属性を用いた問合せの効率化に向けたグラフデータベースシステムの併用

楠　和馬†　波多野賢治††

† 同志社大学大学院文化情報学研究科　〒 610–0394 京都府京田辺市多々羅都谷 1–3
†† 同志社大学文化情報学部　〒 610–0394 京都府京田辺市多々羅都谷 1–3
E-mail: †{kkusu,hatano}@acm.org

あらまし　グラフは，実体のある情報を節点，実体同士の関係を辺で表現することができ，ソーシャルネットワークや地理情報，Web 文書間リンク情報などのような関係データを表現することができる．また，節点と辺のそれぞれに属性を持たせられるプロパティグラフモデルは，関係データの他にトランザクションデータも扱うことが可能である．このようにデータへの適用性の高さから，グラフデータベースにおいて，プロパティグラフモデルは一般的なモデルとなっている．しかしグラフデータベースにおいて，属性を利用した問合せの際には基本的に節点や辺の読込みが先行しており，属性値による絞込みはその後に実行されている．この処理はデータベースに格納されているデータ量と問合せ性能が直接的に依存することになるため，グラフデータベースにおけるグラフ問合せ性能に改善の余地がある．そこで本研究では，節点同士の接続情報と節点の属性情報をそれぞれ別のグラフモデルで管理することにより，直接的に特定の属性値を持つ節点の読込みを可能にする．また性能評価実験では，データベースベンチマークを利用することで，本研究で提案するグラフデータベースの問合せ性能とそのスケーラビリティについて評価する．

キーワード　グラフデータベースシステム, プロパティグラフモデル, ハイパーグラフモデル

1 はじめに

近年の電子商取引（以下，EC）サイトやソーシャルネットワーキングサービス（以下，SNS）の普及により，企業が保有するデータは巨大で複雑になってきており，グラフデータとして管理する方が妥当な現状にある．グラフは，データ内で実体のある事物（以下，エンティティ）およびエンティティ間に存在する関係（以下，リレーション）を，それぞれ節点および辺で表現することができる．またグラフは，他のデータモデルでは直接的に表現できないデータ内のリレーションを扱えるため，EC サイトにおけるトランザクションデータや SNS 上の人間関係をデータ化したソーシャルグラフなどのように多様な分野でグラフが扱われている．このような巨大で複雑なグラフデータに対する高速な問合せや効率的な分析への需要が高まっているため，多くのグラフデータベースシステム（以下，GDB）が開発されている．

汎用的にグラフデータを扱うことが可能な GDB はプロパティグラフ（以下，PG）モデルと呼ばれるグラフモデルでデータを格納している．PG モデルは節点および辺に属性を持つようなグラフデータを柔軟に表現することができるため，多様な分野におけるグラフデータに適用可能なグラフモデルである．また，PGQL や Cypher, GCORE [1,3,12] のような PG モデル用問合せ言語も多く提案されており，それら問合せ言語を統合する方策も立てられている [1]．このように，汎用的なグラフデータ管理を行う GDB において PG モデルがデファクトスタンダードになっている．しかし，PG モデルを採用している GDB は任意の属性を持っている節点を効率的に読み込むことができない．この原因は，既存の GDB では，節点や辺が持つ属性を個別に格納するようにして管理しており，直接的にグラフの要素を読み込まなければ特定の属性を持つか否か判断することができない．それゆえに，属性を利用したグラフ走査やグラフ分析を行うグラフ問合せのパフォーマンスを向上させるためには，指定された属性を持つ節点を直接的に GDB から読込み可能にする必要がある．

そこで本研究では，節点および辺情報とそれらの属性情報を分割して管理することにより，特定の属性を保有している節点を直接的に読込み可能な GDB アーキテクチャを提案する．また，本研究で提案する GDB のグラフ問合せ性能やスケーラビリティについて評価するために，ベンチマークを利用した評価実験を行う．

1 : The GQL Manifest: One Property Graph Query Language, https://gql.today/（2019 年 2 月 12 日閲覧）.

2 グラフデータベースのグラフモデル

1 節で述べたように，グラフデータを取り扱うアプリケーションは，SNS や EC サイトなどのように数多く存在している．節点および辺で基本的に構成されるグラフの種類には，下記のようにアプリケーションに合わせて複数存在している [9].

