

【VLDB2011勉強会】

Session 11: Graph Data

担当：井上（筑波大学）

Session 11 について

▶ Graph Data

- ▶ 主に, RDF, SPARQLなどに関する処理
- ▶ (2つ目は異なる)

▶ 前提知識

- ▶ RDF (Resource Description Framework)
 - ▶ リソース間の関係を記述するフレームワーク
 - ▶ Subject, Predicate, Object. ラベル付き有向グラフ
- ▶ SPARQL
 - ▶ RDF問合せ言語の一つ. 問合せクエリはグラフパターンが含まれる
 - ▶ 処理系はグラフのマッチングを行う
- ▶ RDFストア

[1]gStore: Answering SPARQL Queries via Subgraph Matching

- ▶ Lei Zou, Jinhui Mo, Lei Chen, M. Tamer, Dongyan Zhao
- ▶ 概要
 - ▶ ワイルドカードを用いた、大規模なRDFデータに対するスケーラブルなSPARQL問合せにを実現
 - ▶ 既存のRDF-Triple ストア
 - ▶ ワイルドカードを用いたクエリに対応するもので、スケーラブルなものがない
 - ▶ スケーラブルなもので、ワイルドカードを用いたクエリに対応したものがない
 - ▶ 対応するためにRDFデータを大きな木として格納
 - ▶ SPARQLクエリは、サブ グラフ マッチングとして処理
 - ▶ 高速な処理のための新しいインデックス手法を開発
 - ▶ 効率的な枝刈り, 検索アルゴリズム
 - ▶ (RDFリポジトリの頻繁な更新を実現)

前処理

1. RDFグラフをディスク格納に適した隣接リストに変換
 1. RDFグラフの頂点(エンティティ, クラス)を符号化
 2. (メンテナンスオーバーヘッドを軽減する,) VS-Tree を構成
- ▶ エンコード, VS-Tree作成

vID	vLabel	adjList {(eLabel, nLabel)*}
001	y:Abraham_Lincoln	(hasName, "Abraham Lincoln") (BornOnDate, "1809-02-12"), (DiedOnDate, "1865-04-15") (DiedIn, y:Washington_D.C)
	y:Washington_D.C	(hasName, "Washington D.C.") (FoundYear, "1790")
	D.C	(rdf:type, y:city)
003	y:United_States	(hasName, "United States") (hasCapital, y:Washington_D.C) (rdf:type, y:country)
004	y:Reese_Witherspoon	(hasName, "ReeseWitherspoon") (BornOnDate, "1976-03-22") (hasCapital, y:New_Orleans_Louisiana) (rdf:type, y:Actor)
005	y:New_Orleans_Louisiana	(FoundYear, "1718"), (locatedIn, y:United_States) (rdf:type, y:city)

Figure 4: Disk-based Adjacency List Table T

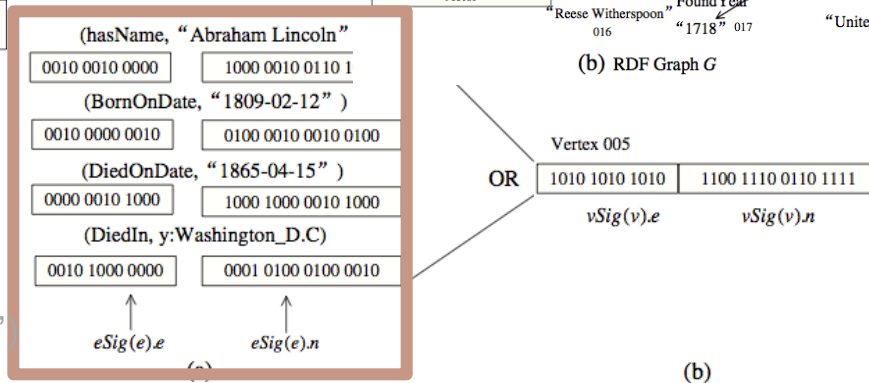
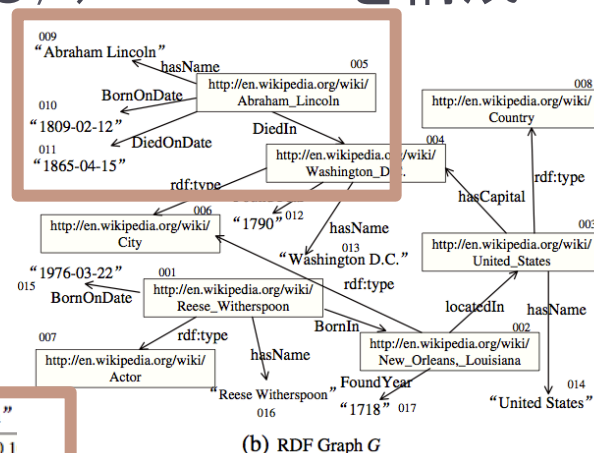


