重複排除を含む SPARQL クエリの等価変換と計算量

兼岩 憲 [1]　平山 健太 [2]

セマンティック Web において，RDF (Resource Description Framework) は Web 上に存在するリソースに関するメタデータやオントロジーを記述する枠組みである．その RDF データの検索に利用されるクエリ言語として，SPARQL (SPARQL Protocol and RDF Query Language) がある．SPARQL のクエリ (SELECT 文など) において DISTINCT キーワードを用いると，検索結果から重複を排除できるが，重複排除処理の対象となるデータの件数が多いほど解決の計算量が増加する課題がある．本論文では，この問題を形式的に解決するため，DISTINCT キーワードを用いたクエリに対し，集約関数を利用した等価なクエリへの書き換え手法を提案する．そのために、集約関数と GROUP BY 句に対する SPARQL の構文および代数的意味論を拡張する．それにより，SPARQL の意味論を用いたクエリ書き換えの等価性と書き換えたクエリの計算量を証明する．

1　はじめに

従来の Web ページは単純なテキストとして扱われてきたが，セマンティック Web では Web ページのデータを機械的に解釈可能して取り扱うことができる．具体的には，セマンティック Web は Web ページに対しメタデータやオントロジー (Web ページ自身の持つ意味に関する情報) を付与することで実現される．

セマンティック Web の技術要素のひとつに，リソース間の関係を記述する RDF (Resource Description Framework) [7] がある．RDF データは，URI(Uniform Resource Identifier) によって表現されるリソースの三つ組み (RDF トリプル) の集合である．RDF トリプルの要素は順に主語，述語，目的語と呼ばれ，それをラベル付きグラフの始点，ラベル付き辺，終点に対応づけることで，RDF データ全体をラベル付きグラフとして表現できる．これを RDF グラフと呼ぶ．

RDF データに対するクエリ言語 [5] として，W3C(World Wide Web Consortium) において SPARQL [9] (SPARQL Protocol and RDF Query Language) が標準化されている．SPARQL クエリは，RDF データを蓄積し検索する RDF ストア [3] に対する問い合わせ文である．SPARQL を用いることで，RDF グラフ上のグラフパターンにマッチする部分グラフの検索や，それらの検索結果に対する重複排除（DISTINCT），集約などの操作ができる．

SPARQL クエリのうち，DISTINCT キーワードを用いたクエリの解決では，重複排除処理の際に中間結果をすべて参照して重複検査を行う必要がある．重複排除処理は中間結果の件数が多ければ多いほど計算ステップが増加するため，中間結果の件数によっては重複排除処理がクエリの解決計算量を増大させてしまう．

一方で SPARQL クエリの計算量を証明するために，SPARQL の構文と意味論 [1,8,10] が厳密に形式化されている．しかしながら，これらの構文および意味論において重複排除（DISTINCT），集約関数，GROUP BY 句が定義されておらず，それらを含んだ SPARQL クエリの計算量を示すことが難しい．

以上の問題を解決するため，本研究では以下の 3 点を実現する．

- 重複排除（DISTINCT），集約関数と GROUP BY 句に対する SPARQL の構文および代数的意味論を拡張する．
- DISTINCT キーワードによる結果の重複排除に着目した等価なクエリへの書き換え手法を提案する．
- SPARQL の意味論を用いたクエリ書き換えの等価性と書き換えたクエリの計算量を証明する．

本研究では，クエリの等価性と解決計算量を数学的に示すために，多重集合に基づく SPARQL の構文と意味論を定義して SPARQL 代数によるクエリ解釈を行う．さらに，SPARQL の DISTINCT キーワードによる結果の重複排除に対して，集約関数を用いた等価なクエリへの書き換えによるクエリ解決を試みる．その結果，SPARQL の意味論を用いてこのクエリ書き換えの等価性とクエリ変換後における解決の計算量を示す．

本論文の構成は次のとおりである．2 章では RDF データと SPARQL の構文を定義し，3 章に重複排除，集約関数，GROUP BY に関して SPARQL の意味論を拡張する．4 章にクエリ書き換えの手法について記述し，意味論に基づいたクエリ書き換えの等価性とクエリ解決の計算量を示す．5 章では，本論文のまとめと今後の課題を述べる．

2　SPARQL の構文

本章では，RDF データと SPARQL クエリの抽象構文 [4,8] を定義し，重複排除，集約関数，および GROUP BY 句に関して拡張する．

2.1　RDF データ

RDF データは Web リソース間の関係を表現したデータの集合である．RDF データの最小単位は RDF トリプルである．RDF トリプルはリソースの URI(Uniform Resource Identifier) [2] の三つ組 (s, p, o) によって表され，各要素 s, p, o は順に主語，述語，目的語と呼ばれる．図 1 は RDF データの例であり，各行が空白文字で区切られた主語，述語，目的語からなる RDF トリプルを表す．図 1 の 1 行目に示した RDF トリプルは主語<hydrogen>，述語 rdf:type，目的語<atom>から構成される．

空白ノード集合 B, リテラル集合 L, URI 集合 U に対し，RDF トリプルは $(B \cup U) \times U \times (B \cup L \cup U)$ の元である．RDF データは RDF トリプルの有限集合 $D \subseteq (B \cup U) \times U \times (B \cup L \cup U)$ から構成される．

[1] 正会員　電気通信大学大学院 情報理工学研究科 情報・ネットワーク工学専攻
kaneiwa@uec.ac.jp
[2] 非会員　電気通信大学大学院 情報理工学研究科 情報・ネットワーク工学専攻
k-hirayama@mail.uec.jp

```
<hydrogen> rdf:type <atom> .
<hydrogen> <atomic_weight> "1" .
<CH4> <contains> <hydrogen> .
<CH4> <contains> <carbon> .
<CH3COOH> <contains> <hydrogen> .
<CH3COOH> <contains> <carbon> .
<CH3COOH> <contains> <oxygen> .
...
```