Multi-graph:　一つの節点対の間で複数の辺を持つことが許されているグラフである．

Vertex-labeled graph:　エンティティの種別を意味するラベルが節点に付いているグラフである．

Edge-labeled graph: リレーションの種別を意味するラベルが辺に付いているグラフである．

Vertex-attributed graph: エンティティに関する属性情報が節点に付随しているグラフである．

Edge-attributed graph: リレーションに関する属性情報が辺に付随しているグラフである．

Directed graph: リレーションに方向性を有する辺から成るグラフである．

Resource description framework（以下，RDF）: World Wide Web コンソーシアム [2]により開発されたグラフ標準であり，節点や辺を Uniform Resource Identifiers（以下，URI）で表したグラフである．

Hypergraph: 任意数の節点間でリレーションを定義可能なハイパエッジから成るグラフである．

このように，多様なグラフの形状がアプリケーションに合わせて定義されてきており，グラフ処理だけではなくグラフデータの管理に関する要求も徐々に高まりつつある．グラフデータの管理および処理に特化したデータベースである GDB は，基本的に以下のモデルが採用されている [8].

Property Graph Model（以下，PG モデル）: PG モデル は Multi-graph，Vertex/Edge labeled graph，Vertex/Edge attributed graph，Directed graph の六つの特徴を組み合わせたグラフモデルである．このモデルはデータの表現能力が高く，多くの GDB においてサポートされている．

RDF: Linked Open Data のデータ記述方法として標準化されている．前述の通り，節点や辺の記述方法が W3C により厳格に定められているため，データに関する理解が容易にできる一方で，表現力が低くなる．また，節点自体がエンティティではなく，属性値を表現することもあり，その値の数だけ節点や辺の数が増加する．

Hyper Graph Model（以下，HG モデル）: グラフ理論における定義では，ハイパエッジは任意数の節点集合と定義されている．その定義を模して，同一の関係性を持つ任意数の節点に対して，その関係性を一つのハイパーエッジで表現することができる．また，データ管理において辺の数を削減することができ，ハイパーエッジ同士の集合演算を実行することが可能である．しかし，ハイパーエッジ内の節点間で属性情報の違いを表現することは不可能である．そのため，節点が保有する属性値の違いを表現したい数だけハイパーエッジを作成する必要がある．

3 提 案 手 法

既存の GDB は，グラフ問合せで指定された属性値を持つか否かに関係なく，問合せ対象に該当する全節点をデータベースから読み込んでいる．そのため既存の GDB では，グラフ問合せ内で指定された属性値を持たないような，問合せ結果の導出に不要な節点まで計算機のメモリに読み込んでしまっている．これにより，グラフ問合せの結果導出は GDB に格納されたグラフ要素の数に依存した性能になり，データベースサイズ料に対してグラフ走査や分析の処理時間はスケールしないと考えられる．この問題を解決するためには，GDB は問合せ内で指定されている属性値を持つ節点の読込みが直接的に行える必要がある．

2 節で述べたように，PG モデルは多様なグラフデータを表現することができ，効率的なグラフ走査を可能にする．PG モデルを利用したグラフ走査は，大規模なグラフにおいても効率的に実行することができることから，そのパフォーマンスを線形的にスケールすることが可能になる [10]．グラフデータの属性情報の管理については，問合せ内で指定された属性を持つ節点を直接的に読み込めるように設計可能なグラフモデルを選択する必要がある．また，属性を利用したグラフ分析のパフォーマンスを向上させることが可能なグラフの管理方法である必要がある．2 節で紹介した中から本研究では，上記の要件を満たすグラフモデルとして，ハイパーエッジにより節点集合を管理可能で，集合演算を適用可能な HG モデルを選択する．HG モデルを採用することにより，問合せ内で実行される集約演算を集合演算で効率的に実行可能にする．

したがって本研究では，PG モデルの GDB と HG モデルの GDB を併用することで，属性を利用したグラフ問合せパフォーマンスの効率化が可能な GDB アーキテクチャを提案する．

3.1 節点および辺情報のデータ管理

節点および辺情報のデータ管理には，グラフ走査のパフォーマンスの高い GDB を扱う．そのため本研究では，他の GDB よりもグラフ走査の性能が高いと報告されている Neo4j を用いる [5,6]．Neo4j で管理するグラフ要素は，管理対象としているグラフデータの節点，辺，及びこれらのラベルと属性を格納する．基本的に全てのデータを格納するが，Neo4j 上での属性へのアクセスは，グラフ問合せの結果を表示する際に要求された節点および辺の属性を表示するときにのみである．

