきが存在する．したがって，各欠陥をどの程度重く扱うべきかという点については今後さらなる検討が必要な課題である．

6 まとめと今後の課題

本研究では，スキーマレスに運用される PG に対して複数のスキーマ候補を客観的かつ定量的に比較するためのスキーマ品質評価手法を提案した．具体的には，インスタンスに観測される構造をどの程度捉えているかを測る**網羅性**と不要な冗長性や過剰な細分化を抑えた記述であるかを測る**簡潔性**を定義し，それらを統合した指標として **C2 スコア**を導入した．

また評価実験では提案指標が大規模な PG インスタンスに対しても実用的な計算時間で算出可能であることを確認した．加えて，提案手法がスキーマに含まれる欠損や冗長といった誤りの種類および程度に応じて評価値を直感的に妥当な形で変化させることや，既存のスキーマ抽出手法によって生成されたスキーマ間の構造的差異を説明可能な形で評価値に反映できることを示した．

さらに，ユーザスタディを通じて人間のスキーマ評価における主要な評価軸とその相対的重要度を明らかにし，提案指標による評価結果が人間による集合的なスキーマ品質判断と整合する傾向を持つことを確認した．一方で，人間のスキーマ評価には欠陥タイプによらず評価者間のばらつきが存在し，スキーマ評価という行為自体が一定の属人性を内在することも明らかとなった．この結果は，スキーマの比較や選択を人間の主観的判断のみに依存することの限界を示すものであり，再現可能で一貫した基準に基づく自動的なスキーマ評価手法の必要性を裏付けるものである．

今後の課題として，本研究では評価対象から除外したプロパティの値域や型，エッジの多重度などを評価指標に組み込むことが挙げられる．また，データ特性や利用目的に応じて自動的に各種ハイパーパラメータを調整する手法の検討も重要な課題である．これらの課題に取り組むことで，多様な運用要件やデータ特性を持つ PG に対しても，一貫した基準に基づくスキーマ品質評価を可能とする汎用的な評価基盤を確立できると考える．

謝　　辞

本研究の一部は JST 創発的研究支援事業 JPMJFR232P，ならびに，JSPS 科研費 若手研究 22K17894 の支援を受けたものである．

文　献

[1] Xue Lei. Property graph schema extraction. Master's thesis, Eindhoven University of Technology, 2021.

[2] Hanâ Lbath, Angela Bonifati, and Russ Harmer. Schema inference for property graphs. In Proceedings of the 24th International Conference on Extending Database Technology (EDBT 2021), pp. 499–504, 2021.

[3] Angela Bonifati, Stefania Dumbrava, and Nicolas Mir. Hierarchical clustering for property graph schema discovery. In 25th Proceedings of the International Conference on Extending Database Technology (EDBT 2022), pp. 449–453, 2022.

[4] Angela Bonifati, Stefania Dumbrava, Emile Martinez, Fatemeh Ghasemi, Malo Jaffré, Pacôme Luton, and Thomas Pickles. Discopg: property graph schema discovery and exploration. Proc. VLDB Endow., Vol. 15, No. 12, p. 3654 – 3657, August 2022.

[5] Svetlozar Nestorov, Serge Abiteboul, and Rajeev Motwani. Inferring structure in semistructured data. ACM SIGMOD Record, Vol. 26, No. 4, pp. 39–43, 1997.

[6] Tushar Sharma, Marios Fragkoulis, Stamatia Rizou, Magiel Bruntink, and Diomidis Spinellis. Smelly relations: measuring and understanding database schema quality. In Proceedings of the 40th International Conference on Software Engineering: Software Engineering in Practice, ICSE-SEIP '18, p. 55 – 64, New York, NY, USA, 2018. Association for Computing Machinery.

[7] Roy Goldman and Jennifer Widom. Dataguides: Enabling query formulation and optimization in semistructured databases. In Proceedings of the 23rd International Conference on Very Large Data Bases (VLDB 1997), pp. 436–445. Morgan Kaufmann, 1997.

[8] Geert Jan Bex, Wim Martens, Frank Neven, and Thomas Schwentick. Expressiveness of xsds: From practice to theory, there and back again. In Proceedings of the 14th International Conference on World Wide Web (WWW 2005), pp. 712–721, 2005.