図 1　RDF データの例

2.2　SPARQL クエリ

RDF データはクエリ言語 SPARQL を用いて検索される．そのとき，SPARQL クエリに条件文を付与することで制約付きの検索が可能になる．最初に，SPARQL クエリとその構成要素について定義する．トリプルパターンは RDF トリプルの成分のうち 0 個以上を変数に置き換えた表現である．

定義 2.1 (トリプルパターン)**.** URI 集合 U, リテラル集合 L, 変数集合 V を用いると, トリプルパターンは $(U\cup V)\times(U\cup V)\times(L\cup U\cup V)$ の要素である．尚，トリプルパターン内の変数を $?x, ?y$ のように表し，それ以外のものはリソース定数 ($L\cup U$ の要素) であるとする．

以下では，項および，項の制約式から表現される条件文を定義する．

定義 2.2 (項)**.** 変数集合 V，リソース集合 $B\cup L\cup U$ に対し，変数 $?x\in V$ とリソース $e\in B\cup L\cup U$ は項である．また，n 項関数 f と n 個の項 $t_1,\ldots,t_n$ に対し，$f(t_1,\ldots,t_n)$ は項である．

定義 2.3 (条件式)**.** 項 t_1, t_2，二項述語 r に対し，$r(t_1,t_2)$ は (原子) 条件式である．また，R_1, R_2 が条件式であるとき，$\neg R_1, R_1\wedge R_2, R_1\vee R_2$ も条件式である．二項述語には $<, \leq, >, \geq, =, \neq$ などがある．

本論文では，特に集約表現とその集約に関する条件式の定義を新たに追加する．集約関数 $\mathbf{f}$，項 t に対して，$\mathbf{f}(t)$ は集約表現である．SPARQL 1.1 [9] の標準仕様では，集約関数として `COUNT`, `SUM`, `MIN`, `MAX`, `AVG`, `SAMPLE`, `GROUP_CONCAT` の 7 種類が導入されている．

定義 2.4 (集約に関する条件式)**.** 集約表現 $\mathbf{a}$，項 t，二項述語 r，に対し，$r(\mathbf{a},t)$ は集約に関する (原子) 条件式である．

次に SPARQL クエリの表現を定義する．トリプルパターン tp，条件式 R，SPARQL クエリ Q_1, Q_2 に対し，tp, Q_1 And Q_2, Q_1 Union Q_2, Q_1 Filter R は SPARQL 表現である．以下では，新たに DISTINCT キーワードと GROUP BY 句を導入した SPARQL クエリを定義する．

定義 2.5 (SPARQL クエリ)**.** SPARQL 表現 q，変数 $?v_1, \ldots, ?v_k$，変数集合 $S\subseteq V$，集約に関する条件式 $\mathbf{R}$ および集約表現 $\mathbf{a}_1,\ldots,\mathbf{a}_k$ に対し，以下は SPARQL クエリである．

1. q.
2. Select$_S(q)$.
3. Select Distinct(q).
4. Select$_{\mathbf{a}_1\text{ AS}?v_1,\ldots,\mathbf{a}_k\text{ AS}?v_k}(q)$ Group by S Having $\mathbf{R}$.

ここで, S が空である場合, $\mathbf{R}$ が恒真のときそれぞれ Group by S と Having $\mathbf{R}$ を省略できる．

3　SPARQL の意味論

本章では SPARQL 代数により RDF データにおける SPARQL クエリの意味論を定義する．

3.1　代入と集約関数

SPARQL クエリに対して，トリプルパターンに含まれる変数への代入は 1 つのクエリ解決を意味する．変数集合 V からリソース集合 $B\cup L\cup U$ への部分関数 $\mu: V\to B\cup L\cup U$ を代入という．本論文では，代入 μ を部分集合 $\mu\subseteq V\times(B\cup L\cup U)$ としても扱う．代入の全体多重集合を Sub と表し，代入 μ の定義域を $dom(\mu)$ と表す．

定義 3.1 (代入の拡張)**.** 項とトリプルパターンに対し代入 μ の拡張 $\mu(E)$ を次のように定義する．ここで，$t_1,\ldots,t_n$ は項，f は n 項関数である．

1. $e\in B\cup L\cup U\Rightarrow\mu(e)=e$.
2. $\mu(f(t_1,\ldots,t_n))=f(\mu(t_1),\ldots,\mu(t_n))$.
3. $\mu((s,p,o))=(\mu(s),\mu(p),\mu(o))$.

この定義により，トリプルパターン tp に代入 μ を適用した結果を $\mu(tp)$ と表す．

定義 3.2 (条件式と充足可能関係)**.** ある代入 μ により条件式 R が真となることを $\mu\models R$ と表し，関係 $\models$ を充足可能関係と呼ぶ．任意の条件式に対し，この関係を帰納的に次のように定義する．

1. $\mu\models r(t_1,t_2)\Leftrightarrow r(\mu(t_1),\mu(t_2))$.
2. $\mu\models\neg R_1\Leftrightarrow\mu\not\models R_1$.
3. $\mu\models R_1\wedge R_2\Leftrightarrow\mu\models R_1\wedge\mu\models R_2$.
4. $\mu\models R_1\vee R_2\Leftrightarrow\mu\models R_1\vee\mu\models R_2$.