また，グラフ問合せを向上させる方策として，インデックスの構築を行う必要がある．Neo4j で管理するグラフデータに対しては，属性に基づく検索処理に関しては HG モデルの GDB により行う．そのため，Neo4j の仕様によって生成される，節点や辺のラベルに対するインデックスのみを利用する．

したがって本手法では，グラフ問合せの際にグラフ要素の属性を利用する場合に，グラフ走査に必要な開始節点の読込みは HG モデルで行い，それを基に Neo4j でグラフ走査処理を実行する．一方，属性を利用しないグラフ問合せの場合，グラフ走査に関連する全処理を Neo4j で実行する．つまり，属性を利用しないグラフ問合せの場合，グラフ走査処理は Neo4j の性能に近似する．

3.2 属性情報のデータ管理

本手法では，グラフ問合せで指定された属性を持つ節点や属性を利用したグラフ分析を効率に行えるようにする．その実現には，2 節で紹介した，ハイパーエッジにより節点集合を定義でき，集合演算の適用が可能な HG モデルの GDB を用いる．

2 : W3C: https://www.w3.org/（2019 年 2 月 12 日閲覧）.

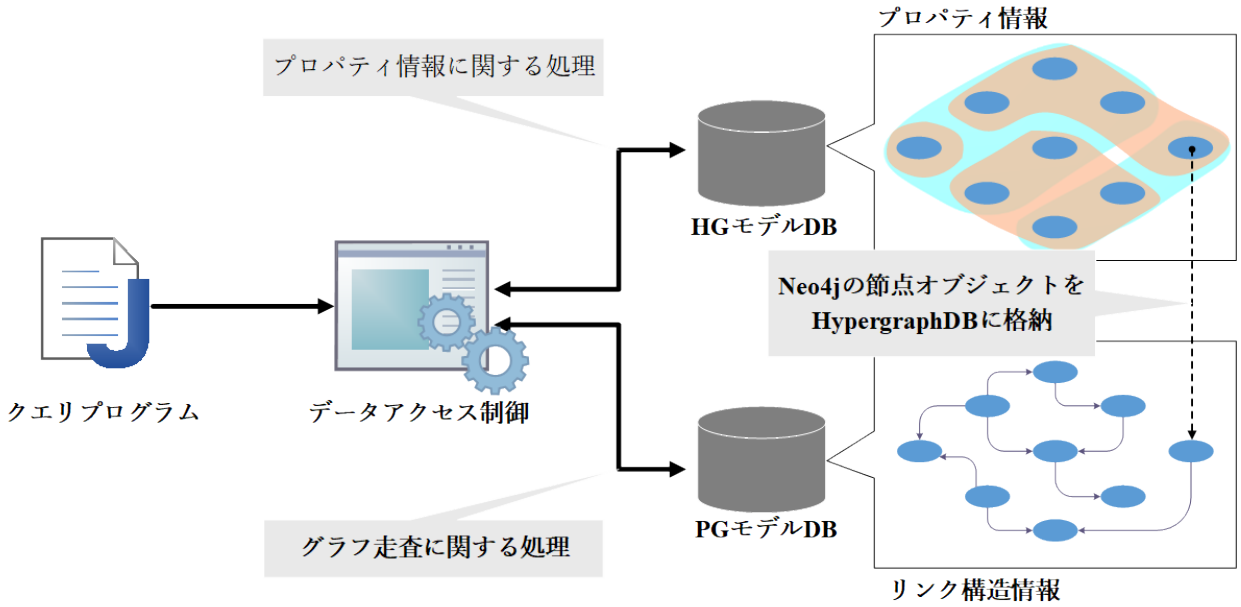

図 1　2 種 GDB を併用した GDB アーキテクチャ

HG モデルを採用した既存の GDB は HypergrahDB のみであるため，本手法では HypergraphDB をグラフ要素の属性情報を管理する GDB として利用する [4]．HypergraphDB は Java 言語で実装された GDB であり，格納できる要素として `atom` と `link` を定義している．`atom` は HypergraphDB における用語で，Java 言語における基本値やクラスのインスタンスを `atom` として格納することが可能である．2 節で説明した多項関係性の存在を実現するために，HypergraphDB では `link` で多項関係を表現することが可能になる．つまり，`link` の構成要素は `atom` であり，任意の数を `link` に含めることが可能である．また，この `link` は一つの `atom` として定義されているため，関係の関係という高次の関係性を表現することが可能になる．