[9] Felipe Pezoa, L. Reutter, Juan Fernando Suárez, Martín Ugarte, and Domagoj Vrgoc. Foundations of json schema. In Proceedings of the 25th International Conference on World Wide Web (WWW '16), pp. 263–273, Geneva, Switzerland, 2016. International World Wide Web Conferences Steering Committee.

[10] Nicholas Gibbins and Nigel Shadbolt. Resource description framework (RDF). In Encyclopedia of Library and Information Sciences, pp. 1–30. Taylor & Francis, 2009.

[11] Grigoris Antoniou and Frank van Harmelen. Web ontology language: OWL. In Handbook on Ontologies, pp. 91–110. Springer, 2009.

[12] Renzo Angles. The property graph database model. In Proceedings of the 12th Alberto Mendelzon International Workshop on Foundations of Data Management (AMW 2018), 2018.

[13] Angela Bonifati, Peter Furniss, Alastair Green, Russ Harmer, Eugenia Oshurko, and Hannes Voigt. Schema validation and evolution for graph databases. In Proceedings of the 38th International Conference on Conceptual Modeling (ER 2019), pp. 448–456, 2019.

[14] Nimo Beeren and George Fletcher. A formal design framework for practical property graph schema languages. In Proceedings of the 26th International Conference on Extending Database Technology (EDBT 2023), pp. 478–484, 2023.

[15] Renzo Angles, Angela Bonifati, Stefania Dumbrava, George Fletcher, Alastair Green, Jan Hidders, Bei Li, Leonid Libkin, Victor Marsault, Wim Martens, Filip Murlak, Stefan Plantikow, Ognjen Savković, Michael Schmidt, Juan F. Sequeda, Sławek Staworko, Dominik Tomaszuk, Hannes Voigt, Domagoj Vrgoč, Mingxi Wu, and Dušan Živković. PG-Schema: Schemas for property graphs. Proceedings of the ACM on Management of Data, Vol. 1, No. 2, pp. 1–25, 2023.

[16] Renzo Angles, Angela Bonifati, Stefania Dumbrava, George Fletcher, Keith W. Hare, Jan Hidders, Victor E. Lee, Bei Li, Leonid Libkin, Wim Martens, Filip Murlak, Josh Perryman, Ognjen Savković, Michael Schmidt, Juan F. Sequeda, Sławek Staworko, and Dominik Tomaszuk. PG-Keys: Keys for property graphs. In Proceedings of the 2021 ACM SIGMOD International Conference on Management of Data (SIGMOD 2021), pp. 2423–2436, 2021.

[17] Philipp Skavantzos and Sebastian Link. Normalizing property graphs. Proceedings of the VLDB Endowment, Vol. 16, No. 11, pp. 3031–3043, 2023.

[18] Philipp Skavantzos and Sebastian Link. Third and boyce–codd normal form for property graphs. The VLDB Journal, Vol. 34, No. 2, 2025.

[19] Sam Skartsis, Alexey Volkov, and Olaf Hartig. Transforming RDF graphs to property graphs using standardized schemas. In Proceedings of the 2025 ACM SIGMOD International Conference on Management of Data (SIGMOD 2025) – Companion Volume, 2025.

[20] Chandan Sharma, Pierre Genevès, Nils Gesbert, and Nabil Layaïda. Schema-based query optimisation for graph databases. Proceedings of the ACM on Management of Data, Vol. 3, No. 1, pp. 1–29, 2025.

[21] Minos N. Garofalakis, Aristides Gionis, Rajeev Rastogi, S. Seshadri, and Kyuseok Shim. Xtract: A system for extracting document type descriptors from XML documents. In Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data (SIGMOD 2000), pp. 165–176. Association for Computing Machinery, 2000.

[22] Geert Jan Bex, Frank Neven, Thomas Schwentick, and Karl Tuyls. Inference of concise DTDs from XML data. In Proceedings of the 32nd International Conference on Very Large Data Bases (VLDB 2006), pp. 115–126. VLDB Endowment, 2006.

[23] Mohamed-Amine Baazizi, Dario Colazzo, Giorgio Ghelli, and Carlo Sartiani. Parametric schema inference for massive json datasets. The VLDB Journal, Vol. 28, No. 4, pp. 497–521, 2019.

