1N4-OS-39c-1in2

推論付きSPARQLクエリ実行負荷軽減のための制約付き多目的GAを用いたOWLオントロジー最適化

OWL Ontology Optimization using Constrained Multi-objective Genetic Algorithm for Reducing Load of Inference Enabled SPARQL Query Execution

山田 直希 *1　福田 直樹 *2
Yamada Naoki　Fukuta Naoki

*1静岡大学情報学部情報科学科
Department of Computer Science, Shizuoka University

*2静岡大学学術院情報学領域
Department of Informatics, Shizuoka University

On an inference-enabled Linked Open Data(LOD) endpoint, usually a query execution takes longer than on an LOD endpoint without inference engine due to its processing of reasoning. In this paper, for reducing query execution time on an inference-enabled LOD endpoint, we propose a preliminary idea and its implementation to employ a constrained multi-objective evolutionary approach to make modification of ontologies based on the past-processed queries and their results. We show how the approach works well on implementing an inference-enabled LOD endpoint by a cluster of SPARQL endpoints.

1. はじめに

Linked Open Data (LOD) の公開や利用の拡大が進んでおり，現在少なくとも 200 以上の LOD エンドポイントが利用可能となっている *1．LOD とはオープンなデータ同士を関係性によってつなげたものであり *2，LOD は Resource Description Framework (RDF)*3 および Web Ontology Language (OWL)*4 で記述可能である．LOD では W3C で定められた SPARQL*5 と呼ばれる標準クエリ言語を用いて LOD エンドポイントに格納された RDF データを検索することが可能である．

オントロジー記述言語 OWL[Antoniou 09] は記述論理 [Baader 09] に基づいたものであり，記述論理に基づく推論が可能である．LOD エンドポイント上で OWL オントロジーの推論を利用する場合は，Pellet[Sirin 07] などの推論器が適用可能である．LOD 検索において OWL オントロジーに基づく推論を利用することで，明示的に書かれていない関係を利用したデータ検索が可能になる [Baader 08]．データ側に全ての関係を厳密に記述しなくても推論によって明示的でない関係を導き出せるので，データの公開が簡単になることなどが期待される．

OWL オントロジーに基づく推論を利用することで上記の利点が得られるが，推論の計算量が大きいため（OWL 2 オントロジーの最悪の場合の計算複雑性は 2NEXPTIME-complete[Grau 08]），推論付きの LOD エンドポイントでの SPARQL クエリ実行は，推論エンジンが用いられていないエンドポイントでの SPARQL クエリ実行よりも推論の処理が原因で時間がかかることが多い．オントロジーに基づく推論を適用するために，最悪の場合の計算複雑性などの問題を解決するための技術も開発されてきた [Baumgartner 96][Motik 09][Romero 12][Kang 12] [Gonçalves 12]．推論にかかる計算負荷をオントロジーの構造などから推定することも容易ではなく [Kang 12]，推論のトレースを取ること自体にも特別な手法が必要となる場合がある [Kazakov 14].

推論を用いた LOD 検索における推論処理負荷に関しては，推論付き SPARQL クエリにおける推論処理負荷の軽減や回避を行うために，実行時間のかかる推論付き SPARQL クエリを判別し，遺伝的アルゴリズムを用いて高速化を行う試み [Yamagata 15] がなされている．文献 [Yamagata 15] では，特殊な中継サーバを用いて推論負荷の大きいクエリの負荷を軽減する仕組みが提案されており，書き換え手法をオントロジーに拡張するする手法も提案されている [Yamada 16b].

本研究では，制約付き多目的 GA を用いた OWL オントロジーの最適化について述べる．

2. 背景

2.1 遺伝的アルゴリズを利用したクエリ書き換えルール生成

SPARQL クエリを利用した LOD 検索において，クライアント側が SPARQL クエリを用意して，SPARQL エンドポイントに直接クエリを送る．クライアントが用意した SPARQL クエリの実行に時間がかかる場合でも，標準的な SPARQL エンドポイントの実装では多くのコストをかけてその SPARQL クエリを実行しようとするので，SPARQL エンドポイントが実行に負荷のかかる SPARQL クエリを一度に大量に処理することになった場合，SPARQL エンドポイントがサーバーダウンを起こすことがある．SPARQL エンドポイントが OWL オントロジー推論に対応した推論器を含んだ SPARQL エンドポイントにおいては，こういった SPARQL エンドポイントにかかる負荷の問題は特に重要である [Yamagata 15].