そこで，本手法では Neo4j 社が提供する Java API を用いて，節点の Java オブジェクトを HypergraphDB に `atom` として管理する．また，`link` に含まれる `atom` の意味合いを「ある属性の特定値を保有する節点」となるように，HypergraphDB に `link` を格納する．

4　評 価 実 験

本節では，本研究で提案する手法の有用性を評価するための実験方法について説明する．本研究で提案する GDB アーキテクチャのスケーラビリティを保証するために，さまざまなデータ量のデータセットを生成でき，具体的なグラフ問合せが用意されているベンチマークを利用する．

4.1　データセット

本研究では，データ量をスケールさせることが可能で，属性が付いた節点に対する問合せが同梱されている，非営利団体の Linked Data Benchmark Council（以下，LDBC）が提供する Social Network Benchmark（以下，SNB）を用いる．LDBC

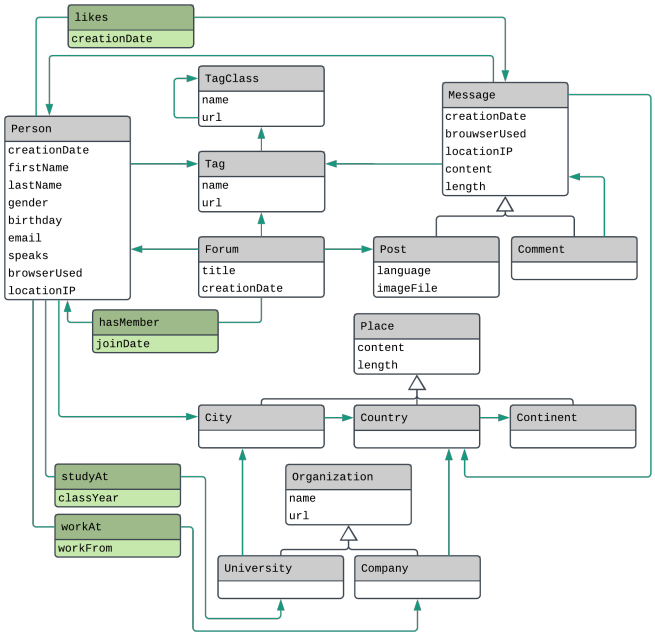

図 2　LDBC SNB データのグラフモデル

は，近年におけるデータの膨大化・分散化・複雑化に伴い，要求が高まりつつあるグラフ処理および分析の性能を計測可能にすることを目的としている．LDBC が提供している SNB は，グラフを取り扱うサービスとして代表的な SNS サイトで管理されるようなデータをモデル化したものである．LDBC SNB のグラフモデルを図 2 に示す [3]．また，さまざまなデータ規模での問合せ性能の計測を可能にするため，Scale Factor（以下，SF）の値によってデータ量を制御することが可能になる．さらに SNB のグラフ問合せには，Interactive Workload [2] と Business Intelligence Workload [11] の 2 種類が用意されてい

3：これは LDBC SNB のマニュアル [7] に掲載されている．

る．Interactive Workload は，サービス上でユーザが扱う情報の検索および更新に関するトランザクションのテストセットであり，その中に簡単なグラフ走査を行う Interactive Short（以下，IS）Workload が 7 種類，複雑なグラフ走査や分析を行う Interactive Complex（以下，IC）Workload が 14 種類，グラフ要素の追加・削除・更新を行う命令が 8 種類用意されている．Business Intelligence Workload は，サービス運営者がビジネスに役立つ情報を得る目的で行うようなデータ分析に関するトランザクションのテストセットであり，分析的なグラフパタン検索が 25 種用意されている．このように，LDBC SNB には計 54 種類のグラフ問合せが用意されている．

本研究では，LDBC SNB のデータ生成プログラム [4] を用いて，SF 値が 0.1, 1, 10 に対応するデータセットを生成する．次に，生成した LDBC SNB データセットは，LDBC SNB のデータ格納および問合せパラメタ検証用プログラム [5]を用いて，データセットを Neo4j へ格納し，有効な問合せパラメタの生成を行う．