[24] Redouane Bouhamoum, Kenza Kellou-Menouer, Stéphane Lopes, and Zoubida Kedad. Scaling up schema discovery for rdf datasets. In Proceedings of the 2018 IEEE 34th International Conference on Data Engineering Workshops (ICDEW), ICDEW '18, pp. 84–89, Piscataway, NJ, USA, 2018. IEEE.

[25] Artem Lutov, Soheil Roshankish, Mourad Khayati, and Philippe Cudré-Mauroux. Statix – statistical type inference on linked data. In Proceedings of the 2018 IEEE International Conference on Big Data (Big Data '18), Big Data '18, pp. 2253–2262, Piscataway, NJ, USA, 2018. IEEE.

[26] Kenza Kellou-Menouer and Zoubida Kedad. Schema discovery in rdf data sources. In Proceedings of the 34th International Conference on Conceptual Modeling (ER 2015), Vol. 9381 of Lecture Notes in Computer Science, pp. 481–495, 2015.

[27] Aldo Gangemi, Andrea G. Nuzzolese, Valentina Presutti, Francesco Draicchio, Alberto Musetti, and Paolo Ciancarini. Automatic typing of DBpedia entities. In Proceedings of the 11th International Semantic Web Conference (ISWC 2012), Vol. 7649 of Lecture Notes in Computer Science, pp. 65–81, 2012.

[28] Information technology — Database languages — GQL, 2024.

[29] Xiaohui Victor Li and Francesco Sanna Passino. Findkg: Dynamic knowledge graphs with large language models for detecting global trends in financial markets. In Proceedings of the 5th ACM International Conference on AI in Finance, ICAIF '24, p. 573 – 581. ACM, November 2024.
[30] Orri Erling, Alex Averbuch, Josep Larriba-Pey, Hassan Chafi, Andrey Gubichev, Arnau Prat, Minh-Duc Pham, and Peter Boncz. The ldbc social network benchmark: Interactive workload. In Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data, SIGMOD '15, p. 619 – 630, New York, NY, USA, 2015. Association for Computing Machinery.
[31] Takemura et al. A connectome of a learning and memory center in the adult *Drosophila* brain. eLife, Vol. 6, p. e26975, jul 2017.
[32] Neo4j. Network management graph example dataset. `https://github.com/neo4j-graph-examples/network-management`, 2025. Accessed 2025-01-04.
[33] Microsoft. Northwind sample database. `https://github.com/microsoft/sql-server-samples/tree/master/samples/databases/northwind-pubs`, 2025. Accessed 2025-01-04.
[34] Hamed Zamani, Markus Schedl, Paul Lamere, and Ching-Wei Chen. An analysis of approaches taken in the acm recsys challenge 2018 for automatic music playlist continuation. ACM Trans. Intell. Syst. Technol., Vol. 10, No. 5, September 2019.
[35] Transaction Processing Performance Council. Tpc benchmark™ h (decision support) standard specification. `https://www.tpc.org/tpc_documents_current_versions/pdf/tpc-h_v2.17.3.pdf`, 2017. Accessed 2025-01-04.
[36] Neo4j. Twitter v2 graph example dataset. `https://github.com/neo4j-graph-examples/twitter-v2`, 2025. Accessed 2025-01-04.
[37] George A. Miller. Wordnet: a lexical database for english. Commun. ACM, Vol. 38, No. 11, p. 39 – 41, November 1995.

付　　録

1 アルゴリズム

1.1 インスタンス由来スキーマの抽出アルゴリズム

アルゴリズム 1 は，プロパティグラフのインスタンス $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \lambda, \kappa, \sigma)$ から，インスタンス由来スキーマ $\mathcal{S}^{\mathcal{G}} = \mathbf{Abs}(\mathcal{G})$ を構成する．このアルゴリズムでは，構造が類似するインスタンスオブジェクトをまとめるために，ノードとエッジをそれぞれ互いに素な部分集合へ分割し，各部分集合から 1 つのノード型またはエッジ型を生成する．

(1) ノードの型の生成 アルゴリズム 1 の 4-16 行目がノード型の生成に対応する．ノードは，ラベル集合が完全に一致するもの同一のグループとしてまとめ，各グループからノード型を 1 つ生成する．