Figure 10: The Encoding Technique

ラベル(predicate)とその値(object)について、ハッシュ関数[6]を使って符号化[26]
 $(eSig(e).e)$ の, $(H_i(eLabel) \text{ MOD } M)$ 番目のビットが '1'
 $(eSig(e).n)$ の, $(H(g) \text{ MOD } N)$ 番目のビットが '1'
 (最後に論理和をとる)

Vertex Signature-Tree

- ▶ SPARQL問合せも、同様にしてシグネチャツリーに変換
- ▶ S-Tree[7]では多重結合が行えない
 - ▶ Super Edge を導入した VS-Tree を考案

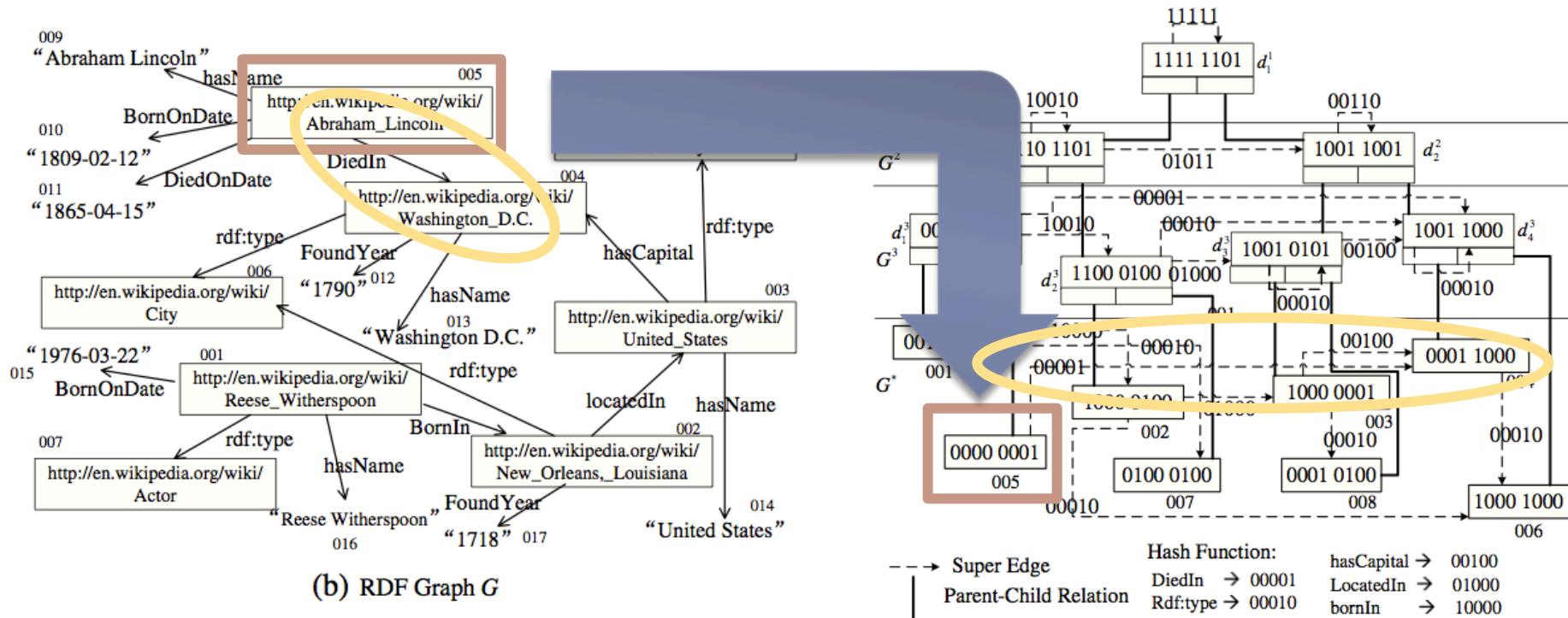


Figure 5: VS-tree

実験, 評価

▶ 枝狩り

▶ 評価

- ▶ X-RDF-3X[15]で紹介されたクエリを利用
- ▶ 正確な(exact)クエリと、ワイルドカードを含めたクエリで比較

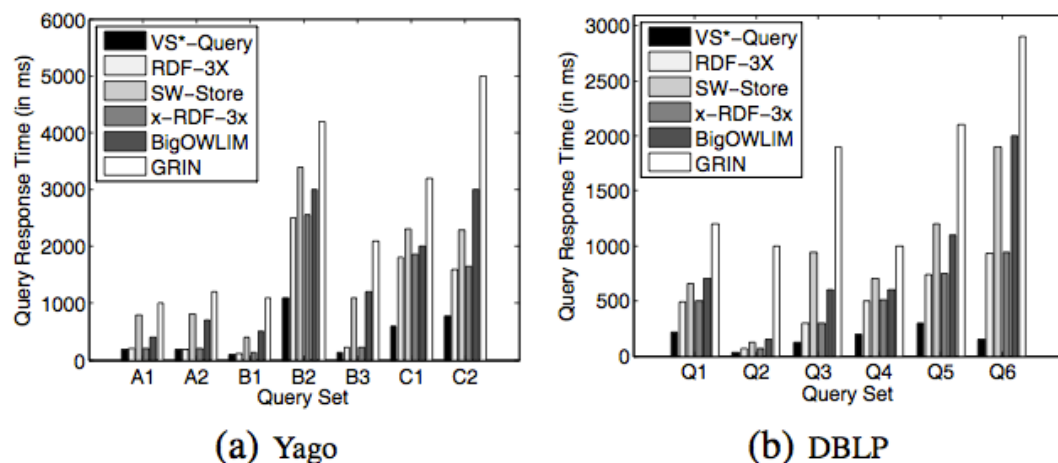


Figure 8: Exact Query Response Time

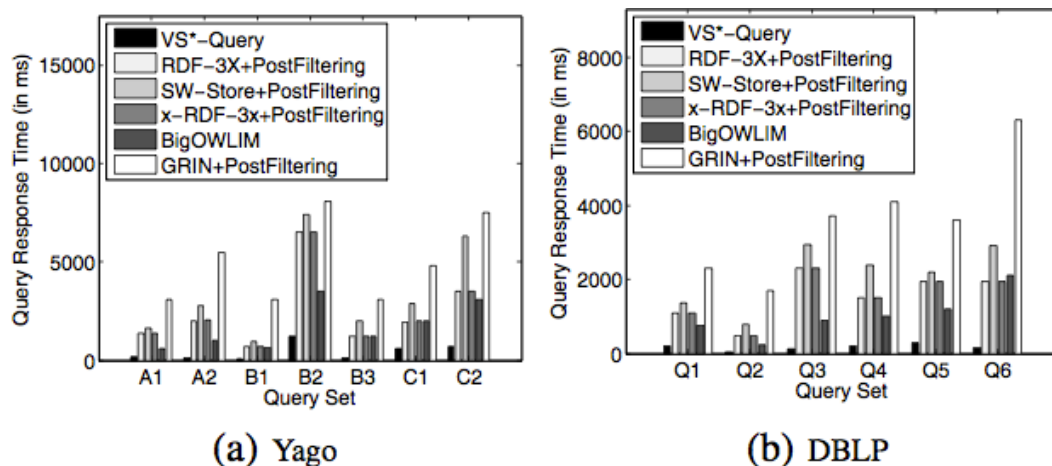


Figure 9: Wildcard Query Response Time

[2] Efficient Subgraph Search over Large Uncertain Graphs

- ▶ Ye Yuan, Guoren Wang, Haixun Wang, Lei Chen
- ▶ 概要
 - ▶ Uncertain Graph Database に対する、しきい値ベースの確率的サブグラフ問合せ