各 SF 値の LDBC SNB データセットを Neo4j に格納し，節点・辺・プロパティそれぞれの数を表 1 に示す．

表 1　Graph datasets

dataset & SF	#nodes	#edges	#properties
LDBCSNB-SF0.1	327,588	1,477,965	1,866,081
LDBCSNB-SF1	3,181,724	17,256,038	18,998,125
LDBCSNB-SF10	29,987,835	176,623,445	180,532,296

4.2　ベンチマークの実行方法

本実験では，4.1 節で挙げた LDBC SNB に用意されている Interactive Workload を実行することで，本研究で提案する GDB アーキテクチャのグラフ走査およびグラフ分析の性能を評価する．4.1 節で述べた通り，Interactive Workload には IS Workload と IC Workload があり，これらを Neo4j および HypergraphDB の Java API[6][7]によって

本研究で提案する GDB アーキテクチャのグラフデータ管理方法の有効性を示すために，他のグラフデータ管理方法との比較評価を必要がある．そのため，本研究で提案する格納方法の他に，以下に挙げる 2 種類のグラフデータ管理方法を用意する．

PurePGM:　この格納方法では PG モデルに準じたグラフ格納方法である．そのため，LDBC SNB で生成されるグラフデータをそのまま Neo4j に格納する．本研究で提案する手法と PurePGM と比較することにより，提案手法がグラフ走査性能およびグラフ分析の性能を向上させることができたか考察する．

4 : LDBC:　ldbc_snb_datagen.git,　https://github.com/ldbc/ldbc_snb_datagen（2019 年 2 月 12 日閲覧）.

5 : LDBC:　ldbc_snb_implementation.git,　https://github.com/ldbc/ldbc_snb_implementation（2019 年 2 月 12 日閲覧）.

6 : Neo4j, Inc.: Package org.neo4j.graphdb.traversal, https://neo4j.com/docs/java-reference/3.5/javadocs/org/neo4j/graphdb/traversal/package-summary.html（2019 年 2 月 12 日閲覧）.

7 : Kobrix Software, Inc.: HyperGraphDB Core 1.3-SNAPSHOT API, http://hypergraphdb.org/docs/javadoc/（2019 年 2 月 12 日閲覧）.

表 2　グラフ問合せの特徴に関する変数

query	# parameters	# hops	# aggregate functions
IS1	1	1	0
IS2	1	3 - ?	0
IS3	1	1	0
IS4	1	0	0
IS5	1	1	0
IS6	1	3 - ?	0
IS7	1	4	0
IC1	2	6 - 8	0
IC2	2	2	0
IC3	5	5 - 6	3
IC4	3	6	1
IC5	2	4 - 5	1
IC6	2	4 - 5	1
IC7	1	3 - ?	0
IC8	1	3	0
IC9	2	2 - 3	0
IC10	2	5	1
IC11	3	3 - 4	0
IC12	2	5 - ?	1
IC13	2	?	0
IC14	2	?	0

A - B: この表記は値域が [A, B] であることを示す．?: この表記はデータセットに依存するため不明であることを示す．

IndexedPGM:　この格納方法では，LDBC SNB で生成されるグラフデータをそのまま Neo4j に格納し，全ての節点の属性にインデックスを構築する．本研究で提案する手法と IndexedPGM を比較することにより，Neo4j で構築される節点属性のインデックスの利用時と性能の違いについて考察する．

また，グラフ問合せ内容と問合せ性能の関係性を分析するために，Interactive Workload に含まれているグラフ問合せの特徴を表す 3 種類の変数を用意する．

 # parameters:　グラフ問合せ実行時に必要なパラメタの数を示している．

 # hops:　グラフ問合せ実行中に発生する辺の走査回数を示している．

 # aggregate functions:　グラフ問合せで利用される集約関数および副問合せの数を示している．

これら変数とグラフ問合せの性能との関連を見ることで，本研究で提案した手法がどのようなグラフ問合せの際にグラフ走査およびグラフ分析のパフォーマンスを向上させることができたか否か，を観察することに役立てる．Interactive Workload に含まれている IS1 から IS7 の Interactive Short および IC1 から IC14 の Complex クエリそれぞれに対して，3 種類の変数を計上した結果を表 2 に示す．

最後に，本実験の環境は Microsoft Azure 上でサイズ構成が Standard_D64_v3 の Linux 仮想マシンで行う．また，各種データベースシステムのバージョンは実験時点で最新版である Neo4j ver. 3.4.7 および HypergraphDB ver. 1.3 を用いる．