そのために，まずラベル集合 L を持つノードの集合 $\mathcal{V}_L$ を作成する（5 行目）．次に各グループ $\mathcal{V}_L$ ごとに新しいノード型 v_T を作成し，ラベル集合 L をノード型のラベル集合として設定する（6-8 行目）．さらに，グループ内のノードが持つプロパティの和集合をノード型のプロパティ集合として設定する（9 行目）．また，プロパティ k の必須性は，グループ内のすべてのノードが k を持つかどうかで決定し，その結果をノード型の必須性制約として設定する（10-14 行目）．継承関係は導入せず（15 行目），次のエッジ型の生成のためにラベル集合 L とノード型 v_T の対応を記憶しておく（16 行目）．

Algorithm 1: インスタンス由来スキーマの生成アルゴリズム

Input: $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \lambda, \kappa, \sigma)$
Output: $\mathcal{S}^{\mathcal{G}} = (\mathcal{V}_T^{\mathcal{G}}, \mathcal{E}_T^{\mathcal{G}}, \lambda^{\mathcal{G}}, \kappa^{\mathcal{G}}, \sigma^{\mathcal{G}}, \mu^{\mathcal{G}}, \rho^{\mathcal{G}})$

1: **function** Abs($\mathcal{G}$)
2: $\quad \mathcal{V}_T^{\mathcal{G}} \leftarrow \varnothing$;
3: $\quad \mathcal{E}_T^{\mathcal{G}} \leftarrow \varnothing$;
$\quad$ // ノード型の生成
4: $\quad$ **foreach** $L \in \{\lambda(v) \mid v \in \mathcal{V}\}$ **do**
5: $\quad\quad \mathcal{V}_L \leftarrow \{v \in \mathcal{V} \mid \lambda(v) = L\}$;
6: $\quad\quad v_T \leftarrow$ `createNewNodeType`();
7: $\quad\quad \mathcal{V}_T^{\mathcal{G}} \leftarrow \mathcal{V}_T^{\mathcal{G}} \cup \{v_T\}$;
8: $\quad\quad \lambda^{\mathcal{G}}(v_T) \leftarrow L$;
9: $\quad\quad \kappa^{\mathcal{G}}(v_T) \leftarrow \bigcup_{v \in O_L} \kappa(v)$;
10: $\quad\quad P_{\mathbf{req}} \leftarrow \bigcap_{v \in O_L} \kappa(v)$;
11: $\quad\quad$ **foreach** $k \in \kappa^{\mathcal{G}}(v_T)$ **do**
12: $\quad\quad\quad \mu^{\mathcal{G}}(v_T, k) \leftarrow$ `True`;
13: $\quad\quad\quad$ **if** $k \notin P_{\mathbf{req}}$ **then**
14: $\quad\quad\quad\quad \mu^{\mathcal{G}}(v_T, k) \leftarrow$ `False`;
15: $\quad\quad \rho^{\mathcal{G}}(v_T) \leftarrow \varnothing$;
16: $\quad\quad M[L] \leftarrow v_T$; $\quad$ // ラベル集合 $\mapsto$ ノード型の対応表
$\quad$ // エッジ型の生成
$\quad$ // $\tau(e) = (\lambda(e), \lambda(\mathbf{src}(e)), \lambda(\mathbf{dst}(e)))$
17: $\quad$ **foreach** tpl $\in \{\, \tau(e) \mid e \in \mathcal{E} \,\}$ **do**
18: $\quad\quad \mathcal{E}_{\text{tpl}} \leftarrow \{\, e \in \mathcal{E} \mid \tau(e) = \text{tpl} \,\}$;
19: $\quad\quad e_T \leftarrow$ `createNewEdgeType`();
20: $\quad\quad \mathcal{E}_T^{\mathcal{G}} \leftarrow \mathcal{E}_T^{\mathcal{G}} \cup \{e_T\}$;
21: $\quad\quad (L_e, L_{\mathbf{src}}, L_{\mathbf{dst}}) \leftarrow$ tpl;
22: $\quad\quad \lambda^{\mathcal{G}}(e_T) \leftarrow L_e$;
23: $\quad\quad \kappa^{\mathcal{G}}(e_T) \leftarrow \bigcup_{e \in \mathcal{E}_{\text{tpl}}} \kappa(e)$;
24: $\quad\quad P_{\mathbf{req}} \leftarrow \bigcap_{e \in \mathcal{E}_{\text{tpl}}} \kappa(e)$;
25: $\quad\quad$ **foreach** $k \in \kappa^{\mathcal{G}}(e_T)$ **do**
26: $\quad\quad\quad$ **if** $k \in P_{\mathbf{req}}$ **then**
27: $\quad\quad\quad\quad \mu^{\mathcal{G}}(e_T, k) \leftarrow$ `True`;
28: $\quad\quad\quad$ **else**
29: $\quad\quad\quad\quad \mu^{\mathcal{G}}(e_T, k) \leftarrow$ `False`;
30: $\quad\quad \sigma^{\mathcal{G}}(e_T) \leftarrow (M[L_{\mathbf{src}}], M[L_{\mathbf{dst}}])$;
31: $\quad\quad \rho^{\mathcal{G}}(e_T) \leftarrow \varnothing$;
32: $\quad$ **return** $(\mathcal{V}_T^{\mathcal{G}}, \mathcal{E}_T^{\mathcal{G}}, \lambda^{\mathcal{G}}, \kappa^{\mathcal{G}}, \sigma^{\mathcal{G}}, \mu^{\mathcal{G}}, \rho^{\mathcal{G}})$;