定義 3.3 (互換な代入)**.** 代入 μ_1,μ_2 に対し，すべての $?v\in dom(\mu_1)\cap dom(\mu_2)$ が $\mu_1(?v)=\mu_2(?v)$ を満たすとき，μ_1 と μ_2 が互換であるといい，$\mu_1\sim\mu_2$ と表す．

$\mu_1\sim\mu_2$ のとき，代入の結合 $\mu_1\cup\mu_2$ が定義される．代入の結合もまた代入である．特に，μ_1 と μ_2 の定義域から与えられる共通部分集合 $V\subseteq dom(\mu_1)\cap dom(\mu_2)$ について，すべての $?v\in V$ に対して $\mu_1(?v)=\mu_2(?v)$ であるとき，$\mu_1\sim_V\mu_2$ と表す．二項関係 $\sim$ および $\sim_V$ は常に同値関係である．特に，$\sim_\emptyset$ は自明な関係となる．図 2 は関係 $\sim, \sim_V$ の視覚的なイメージを表している．例えば，2 つの代入 μ_1,μ_3 の間には $\mu_1(?y)=\mu_3(?y)$ (= <2>) が成り立つので，$\mu_1\sim_{\{?y\}}\mu_3$ が成り立つ．

SPARQL クエリは RDF データ上のすべての解決 (代入) を集合として返す．例えば SPARQL クエリ Q のすべての解決が 3 つの代入 $\mu_1,\mu_2,\mu_3\in Sub$ ならば，それらの代入をまとめて集合 $\{\mu_1,\mu_2,\mu_3\}$ (代入集合と呼ぶ) で表す．本論文における代入集合

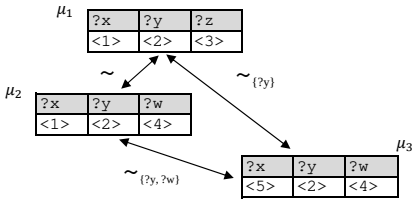

図 2 関係 $\sim, \sim_V$ のイメージ

には要素の重複を許すものとする．集合を内包的あるいは外延的に表す場合，その集合の要素に重複を許さないとき $\{\cdot\}$，それ以外のとき $[\cdot]$ でくくって表す．

集約関数 **f** は SPARQL クエリの代入集合 Ω を集約しリソース e ($\in B \cup L \cup U$) を求める関数である．以下のように，SPARQL 1.1 [9] に導入された集約関数の解釈が代入集合の適用として定義される．

定義 3.4 (集約関数への代入集合の適用)**.** 集約関数 `COUNT`, `SUM`, `MIN`, `MAX`, `AVG`, `SAMPLE`, `GROUP_CONCAT` への代入集合 Ω の適用を以下のように定義する．ここで，t は項である．また，$\bullet$ は文字列連結演算子とする．

1. $\Omega(\text{COUNT}_*) = |\Omega|$
2. $\Omega(\text{COUNT}_t) = |[\mu(t)|\mu \in \Omega]\,|$.
3. $\Omega(\text{SUM}_t) = \sum_{\mu\in\Omega} \mu(t)$.
4. $\Omega(\text{MIN}_t) = \min_{\mu\in\Omega} \mu(t)$.
5. $\Omega(\text{MAX}_t) = \max_{\mu\in\Omega} \mu(t)$.
6. $\Omega(\text{AVG}_t) = \text{SUM}_t(\Omega)/\text{COUNT}_t(\Omega)$.
7. $\Omega(\text{SAMPLE}_t) = \mu(t)$ $(\exists \mu \in \Omega)$.
8. $\Omega(\text{GROUP\_CONCAT}_t) = \bullet_{\mu\in\Omega}\, \mu(t)$.

定義 3.5 (変数とリソースへの代入集合の適用)**.** 任意の変数 $?v$，リソース e に対し，それらへの代入集合 Ω の適用 $\Omega(?v), \Omega(e)$ を次のように定義する．

1. $\Omega(?v) = \left(\bigcap_{\mu\in\Omega} \mu\right)(?v)$.
2. $\Omega(e) = e$.

代入集合の適用により，集約表現を含んだ条件式に対して以下のように充足可能関係を定義できる．

定義 3.6 (集約に関する条件式の充足可能関係)**.** 以下のように，代入集合 Ω が集約表現 **a** に関する条件式 **R** を満たすことを $\Omega \models \mathbf{R}$ と表す．

$$\Omega \models r(\mathbf{a}, t) \Leftrightarrow r(\Omega(\mathbf{a}), \Omega(t)).$$

例えば，$\Omega \models (\text{COUNT}_* = 3) \Leftrightarrow \Omega(\text{COUNT}_*) = 3 \Leftrightarrow |\Omega| = 3$ となる．

3.2 SPARQL 代数

本章では，SPARQL クエリの検索処理を SPARQL 代数として数学的に定義する．

定義 3.7 (SPARQL 代数)**.** 任意の代入集合 $\Omega, \Omega_l, \Omega_r$ に対して，結合 ($\bowtie$), 直和 ($\sqcup$), 射影 (π_S), 選択 (σ_R), 集約 ($\gamma_{S;\{(v_1,\mathbf{a}_1),\ldots,(v_k,\mathbf{a}_k)\};R}$) を次のように定義する．これらの操作を SPARQL 代数と総称する．これらの操作のうち，$\bowtie, \sqcup$ はともに結合的かつ可換な操作である．

1. $\Omega_l \bowtie \Omega_r = [\mu_l \cup \mu_r | \mu_l \in \Omega_l, \mu_r \in \Omega_r : \mu_l \sim \mu_r]$.
2. $\Omega_l \sqcup \Omega_r = [\mu | \mu \in \Omega_l \vee \mu \in \Omega_r]$.
3. $\pi_S(\Omega) = [\mu_1 | \exists \mu_2 : \mu_1 \cup \mu_2 \in \Omega \wedge dom(\mu_1) \subseteq S \wedge dom(\mu_2) \cap S = \emptyset]$.
4. $\sigma_R(\Omega) = [\mu \in \Omega | \mu \models R]$.
5. $\gamma_{S;\{(v_1,\mathbf{a}_1),\ldots,(v_k,\mathbf{a}_k)\};\mathbf{R}}(\Omega) = \{\{(v_1, \Omega'(\mathbf{a}_1)), \ldots, (v_k, \Omega'(\mathbf{a}_k))\} | \Omega' \in (\Omega/{\sim_S}), \Omega' \models \mathbf{R}\}$.