4.3 予備実験

本実験の前に，グラフ走査の性能を悪化させずに，PG モデル GDB と HG モデル GDB を併用することが可能か検証するために，Interactive Workload の Short クエリを実行する．この Short クエリは節点 ID による読込み後，単一の節点に接続する辺をいくつか走査するクエリであるため，純粋なグラフ走査性能を計測することができる．

Short クエリの実行結果を図 3 に示す．この結果より，IS1 から IS3 に関してはデータ量が指数関数的に増加しているが問合せ性能への影響は小さかったが，IS4 から IS7 に関しては問合せ性能もデータ量の増加に従って指数関数的に悪化していることが分かる．これは，IS1 から IS3 は LDBC SNB データセットの `Person` ラベルが付いた節点に対してグラフ走査を行う問合せであり，IS4 から IS7 は `Message` ラベルが付いた節点に対してグラフ走査を行う問合せである．`Message` ラベルの付いた節点は `Person` ラベルの付いた節点に比例して増大していくため，SF 値の増加に従って指数関数的に問合せ性能が悪化したといえる．Neo4j 単体の場合と Neo4j と HypergraphDB 併用の場合での問合せ実行に差は生じなかった．その理由としては，本研究の提案手法では HypergraphDB へのアクセスは節点の属性を利用する場合のみであるため，節点 ID による節点読込みのみの Short クエリでは HypergraphDB を利用した節点読込みは行われなかったためである．したがって，グラフ問合せの性能を悪化させることは生じなかったといえる．

5 おわりに

本研究では，PG モデルのグラフデータを効率的に管理するために，2 種類の GDB を併用することで節点と辺情報およびそれらの属性情報を分割して管理する GDB アーキテクチャを提案した．また，本研究で提案する手法の有用性を評価するため，LDBC SNB を利用したグラフ問合せの性能評価に関する実験の方法についても説明した．さらに，LDBC SNB に用意されている Short クエリを実行し，グラフ走査性能を悪化させることなく 2 種類の GDB を併用可能か検証する簡易な実験を行った．その結果としては，本研究の提案手法は Neo4j と同等の性能でグラフ走査を行うことが可能であったが，問合せ性能がスケールしない問合せが存在していた．

今後の課題は，LDBC SNB の IC Workload および Business Workload を実行し，提案手法のグラフ問合せおよび分析性能に関する評価実験を行い，有用性について検証していく必要がある．また，本稿では SF 値を 100 に設定してデータセットを生成し，更に大規模のデータセットに対する評価実験も行う必要がある．さらに，本稿で行った予備実験により，属性を利用しない問合せの場合には性能の向上に全く貢献することができていないため，属性を利用しない場合に HG モデルの GDB をどのように活用していくか模索する必要がある．