(2) エッジの型の生成 アルゴリズム 1 の 17-31 行目がエッジ型の生成に対応する．エッジは，$(\lambda(e), \lambda(\mathbf{src}(e)), \lambda(\mathbf{dst}(e)))$ の組が完全に一致するものを同一のグループとしてまとめ（18 行目），ノード型と同様に各グループからエッジ型を 1 つ生成する（19-27 行目）．最後に，エッジ型の端点ノード型を設定する（28 行目）．

1.2 継承関係の展開アルゴリズム

継承関係の展開アルゴリズムをアルゴリズム 2 に示す．このアルゴリズムでは，まず元のスキーマの属性・制約を展開後スキーマにコピーする（2-10 行目）．次に，各オブジェクト型に対して親・先祖オブジェクト型から属性・制約を継承する（11-17 行目）．この操作は，図 2 に示すスキーマのノード型 v_1^* のラベ

ル User や必須プロパティ id を，図 5 に示す展開後スキーマの v_2^+ に継承させる操作に対応する．続いて，各エッジ型に対して端点ノード型の子孫ノード型を考慮した新エッジ型を生成する（18-31 行目）．この操作は，図 2 に示すスキーマのエッジ型 e_1^* を図 5 に示す展開後スキーマの e_4^+ に展開する操作に対応する．最後に，継承関係を空にする（32-34 行目）．

Algorithm 2: 継承関係の展開アルゴリズム

Input: $\mathcal{S}^* = (\mathcal{V}_T^*, \mathcal{E}_T^*, \lambda, \kappa, \sigma, \mu, \rho)$：評価対象スキーマ
Output: $\mathcal{S}^+ = (\mathcal{V}_T^+, \mathcal{E}_T^+, \lambda^+, \kappa^+, \sigma^+, \mu^+, \rho^+)$：展開後スキーマ