 - ▶ NP困難な問題
 - ▶ Filtering-and-verification methodology を利用

[3] Scalable SPARQL Querying of Large RDF Graphs (Jiewen Huang, Daniel Abadi, Kun Ren)

▶ 概要

- ▶ 知られている複数ノードRDFデータ管理システムよりも3桁効率的にデータを管理するシステムの紹介
 1. 最先端の単一ノードRDFストアの技術
 2. クエリ的高速化を助ける, ノードを跨いだデータのパーティショニング
 3. 分割されたデータをうまく利用した, SPARQLクエリの分割

▶ 貢献

- ▶ スケーラブルなRDFマネジメント アーキテクチャの提案
 - ▶ 単一ノードRDFストアと, Hadoopを用いた問合せの並列実行
- ▶ データ分割, 配置テクニック
 - ▶ 問合せ実行時のネットワーク通信量の劇的な削減
- ▶ 問合せクエリを並列化チャンクへ自動的に分けるアルゴリズム
- ▶ システムのパラメータを様々に変化させた実験,

システムアーキテクチャ

- ▶ RDFデータ, SPARQLクエリをクラスタに分割して実行
 - ▶ 必要に応じて, マシン間で通信を行う
 - ▶ 複数のマシンを協調させて利用するために, Hadoopでクエリ処理を管理する
 - ▶ 各WorkerがRDFストアを持つ
- ▶ データのパーティション分け
 - ▶ Hashによるパーティション分け
 - ▶ スタークエリの時のみ有効
 - ▶ 頂点(Vertex)でパーティション分け
 - ▶ `rdf:type` でリンクしているエッジを削除
 - RDFノード間の関係が複雑になる
 - => Workerを超えて通信する必要がある