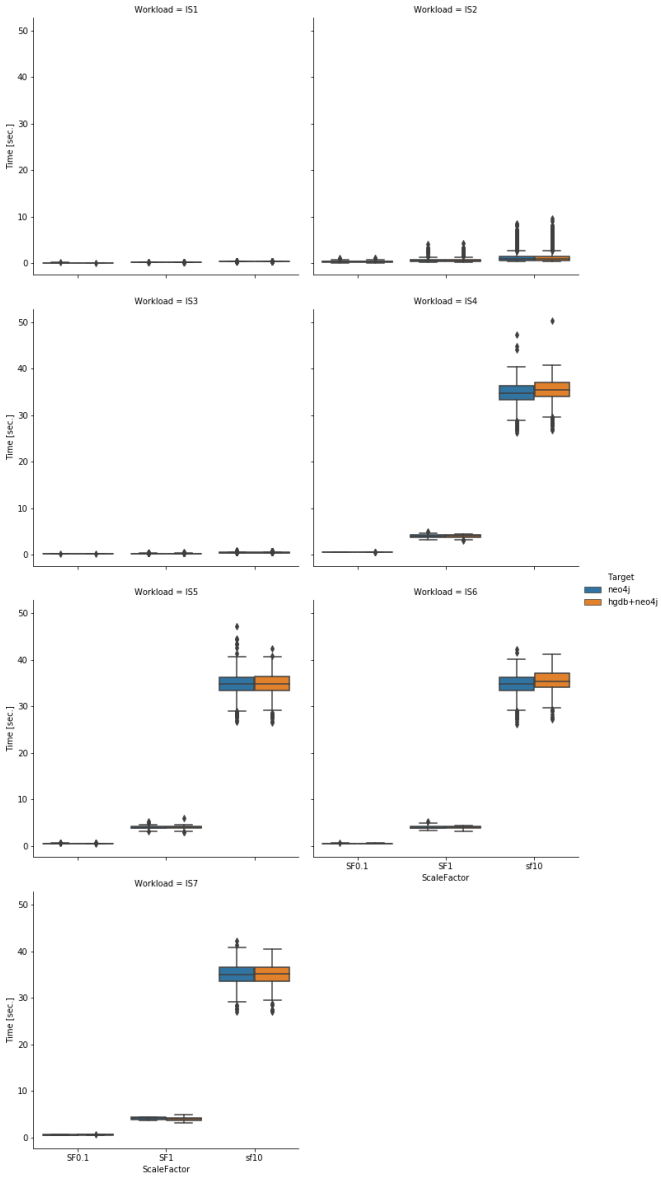

図 3 Interactive Workload のクエリ性能

謝辞

本研究の一部は文部科学省私立大学戦略的研究基盤形成支援事業，JSPS 科研費 JP18H03242/JP18H03342 の助成および同志社大学大学院文化情報学研究科の研究推進補助金を受けて遂行された．

文献

[1] Renzo Angles, Marcelo Arenas, Pablo Barcelo, Peter Boncz, George Fletcher, Claudio Gutierrez, Tobias Lindaaker, Marcus Paradies, Stefan Plantikow, Juan Sequeda, Oskar van Rest, and Hannes Voigt. G-core: A core for future graph query languages. In *Proceedings of the 2018 International Conference on Management of Data*, SIGMOD '18, pp. 1421–1432. ACM, 2018.

[2] 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*, SIG-

MOD '15, pp. 619–630, New York, NY, USA, 2015. ACM.

[3] Nadime Francis, Alastair Green, Paolo Guagliardo, Leonid Libkin, Tobias Lindaaker, Victor Marsault, Stefan Plantikow, Mats Rydberg, Petra Selmer, and Andrés Taylor. Cypher: An evolving query language for property graphs. In *Proceedings of the 2018 International Conference on Management of Data*, SIGMOD '18, pp. 1433–1445. ACM, 2018.

[4] Borislav Iordanov. Hypergraphdb: A generalized graph database. In *International Conference on Web-Age Information Management*, pp. 25–36. Springer, 2010.

[5] Salim Jouili and Valentin Vansteenberghe. An empirical comparison of graph databases. In *Proceedings of the 2013 International Conference on Social Computing*, SOCIALCOM '13, pp. 708–715. IEEE Computer Society, 2013.

[6] Vojtěch Kolomičenko, Martin Svoboda, and Irena Holubová Mlýnková. Experimental comparison of graph databases. In *Proceedings of International Conference on Information Integration and Web-based Applications & Services*, IIWAS'13, pp. 115:115–115:124. ACM, 2013.

[7] Linked Data Benchmark Council. *LDBC The graph & RDF benchmark reference The LDBC Social Network Benchmark (version 0.3.1)*.

[8] Ian Robinson, Jim Webber, and Emil Eifrem. *Graph Databases*. O'Reilly Media, Inc., 2015.

[9] Marko A. Rodriguez and Peter Neubauer. Constructions from dots and lines. *Bulletin of the American Society for Information Science and Technology*, Vol. 36, No. 6, pp. 35–41, 2010.

[10] Sherif Sakr and Eric Pardede. *Graph Data Management: Techniques and Applications*. IGI Global, 2011.

[11] Gábor Szárnyas, Arnau Prat-Pérez, Alex Averbuch, József Marton, Marcus Paradies, Moritz Kaufmann, Orri Erling, Peter Boncz, Vlad Haprian, and János Antal Benjamin. An early look at the ldbc social network benchmark's business intelligence workload. In *Proceedings of the 1st ACM SIGMOD Joint International Workshop on Graph Data Management Experiences & Systems (GRADES) and Network Data Analytics (NDA)*, GRADES-NDA '18, pp. 9:1–9:11, New York, NY, USA, 2018. ACM.

[12] Oskar van Rest, Sungpack Hong, Jinha Kim, Xuming Meng, and Hassan Chafi. Pgql: A property graph query language. In *Proceedings of the Fourth International Workshop on Graph Data Management Experiences and Systems*, GRADES '16, pp. 7:1–7:6. ACM, 2016.