1: **function** Flat($\mathcal{S}^*$)
    // 元のスキーマをコピー
2: $\mathcal{V}_T^+ \leftarrow \mathcal{V}_T^*$;
3: $\mathcal{E}_T^+ \leftarrow \mathcal{E}_T^*$;
4: **foreach** $o \in (\mathcal{V}_T^* \cup \mathcal{E}_T^*)$ **do**
5: $\lambda^+(o) \leftarrow \lambda(o)$;
6: $\kappa^+(o) \leftarrow \kappa(o)$;
7: **foreach** $k \in \kappa(o)$ **do**
8: $\mu^+(o,k) \leftarrow \mu(o,k)$;
9: **if** $o \in \mathcal{E}_T^*$ **then**
10: $\sigma^+(o) \leftarrow \sigma(o)$;
    // 親・先祖からの属性・制約を継承する
11: **foreach** $o \in (\mathcal{V}_T^* \cup \mathcal{E}_T^*)$ **do**
12: **foreach** $a \in \mathtt{Anc}(o)$ **do**
13: $\lambda^+(o) \leftarrow \lambda^+(o) \cup \lambda(a)$;
14: $\kappa^+(o) \leftarrow \kappa^+(o) \cup \kappa(a)$;
15: **foreach** $k \in \kappa(a)$ **do**
        // 子オブジェクト型が既に k の必須性制約を持つ場合は上書きしない
16: **if** $\mu^+(o,k)$ is undefined **then**
17: $\mu^+(o,k) \leftarrow \mu(a,k)$;
    // エッジ型の展開
18: **foreach** $e \in \mathcal{E}_T^*$ **do**
19: $(v_{\mathbf{src}}, v_{\mathbf{dst}}) \leftarrow \sigma(e)$;
20: $D'_{\mathbf{src}} \leftarrow \{v_{\mathbf{src}}\} \cup \mathtt{Desc}(v_{\mathbf{src}})$;
21: $D'_{\mathbf{dst}} \leftarrow \{v_{\mathbf{dst}}\} \cup \mathtt{Desc}(v_{\mathbf{dst}})$;
22: **foreach** $d_s \in D'_{\mathbf{src}}$ **do**
23: **foreach** $d_t \in D'_{\mathbf{dst}}$ **do**
24: **if** $(d_s, d_t) \neq (v_{\mathbf{src}}, v_{\mathbf{dst}})$ **then**
            // 端点ノード型のみが異なる新エッジ型を作成
25: $e' \leftarrow \mathtt{createNewEdgeType}()$;
26: $\mathcal{E}_T^+ \leftarrow \mathcal{E}_T^+ \cup \{e'\}$;
27: $\lambda^+(e') \leftarrow \lambda^+(e)$;
28: $\kappa^+(e') \leftarrow \kappa^+(e)$;
29: **foreach** $k \in \kappa^+(e)$ **do**
30: $\mu^+(e',k) \leftarrow \mu^+(e,k)$;
31: $\sigma^+(e') \leftarrow (d_s, d_t)$;
    // 継承関係を空にする
32: **foreach** $o \in (\mathcal{V}_T^+ \cup \mathcal{E}_T^+)$ **do**
33: $\rho^+(o) \leftarrow \varnothing$;
34: **return** $(\mathcal{V}_T^+, \mathcal{E}_T^+, \lambda^+, \kappa^+, \sigma^+, \mu^+, \rho^+)$;

// 補助関数
// $\mathtt{Anc}(o)$：ρ に基づき o から親方向に到達可能な全ての型を返す
// $\mathtt{Desc}(o)$：ρ に基づき o から子方向に到達可能な全ての型を返す

1.3 網羅性評価アルゴリズム

網羅性の評価アルゴリズムをアルゴリズム 3 に示す．なお，アルゴリズム 3 ではスキーマのノード網羅性 $\mathbf{Cvg}_V(\mathcal{G}, \mathcal{S}^*)$ を計算する場合を示しているが，エッジも同様の手順で計算できる．

Algorithm 3: ノード網羅性の評価アルゴリズム

Input: $\mathcal{G}$：インスタンス，$\mathcal{S}^*$：評価対象スキーマ
Output: $\mathbf{Cvg}_V(\mathcal{G}, \mathcal{S}^*)$

1: **function** CalcCvgV($\mathcal{G}, \mathcal{S}^*$)
2: $\mathcal{S}^{\mathcal{G}} \leftarrow \mathtt{Abs}(\mathcal{G})$; // インスタンス由来スキーマの抽出
3: $\mathcal{S}^+ \leftarrow \mathtt{Flat}(\mathcal{S}^*)$; // 継承関係の展開
4: **foreach** $v^{\mathcal{G}} \in V(\mathcal{S}^{\mathcal{G}})$ **do**
5: $\mathrm{max\_sim}[v^{\mathcal{G}}] \leftarrow \max_{v^+ \in V(\mathcal{S}^+)} \mathtt{sim}_V(v^{\mathcal{G}}, v^+)$;
6: **return** avg $([\,\mathrm{max\_sim}[v^{\mathcal{G}}] \mid v^{\mathcal{G}} \in V(\mathcal{S}^{\mathcal{G}})\,])$;