ただし，$v_1, \ldots, v_k$ は互いに相異なる変数，S は変数集合，$\mathbf{a}_1, \ldots, \mathbf{a}_k$ は集約表現である．また，R は条件式，**R** は集約に関する条件式である．

例えば，代入 $\mu_1 = \{(?x, 1), (?y, 1)\}$, $\mu_2 = \{(?x, 1), (?y, 2)\}$, $\mu_3 = \{(?x, 2), (?y, 4)\}$ からなる代入集合 $\Omega = \{\mu_1, \mu_2, \mu_3\}$ が与えられたとする．このとき，定義 3.7 の 5 によって $\gamma_{\{?x\};\{(?x,?x),(?s,\text{SUM}_{?y})\};\text{SUM}_{?y}\geq 4}$ を計算することを考える．$\mu_1 \sim_{\{?x\}} \mu_2$, $\mu_2 \not\sim_{\{?x\}} \mu_3$, $\mu_3 \not\sim_{\{?x\}} \mu_1$ だから，$\Omega/{\sim_{\{?x\}}} = \{\{\mu_1, \mu_2\}, \{\mu_3\}\}$ となる．これらから，

$$\begin{gathered}\{\mu_1, \mu_2\}(?x) = (\mu_1 \cup \mu_2)(?x) = 1,\\ \{\mu_1, \mu_2\}(\text{SUM}_{?y}) = \mu_1(?y) + \mu_2(?y) = 3,\\ \{\mu_3\}(?x) = \mu_3(?x) = 2,\\ \{\mu_3\}(\text{SUM}_{?y}) = \mu_3(?y) = 4\end{gathered}$$

が得られる．このうち，$\Omega \models \text{SUM}_{?y} \geq 4$ を満たす代入集合 Ω は $\{\mu_3\}$ だから，$\gamma_{\{?x\};\{(?x,?x),(?s,\text{SUM}_{?y})\};\text{SUM}_{?y}\geq 4}$ の値は $\{\{(?x, 2), (?s, 4)\}\}$ と計算される．

定義 3.8 (SPARQL クエリの意味論)**.** SPARQL クエリの解釈を SPARQL 代数によって次のように定義する．SPARQL クエリ Q に対し，それを RDF データ D において評価した結果を $[\![Q]\!]^+_D$ と表す．また，$[\![Q]\!]^+_D$ から重複を排除した集合を $[\![Q]\!]_D = \{\mu | \mu \in [\![Q]\!]^+_D\}$ と表す．

1. $[\![tp]\!]^+_D = \{\mu | dom(\mu) = Var(tp) \wedge \mu(tp) \in D\}$.
2. $[\![Q_1 \text{ AND } Q_2]\!]^+_D = [\![Q_1]\!]^+_D \bowtie [\![Q_2]\!]^+_D$.
3. $[\![Q_1 \text{ UNION } Q_2]\!]^+_D = [\![Q_1]\!]^+_D \sqcup [\![Q_2]\!]^+_D$.
4. $[\![Q \text{ FILTER } R]\!]^+_D = \sigma_R([\![Q]\!]^+_D)$.
5. $[\![\text{SELECT}_S(Q)]\!]^+_D = \pi_S([\![Q]\!]^+_D)$.
6. $[\![\text{SELECT DISTINCT}_S(Q)]\!]^+_D = [\![\text{SELECT}_S(Q)]\!]_D$.
7. $[\![\text{SELECT}_{\mathbf{a}_1 \text{ AS } v_1,\ldots,\mathbf{a}_k \text{ AS } v_k}(Q) \text{ GROUP BY } S \text{ HAVING } \mathbf{R}]\!]^+_D = \gamma_{S;\{(v_1,\mathbf{a}_1),\ldots,(v_k,\mathbf{a}_k)\};\mathbf{R}}([\![Q]\!]^+_D)$.

但し，誤解を招くおそれのないときは，RDF データを表す D を省略して $[\![Q]\!]^+$ のように表す．

3.3 クエリ解決の計算量

本節では，SPARQL クエリの解決にかかる時間計算量と解決結果の個数を評価する．

定理 3.1 (トリプルパターンの時間計算量と解決結果の個数)**.** トリプルパターン tp の解決にかかる時間計算量の上界は RDF デー

```
SELECT DISTINCT ?p WHERE
{
  ?x1 ?p <atom>.
  ?x2 ?p <atom>.
  ?x2 <contains> ?x3.
}
```

図 3　書き換え対象のクエリの例

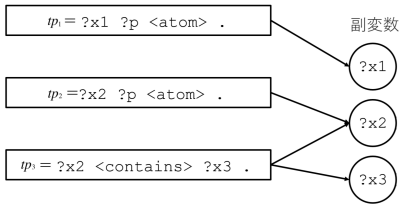

図 4　図 3 のクエリに対する依存関係グラフ

タの要素数 $|D|$ によって $O(|D|)$ で与えられる．また，解決結果の個数の上界は $|D|$ である．

証明　RDF データの要素は主語，述語，目的語の 3 つの要素からなるので，各要素に対してトリプルパターン tp にマッチするかを時間計算量 $O(1)$ で決定できる．したがって，RDF データに含まれる $|D|$ 個すべての要素に対してこの操作を行うことで，トリプルパターン tp を時間計算量 $O(|D|)$ で解決できる．D はすべてのトリプルを包含するので，解決結果の個数の上界は $|D|$ である．