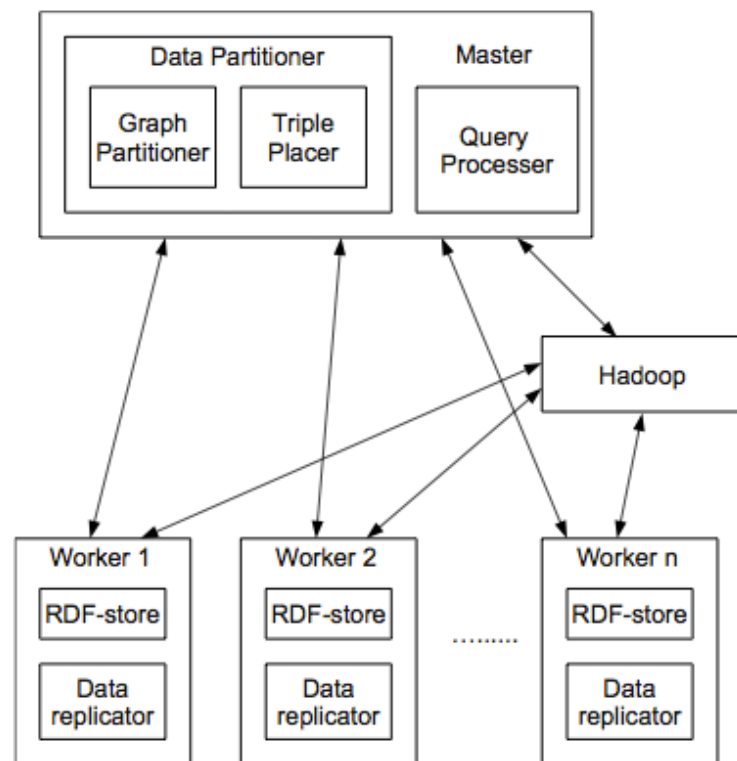


Figure 3: System Architecture.
画像は論文から引用

- ▶ 9 ▶ 別Workerへのホップ数を保証(N-hop)

実験・評価

▶ データのロード時間の比較

- ▶ 2.7 億トリプル(N-Triple形式), 50GB

▶ LUBMベンチマークの比較

- ▶ 14クエリ

System	Load Time
RDF-3X	2.5h
SHARD	6.5h
Hash Partitioning	0.5h
Graph Partitioning	Vertex Partitioning: 1h Triple Placement: 3h Loading into RDF-3Xs: 10min

Figure 5: Load Time.

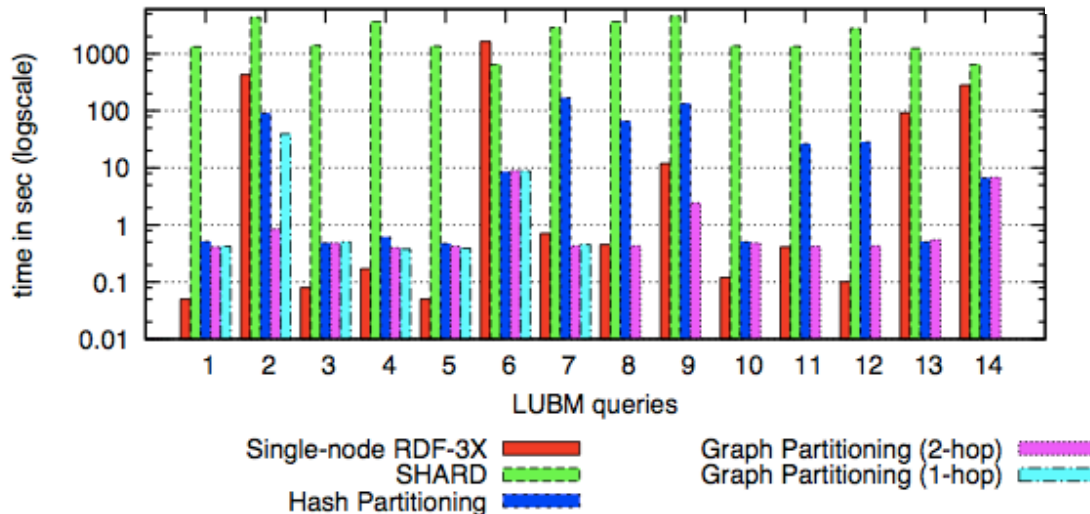


Figure 6: LUBM Execution Time.

- RDF-3X
 - 2009年に登場し, おそらく現在最速のRDFストア
- SHARD
 - Hadoop上のRDFストア