1.4 簡潔性評価アルゴリズム

簡潔性の評価アルゴリズムをアルゴリズム 4 に示す．なお，アルゴリズム 4 ではスキーマのノード簡潔性 $\mathbf{Con}_V(\mathcal{G}, \mathcal{S}^*)$ を計算する場合を示しているが，エッジの場合も同様の手順で計算できる．

Algorithm 4: ノード簡潔性の評価アルゴリズム

Input: $\mathcal{G}$：インスタンス，$\mathcal{S}^*$：評価対象スキーマ
Output: $\mathbf{Con}_V(\mathcal{G}, \mathcal{S}^*)$

1: **function** CalcConV($\mathcal{G}, \mathcal{S}^*$)
    // 元のスキーマの網羅性を計算
2: $\mathbf{cvg}_V \leftarrow \mathtt{CalcCvgV}(\mathcal{G}, \mathcal{S}^*)$;
3: $\mathbf{cvg}_E \leftarrow \mathtt{CalcCvgE}(\mathcal{G}, \mathcal{S}^*)$;
    // 寄与下限を計算
4: $\mathcal{S}^+ \leftarrow \mathtt{Flat}(\mathcal{S}^*)$;
5: $\epsilon_{\mathcal{V}} \leftarrow \dfrac{c_{\mathcal{V}}}{|\,\mathcal{V}_T^+\,|} \times \gamma$;
6: $\epsilon_{\mathcal{E}} \leftarrow \dfrac{c_{\mathcal{E}}}{|\,\mathcal{E}_T^+\,|} \times \gamma$;
7: $R \leftarrow 0$; // 冗長オブジェクト数
8: **foreach** $v^* \in \mathcal{V}_T^*$ **do**
        // ノード型 v^* を除去したスキーマで網羅性を再計算
9: $\mathbf{cvg}'_V \leftarrow \mathtt{CalcCvgV}(\mathcal{G}, \mathcal{S}^* \setminus \{v^*\})$;
10: $\mathbf{cvg}'_E \leftarrow \mathtt{CalcCvgE}(\mathcal{G}, \mathcal{S}^* \setminus \{v^*\})$;
        // 網羅性の低下量を計算
11: $\Delta_V \leftarrow \mathbf{cvg}_V - \mathbf{cvg}'_V$;
12: $\Delta_E \leftarrow \mathbf{cvg}_E - \mathbf{cvg}'_E$;
        // 網羅性の低下量が寄与下限以下なら冗長とみなす
13: **if** $\Delta_{\mathcal{V}} < \epsilon_{\mathcal{V}} \ \wedge \ \Delta_{\mathcal{E}} < \epsilon_{\mathcal{E}}$ **then**
14: $R \leftarrow R + 1$;
15: **return** $1 - \dfrac{R}{|\,\mathcal{V}_T^*\,|}$;

2 計算量の解析

本節では，提案手法の計算量について整理する．以降，インスタンスを $\mathcal{G}$，評価対象スキーマを $\mathcal{S}^*$，展開後スキーマを $\mathcal{S}^+ = \mathbf{Flat}(\mathcal{S}^*)$，インスタンス由来スキーマを $\mathcal{S}^{\mathcal{G}} = \mathbf{Abs}(\mathcal{G})$ とする．

まず，前処理である $\mathbf{Abs}(\mathcal{G})$ は，インスタンス中の各ノードおよびエッジを一度ずつ走査し，ラベル集合およびプロパティ集合に基づいて分類を行うため，計算量は $O(|V(\mathcal{G})| + |E(\mathcal{G})|)$ である．同様に，$\mathbf{Flat}(\mathcal{S}^*)$ は，スキーマの継承関係を推移的に展開する操作であり，スキーマサイズに対して多項式時間で実行可能である．

次に，網羅性（Coverage）の計算では，$\mathcal{S}^{\mathcal{G}}$ に含まれる各ノード型（エッジ型）に対して，$\mathcal{S}^+$ に含まれるすべてのノード型（エッジ型）との類似度を計算し，最大値を求める．したがって，ノード網羅性およびエッジ網羅性の計算量は，それぞれ $O(|V(\mathcal{S}^{\mathcal{G}})||V(\mathcal{S}^+)|)$，$O(|E(\mathcal{S}^{\mathcal{G}})||E(\mathcal{S}^+)|)$ である．