定理 3.2 (AND パターンの時間計算量と解決結果の個数)**.** クエリ Q_1, Q_2 の解決をそれぞれ $[\![Q_1]\!]^+, [\![Q_2]\!]^+$ とすると，クエリ Q_1 AND Q_2 の解決にかかる時間計算量の上界は $O(|[\![Q_1]\!]^+||[\![Q_2]\!]^+|)$ で与えられる．解決結果の個数の上界は $|[\![Q_1]\!]^+||[\![Q_2]\!]^+|$ である．

証明　定義 3.7 の 1 より，すべての $(\mu_1, \mu_2) \in [\![Q_1]\!]^+ \times [\![Q_2]\!]^+$ について，$\mu_1 \sim \mu_2$ であるかを確かめることで，$[\![Q_1$ AND $Q_2]\!]^+$ を計算できる．したがって，Q_1 AND Q_2 の解決にかかる時間計算量の上界は $O(|[\![Q_1]\!]^+ \times [\![Q_2]\!]^+|) = O(|[\![Q_1]\!]^+||[\![Q_2]\!]^+|)$ で与えられる．すべての $(\mu_1, \mu_2) \in [\![Q_1]\!]^+ \times [\![Q_2]\!]^+$ について，$\mu_1 \cup \mu_2$ が検索結果に含まれうるので，解決結果の個数の上界は $|[\![Q_1]\!]^+ \times [\![Q_2]\!]^+| = |[\![Q_1]\!]^+||[\![Q_2]\!]^+|$ である．

4　SPARQL クエリ書き換え

本章では，DISTINCT キーワードを用いたクエリに対し，等価なクエリへの書き換えを行う．

4.1　クエリ書き換えの手続き

複数のトリプルパターンは BGP (Basic Graph Pattern) と呼ばれ，グラフ構造のテンプレートを表す．クエリ書き換えでは，DISTINCT キーワードが使われ，選択が 1 変数で，集約関数などを含まない BGP で構成されるクエリを対象とする．選択の対象となっている唯一の変数を**主変数**と呼び，それ以外の変数を総称して**副変数**と呼ぶ．図 3 に書き換え対象のクエリの例を示す．図 3 のクエリの主変数は`?p` である．また，副変数は`?x1`, `?x2`, `?x3` である．

BGP（n 個のトリプルパターン）からなるクエリ

$$Q_o = \text{SELECT DISTINCT}\{?v_p\}(tp_1 \text{ AND } tp_2 \text{ AND} \ldots \text{AND } tp_n).$$

の書き換えを考える．Q の変数集合を S とし，トリプルパターンの集合を $TP = \{tp_1, tp_2, \ldots, tp_n\}$ とする．また，クエリ Q の主変数を $?v_p$ とする．

副変数 $?v \in S \setminus \{?v_p\}$ について，$tp \in TP, ?v \in Var(tp)$ のとき tp から $?v$ への有向辺を作成する．このようにしてクエリ Q に対して作成される有向辺の集合を $A = \{(tp, ?v) | ?v \in S \setminus \{?v_p\}, tp \in TP, ?v \in Var(tp)\}$ として依存関係グラフ $G = (TP \cup S \setminus \{?v_p\}, A)$ を構成する．

G に含まれる各弱連結成分を $G_1, G_2, \ldots, G_k$ とする．G の i 個目の弱連結成分に含まれるトリプルパターンの集合を TP_i, 変数の集合を V_i, 有向辺の集合を A_i とする $(1 \le i \le k)$.

G に含まれるすべての弱連結成分に，主変数を含むトリプルパターンが存在している場合 (すなわち，$\forall i \in \{1, 2, \ldots, k\}, \exists tp \in TP_i, ?v_p \in Var(tp)$ を満たす場合)，G に含まれる各弱連結成分に対応するサブクエリ $Q_1, Q_2, \ldots, Q_k$ を作成することで，クエリ書き換えを適用することが可能である．ただし，作成する i 個目のサブクエリ Q_i について，$c_i = |TP_i|, TP_i = \{tp_{i1}, \ldots, tp_{ic_i}\}$ とおき，$Q_i = \text{SELECT}_{\{?p\}}(tp_{i1} \text{ AND} \ldots \text{AND } tp_{ic_i})$ とする．例えば，図 3 にに示したクエリからは図 4 に示すような依存関係グラフが構成できる．

代入 μ が k 個のサブクエリの解決の集合 $[\![Q_1]\!], [\![Q_2]\!], \ldots [\![Q_k]\!]$ のすべてに含まれるとき，かつその場合に限り，μ はもとのクエリ Q の解決結果 $[\![Q]\!]$ にも含まれる．よって，解決の集合 $[\![Q_1]\!], [\![Q_2]\!], \ldots, [\![Q_k]\!]$ の共通部分を計算するクエリを作成する．ただし，SPARQL には複数のクエリの結果の共通部分を直接計算するための構文は存在しないので，集約関数 (COUNT 関数) によって共通部分を求めるようクエリを書き換える．クエリ Q_o に対しクエリ書き換え適用後のクエリ Q_r は次のようになる．

$$\begin{aligned} Q_r = {} & \text{SELECT}_{?v_p \text{ AS} ?v_p} \\ & ((\text{SELECT DISTINCT}_{\{?v_p\}}(Q_1)) \text{ UNION} \\ & \ldots \text{UNION}(\text{SELECT DISTINCT}_{\{?v_p\}}(Q_k))) \\ & \text{GROUP BY}\{?v_p\} \text{ HAVING}(\text{COUNT}(?v_p) = k). \end{aligned} \quad (1)$$