簡潔性（Concision）の計算では，スキーマ中の各オブジェクト型 $o \in \mathcal{S}^*$ を 1 つずつ除去し，その都度，網羅性の低下量を評価する．素朴に実装した場合，各オブジェクト除去ごとに網羅性を再計算する必要があるため，計算量は $O(|V(\mathcal{S}^*)||V(\mathcal{S}^{\mathcal{G}})||V(\mathcal{S}^+)|)$（エッジについても同様）となる．

しかし，本研究では，$\mathcal{S}^{\mathcal{G}}$ と $\mathcal{S}^+$ の間で計算される類似度および最大類似度の結果をメモ化し，オブジェクト除去のたびに再利用することで，重複する計算を省いている．この最適化により，簡潔性計算において新たに必要となる計算は，除去対象に対応する一部の寄与の更新に限られ，網羅性全体を再計算する必要はない．その結果，簡潔性計算を含む全体の計算量は，実効的に $O(|V(\mathcal{S}^{\mathcal{G}})||V(\mathcal{S}^+)| + |E(\mathcal{S}^{\mathcal{G}})||E(\mathcal{S}^+)|)$ に抑えられる．

以上をまとめると，提案手法の計算量は

$$O(|V(\mathcal{G})| + |E(\mathcal{G})| + |V(\mathcal{S}^{\mathcal{G}})||V(\mathcal{S}^+)| + |E(\mathcal{S}^{\mathcal{G}})||E(\mathcal{S}^+)|)$$

である．一般に，スキーマサイズは対応するインスタンスサイズに比べて十分に小さいことが多いため，実運用においては $|V(\mathcal{G})|$ および $|E(\mathcal{G})|$ による走査コストが支配的になると考えられる．実際に，本研究のスケーラビリティ評価実験においても，実行時間はインスタンスサイズの増加に対してほぼ線形に増加しており，本解析と整合した挙動が確認された．

3 実験設定

3.1 提案手法のハイパーパラメータ

本研究で用いたハイパパラメータの値は $\alpha = 0.5$，$\beta = 0.5$，$\gamma = 0.15$. である．

α はノード型およびエッジ型の類似度計算において，ラベル一致とプロパティ一致の相対的な重要度を調整するパラメータである．$\alpha = 0.5$ とすることで，ラベルとプロパティの双方を同程度に考慮する中立的な設定とした．

β はエッジ型の類似度計算において，エッジ本体の属性と，接続先ノード型の一致度の寄与を調整するパラメータである．本研究では，エッジの意味と構造のいずれも同様に重要であると考え，$\beta = 0.5$ に固定した．

γ は簡潔性の計算において，冗長オブジェクトを判定する際の厳しさを制御するパラメータである．$\gamma = 0.15$ は，LDBC，Northwind，Spotify，TPC-H の各データセットにおける正解スキーマを冗長と判定しない上限値として経験的に決定した．これよりも大きな値を用いると，LDBC のように比較的スキーマサイズの大きい正解スキーマにおいて，明らかにインスタンスのデータ構造を説明するために必要と判断されるオブジェクト型が冗長と判定される事例が確認された．本研究では実験条件を単純化するため，冗長判定に対して比較的緩やかな値である $\gamma = 0.15$ を全スキーマに対して一律に用いた．

インスタンスや人間の評価特性に応じた適切なハイパーパラメータの自動調整手法の検討は今後の課題である．

3.2 スキーマ抽出手法のハイパーパラメータ

本研究で用いたスキーマ抽出手法のハイパーパラメータ設定を表 A·1 に示す．

表 A·1: 各スキーマ抽出手法のハイパーパラメータ設定

手法	ハイパーパラメータ	値
類似度ベース クラスタリング手法 [1]	l	1.0
	p	0.0
	t	0.0
	k	0
	θ	0.5
ルールベース手法 [2]	なし	–
階層型 クラスタリング手法 [3]	sampling rate	0.7
	top_k_props	1
	num_clusters	2