4.2　時間計算量の削減

書き換え前のクエリ Q_o について，トリプルパターンの連言 tp_1 AND tp_2 AND $\ldots tp_n$ の解決計算量と結果の個数の上界は定理 3.1 と定理 3.2 を繰り返し適用することで $O(|D|^n), |D|^n$ で与えられる．tp_1 AND tp_2 AND $\ldots$ tp_n の解決結果をソートし，重複を排除することを考える．ソートにかかる計算量は比較ソートを用いる場合 $O(|D|^n \log(|D|^n)) = O(n|D|^n \log|D|)$，ソート後の処理にかかる計算量は $O(|D|^n)$ で与えられる．したがって，全体では $O(n|D|^n \log|D|)$ ステップで処理ができる．

一方で，書き換え後のクエリ Q_r の解決計算量について考える．サブクエリ Q_i に含まれるトリプルパターンの連言 tp_{i1}

```
SELECT ?p WHERE
{
    {  SELECT DISTINCT ?p WHERE
       { ?x1 ?p <atom>. }
  }
  UNION
  {  SELECT DISTINCT ?p WHERE
     { ?x2 ?p <atom>.
       ?x2 <contains> ?x3. }
  }
}
GROUP BY (?p) HAVING (COUNT(?p) = 2)
```

図 5　書き換え後のクエリの例

AND tp_2 AND ... tp_{ic_i} の解決時間と結果の個数は定理 3.1 と定理 3.2 を繰り返し適用することで $O(|D|^{c_i})$, $|D|^{c_i}$ で与えられる．したがって，書き換え前のクエリ Q_o の場合と同様にして，SELECT DISTINCT$_{\{?p\}}$ (Q_i) の解決時間と解決結果の個数の上界を $O(c_i|D|^{c_i}\log|D|)$, $|D|^{c_i}$ と計算できる．

故に，クエリ SELECT DISTINCT$_{\{?p\}}(Q_1)$ UNION ... UNION SELECT DISTINCT$_{\{?p\}}(Q_k)$ の解決時間と結果の個数の上界は $\max_{1\le i\le k} c_i = c$ として $O(c^2|D|^c\log|D|)$, $c|D|^c$ と計算できる．これらの計算結果について COUNT 関数による計数と HAVING 句による選択はいずれも $O(c|D|^c)$ 時間で処理できるので，全体として $O(c^2|D|^c\log|D|)$ 時間で解決できる．

ここで，c はクエリ書き換えを行った後のサブクエリに含まれるトリプルパターンの最大数だから，$c < n$ すなわち $c^2 < n^2$ が成り立つ．さらに，一般的に用いられている RDF データでは，$n \ll |D|$ より $|D|/n \gg 1$ である．これらのことから，次のことが言える．

$$\begin{aligned} T_r &= O(c^2|D|^c\log|D|) \\ &< O(n^2|D|^c\log|D|) \\ &< O(n|D|^{c+1}\log|D|) \\ &\le O(n|D|^n\log|D|) = T_o. \end{aligned}$$

以上から $T_r < T_o$ となり，解決時間の上界が小さくなることがわかる．

4.3　クエリ書き換えの等価性

本節では，4.1 節に述べた書き換えで得られたクエリ Q_r の結果が書き換え前のクエリ Q_o の結果と等しいことを示す．

定理 4.1. $[\![Q_r]\!]^+ = [\![Q_o]\!]^+$.

証明　定義 3.8 の 7 により $[\![Q_r]\!]^+$ 内の GROUP BY$\{?v_p\}$ HAVING(...) の部分が以下のように解釈される．

$$\begin{aligned} &\gamma_{\{?v_p\};\{(?v_p,\mathrm{COUNT}_{?v_p})\};\mathrm{COUNT}_{?v_p}=k} \\ &\quad [\![(\text{SELECT DISTINCT}_{\{?v_p\}}(Q_1))\ \text{UNION} \\ &\quad \ldots \text{UNION}(\text{SELECT DISTINCT}_{\{?v_p\}}(Q_k)))]\!]^+. \end{aligned}$$

$[\![\cdot]\!]^+$ 内に定義 3.8 の 3 を適用すると，

$$\begin{aligned} &\gamma_{\{?v_p\};\{(?v_p,\mathrm{COUNT}_{?v_p})\};\mathrm{COUNT}_{?v_p}=k}(\pi_{\{?v_p\}} \\ &\quad ([\![(\text{SELECT DISTINCT}_{\{?v_p\}}(Q_1))]\!]^+ \sqcup \\ &\quad \ldots \sqcup [\![(\text{SELECT DISTINCT}_{\{?v_p\}}(Q_k))]\!]^+)). \end{aligned}$$

さらに定義 3.8 の 5，4 を順に適用すると，

$$\begin{aligned} &\gamma_{\{?v_p\};\{(?v_p,\mathrm{COUNT}_{?v_p})\};\mathrm{COUNT}_{?v_p}=k}(\{\mu \mid \mu \in \\ &\quad \pi_{\{?v_p\}}[\![Q_1]\!]^+\} \sqcup \ldots \sqcup \{\mu \mid \mu \in \pi_{\{?v_p\}}[\![Q_k]\!]^+\}). \end{aligned}$$

ここで，$\Omega_i = \{\mu \mid \mu \in \pi_{\{?v_p\}}[\![Q_i]\!]^+\}$ $(1 \le i \le k)$ とおいて定義 3.7 の 5 を適用すると，

$$\begin{aligned} &\{\{(?v_p, ?v_p(\Omega'))\} \mid \Omega' \in ((\Omega_1 \sqcup \ldots \sqcup \Omega_k)/{\sim_{\{?v_p\}}}), \\ &\quad \mathrm{COUNT}_{?v_p}(\Omega')k\}. \end{aligned} \tag{2}$$

各 Ω_i $(1 \le i \le k)$ は $\{?v_p\}$ を定義域とする代入の集合だから，定義 3.3 より $\mu_1, \mu_2 \in \Omega_i$ について $\mu_1 \sim_{\{?v_p\}} \mu_2 \Leftrightarrow \mu_1 = \mu_2$. さらに，定義 3.6 より $\Omega' \models \mathrm{COUNT}_{?v_p}(\Omega') = k \Leftrightarrow |\{\mu \in \Omega' \mid ?v_p \in dom(\mu)\}| = k$ であるが，Ω' 内のすべての代入が $?v_p$ を定義域に含むから，$\Omega' \models \mathrm{COUNT}_{?v_p}(\Omega') = k \Leftrightarrow |\Omega'| = k$. したがって，$\Omega_i$ が重複を許さないこととあわせると，あらゆる代入 μ に対して，このとき $\Omega_{i,\mu} = [\mu' \mid \mu' \in \Omega_i, \mu' \sim_{\{?v_p\}} \mu]$ は高々 1 個の要素をもつ．よって，代入 μ に対して $|[\mu' \mid \mu' \in \Omega_1 \sqcup \ldots \sqcup \Omega_k, \mu' \sim_{\{?v_p\}} \mu]| = \sum_{i=1}^{k} |\Omega_{i,\mu}| = k$ であるときかつそのときに限り，$\mu \in \Omega_1 \cap \ldots \cap \Omega_k$. このことを (2) 式にあてはめると，

$$[\![Q_r]\!]^+ = \Omega_1 \cap \ldots \cap \Omega_k. \tag{3}$$

一方で $[\![Q_o]\!]^+$ に定義 3.8 の 6 を適用すると，

$$[\![\text{SELECT}_{\{?v_p\}}(tp_1\ \text{AND}\ tp_2\ \text{AND} \ldots \text{AND}\ tp_n)]\!].$$

トリプルパターン $tp_1, tp_2, \ldots, tp_n$ はそれぞれサブクエリ $Q_1, Q_2, \ldots, Q_k$ のうち 1 個以上に含まれるので，

$$\begin{aligned} &[\![\text{SELECT}_{\{?v_p\}}((tp_{11}\ \text{AND} \ldots \text{AND}\ tp_{1c_1})\ \text{AND} \\ &\quad \ldots \text{AND}(tp_{k1}\ \text{AND} \ldots \text{AND}\ tp_{kc_k}))]\!]. \end{aligned}$$

$\Omega'_i = [\![tp_{i1}\ \text{AND} \ldots \text{AND}\ tp_{ic_i}]\!]$ $(1 \le i \le k)$ とおいて定義 3.8 の 2 を適用すると，

$$[\![\text{SELECT}_{\{?v_p\}}(\Omega'_1 \bowtie \cdots \bowtie \Omega'_k)]\!].$$

互換関係 $\sim$ の推移性に注意して $(\cdot)$ 内に定義 3.7 の 1 を適用すると，

$$\begin{aligned} &[\![\text{SELECT}_{\{?v_p\}}([\mu_1 \cup \ldots \cup \mu_k \mid \\ &\quad \mu_i \in \Omega'_i\ (1 \le i \le k), \mu_i \sim \mu_j (1 \le i, j \le k)])]\!]. \end{aligned} \tag{4}$$

ここで，(4) 式中の μ_i, μ_j $(1 \le i, j \le k)$ について，依存関係グラフの作り方より，$Dom(\mu_i) \cup Dom(\mu_j) = \{?v_p\}$ だから，(4) 式は次のように変形できる．

$$\begin{aligned} &[\![\text{SELECT}_{\{?v_p\}}([\mu_1 \cup \ldots \cup \mu_k \mid \\ &\quad \mu_i \in \Omega'_i\ (1 \le i \le k), \mu_i \sim_{\{?v_p\}} \mu_j (1 \le i, j \le k)])]\!]. \end{aligned}$$

続けて定義 3.8 の 5 を適用すると，

$$\{\mu \mid \mu \in \pi_{\{?v_p\}}([\mu_1 \cup \ldots \cup \mu_k \mid \mu_i \in \Omega'_i\ (1 \leq i \leq k), \mu_i \sim_{\{?v_p\}} \mu_j (1 \leq i, j \leq k)])\}.$$

定義 3.7 の 3 を適用すると，

$$[\![Q_o]\!]^+ = \{\mu \mid \exists \mu' : \mu \cup \mu' \in [\mu_1 \cup \ldots \cup \mu_k \mid \mu_i \in \Omega'_i\ (1 \leq i \leq k), \mu_i \sim_{\{?v_p\}} \mu_j (1 \leq i, j \leq k)], dom(\mu) \subseteq \{?v_p\}, dom(\mu') \cap \{?v_p\} = \emptyset\}. \quad (5)$$

ここで，書き換えが可能なクエリの条件より，(5) 式中の $\mu_1 \cup \ldots \cup \mu_k$ について，$?v_p \in dom(\mu_1 \cup \ldots \cup \mu_k)$. したがって，

$$\{\mu \mid \exists \mu' : \mu \cup \mu' \in [\mu_1 \cup \ldots \cup \mu_k \mid \mu_i \in \Omega'_i\ (1 \leq i \leq k), \mu_i \sim_{\{?v_p\}} \mu_j (1 \leq i, j \leq k)], dom(\mu) = \{?v_p\}, dom(\mu') \cap \{?v_p\} = \emptyset\}.$$

いま，$\mu'' \in [\mu_1 \cup \ldots \cup \mu_k \mid \mu_i \in \Omega'_i\ (1 \leq i \leq k), \mu_i \sim_{\{?v_p\}} \mu_j (1 \leq i, j \leq k)]$ かつ $?v_p \in dom(\mu'')$ とする．このとき，$dom(\mu) = \{?v_p\}, dom(\mu') \cap \{?v_p\} = \emptyset$ となるような $\mu = \{(?v_p, \mu''(?v_p))\}$, $\mu' = \mu'' - \mu$ が常に存在する．したがって，

$$\{\{(?v_p, \mu''(?v_p))\} \mid \mu'' \in [\mu_1 \cup \ldots \cup \mu_k \mid \mu_i \in \Omega'_i\ (1 \leq i \leq k), \mu_i \sim_{\{?v_p\}} \mu_j (1 \leq i, j \leq k)]\}.$$

互換関係 $\sim$ の定義より，$\mu_i \sim_{\{?v_p\}} \mu_j \Leftrightarrow \{(?v_p, \mu_i(?v_p))\} = \{(?v_p, \mu_j(?v_p))\}$ だから，

$$\{\{(?v_p, \mu''(?v_p))\} \mid \mu'' \in [\mu_1 \cup \ldots \cup \mu_k \mid \mu_i \in \{\{(?v_p, \mu'''(?v_p))\} \mid \mu''' \in \Omega'_i\}\ (1 \leq i \leq k), \mu_i = \mu_j (1 \leq i, j \leq k)]\}.$$

ここで，$\mu \in \Omega'_i \Rightarrow ?v_p \in dom(\mu)$ より，$\{\{(?v_p, \mu'''(?v_p))\} \mid \mu''' \in \Omega'_i\} = \pi_{\{?v_p\}}(\Omega'_i) = \Omega_i$ である．したがって，

$$\{\{(?v_p, \mu''(?v_p))\} \mid \mu'' \in [\mu_1 \cup \ldots \cup \mu_k \mid \mu_i \in \Omega_1 \cap \ldots \cap \Omega_k\ (1 \leq i \leq k), \mu_i = \mu_j (1 \leq i, j \leq k)]\}.$$

$\mu_i = \mu_j (1 \leq i, j \leq k)$ のとき $\mu_1 \cup \ldots \cup \mu_k = \mu_1$ だから，

$$\{\{(?v_p, \mu''(?v_p))\} \mid \mu'' \in \Omega_1 \cap \ldots \cap \Omega_k\}.$$

定義 3.7 の 3 より，

$$[\![Q_o]\!]^+ = \pi_{\{?v_p\}}(\Omega_1 \cap \ldots \cap \Omega_k).$$

Ω_i の定義より，すべての $\mu \in \Omega_i$ について $dom(\mu) = \{?v_p\}$ だから，

$$[\![Q_o]\!]^+ = \Omega_1 \cap \ldots \cap \Omega_k. \quad (6)$$

(3) 式と (6) 式より，

$$[\![Q_r]\!]^+ = [\![Q_o]\!]^+.$$

5 おわりに

本研究では，重複排除 (DISTINCT キーワード) を必要とする SPARQL クエリの解決に対する時間計算量とそのクエリと同じ解決結果をもたらす SPARQL クエリへの形式的な書き換えを提案した．特に，SPARQL 代数による SPARQL クエリの意味論を拡張して，これまで形式的に考慮されていなかった重複排除，集約関数および GROPU BY 句の表現に対する解釈を形式化した．それにより，重複排除を含むクエリから集約関数と GROPU BY 句を含むクエリへ変換し，それらのクエリの等価性を意味論上で証明した．さらに，提案手法により変換された SPARQL クエリの解決に対して時間計算量が削減されることを証明した．

今後の課題として，重複排除，集約関数と GROPU BY 句の解釈を SPARQL 1.1 で導入されたプロパティパスを含んだ SPARQL クエリの意味論 [6] と組み合わせる定式化が考えられる．それにより，表現力の高い SPARQL クエリの解決に関する時間計算量やクエリ書き換えの研究がある．

謝辞

本研究の一部は，JSPS 科研費 JP18K11547 の助成を受けている．

参考文献

[1] Renzo Angles and Claudio Gutiérrez. The expressive power of SPARQL. In *Proceedings of the 7th International Semantic Web Conference ISWC 2008*, LNCS 5318, pages 114–129, 2008.
[2] Tim Berners-Lee, Larry M Masinter, and Roy T. Fielding. Uniform Resource Identifiers (URI): Generic Syntax. RFC 2396, August 1998.
[3] Olivier Curé and Guillaume Blin. *RDF Database Systems: Triples Storage and SPARQL Query Processing*. Morgan Kaufmann, 2015.
[4] Claudio Gutiérrez, Carlos A. Hurtado, Alberto O. Mendelzon, and Jorge Pérez. Foundations of semantic web databases. *J. Comput. Syst. Sci.*, 77(3):520–541, 2011.
[5] Peter Haase, Jeen Broekstra, Andreas Eberhart, and Raphael Volz. A comparison of RDF query languages. In *Proceedings of the 3rd International Semantic Web Conference ISWC 2004*, LNCS 3298, pages 502–517, 2004.
[6] Egor V. Kostylev, Juan L. Reutter, Miguel Romero, and Domagoj Vrgoc. SPARQL with property paths. In *Proceedings of the 14th International Semantic Web Conference (ISWC 2015)*, LNCS 9366, pages 3–18, 2015.
[7] Ora Lassila and Ralph R. Swick. Resource Description Framework (RDF) model and syntax specification, W3C Recommendation, 1999.
[8] Jorge P'erez, Marcelo Arenas, and Claudio Gutierrez. Semantics and complexity of SPARQL. *Lecture Notes in Computer Science 4273*, pages 30–43, 2006.
[9] Steve Harris and Andy Seaborne. SPARQL 1.1 Query Language, W3C Recommendation, 2013.
[10] Michael Schmidt, Michael Meier, and Georg Lausen. Foundations of SPARQL query optimization. In Luc Segoufin, editor, *Proceedings of the 13th International Conference on Database Theory ICDT 2010*, pages 4–33, 2010.