推論負荷の重い SPARQL クエリの実行に対して SPARQL クエリを書き換えることで対処する試みが行われている [Yamagata 15]．SPARQL クエリに対して「SPARQL クエリ内のある条件を満たすクラスをそのクラスのサブクラスのうちの 1 つに置き換える」,「SPARQL クエリ内のある条件を満たすクラスをそのクラスの親クラスのうちの 1 つに置き換える」や「UNION 演算子の左右を入れ替える」などといったヒューリスティックな手続きで構成される書き換えルールを遺伝子構

連絡先: 山田　直希，静岡大学情報学部情報科学科，〒 432-8011　静岡県浜松市中区城北 3-5-1，E-mail: cs13098@s.inf.shizuoka.ac.jp

∗1 http://sparqles.ai.wu.ac.at/availability
∗2 http://www.w3.org/DesignIssues/LinkedData.html
∗3 https://www.w3.org/TR/rdf-sparql-query/
∗4 https://www.w3.org/TR/owl2-overview/
∗5 https://www.w3.org/TR/rdf-sparql-query/

造として遺伝的アルゴリズムを適用している．

推論付き LOD エンドポイントにおいて実行する SPARQL クエリをユーザとエンドポイントの間で仲介して，推論付き SPARQL クエリの実行に重い負荷がかかると判断された場合は，遺伝的アルゴリズムによる書き換え規則を適用して，負荷を軽くしてから推論付き LOD エンドポイントで実際に実行するといったことをサポートする仲介機構が提案されている [Yamagata 15]．この仲介機構はフロントエンドエンドポイントと呼ばれていて，フロントエンド EP はユーザとエンドポイントの中間に配置され，ユーザーから送られてきた SPARQL クエリを LOD エンドポイントで実行する前に仲介し，その推論付き SPARQL クエリの実行が負荷のかかるものかどうかということを weka[Hall 09] を使って作成した分類器を用いて分類する．もし，その推論付き SPARQL クエリの実行に時間がかかると判断された場合は，遺伝的アルゴリズムに基づく SPARQL クエリ書き換え規則を適用して負荷の軽いものにしてから実際に実行する [Yamagata 14]．また，推論付き SPARQL クエリの実行負荷が大きいために SPARQL クエリを書き換えた場合や，SPARQL クエリの書き換えだけでは対処できず，実行自体を拒否した場合は，その情報をユーザーに伝えることができる仕組みになっている [Yamagata 14]．

2.2 エンドポイントのクラスタ化

大幅な SPARQL クエリの書き換えや OWL オントロジーの改変を行った際に，推論の処理速度を高速化できたとしても，推論付き SPARQL クエリ実行結果の質を著しく損なう可能性がある．できるだけ推論付き SPARQL クエリ実行結果の質を維持するために，異なる SPARQL クエリの書き換え規則を適用できる推論付き LOD エンドポイントや異なる OWL オントロジー改変を適用した推論付き LOD エンドポイントを複数用意して，それらをクラスタ化して 1 つのエンドポイントとする [Yamagata 15]．クラスタ化した LOD エンドポイントでは，実行結果の集約に「最も返答が早かった結果を実行結果とする」や,「最も早く結果が返ってきた n 個の結果の和集合をとる」や,「最も早く結果が返ってきた n 個の結果の積集合をとる」というようなヒューリスティックな手法を用いる [Yamagata 15]．

3. 遺伝的アルゴリズムを用いたオントロジーの改変

OWL オントロジーを推論付きエンドポイント上で利用することを考慮して，遺伝的アルゴリズムを OWL オントロジーに対して適用する際に，推論付き SPARQL クエリの実行結果を目的関数として利用する場合，適応度の計算のために何度も負荷の重い推論付き SPARQL クエリを実行したり，他の推論付き SPARQL クエリ実行結果のキャッシュの影響を受けないためにエンドポイントを SPARQL クエリを実行するごとに立ち上げなおすことは時間的に大きなコストがかかる．遺伝的アルゴリズムの適用において，個体数や世代数に比例して，エンドポイントの起動回数や SPARQL クエリの実行回数が増えるので，少ない個体数と世代数で推論付き SPARQL クエリ実行負荷軽減のための OWL オントロジー改変方法の最適解を見つけることは，時間的なコストを削減することにつながる．

OWL オントロジーの改変に遺伝的アルゴリズムを適用する簡単な方法として，OWL オントロジー自体を遺伝子として表現する方法が考えられる．例えば，ある公理に対応する値が 0 の場合はその公理を OWL オントロジー内から削除し，1 の場合は何もせずそのままにすると定義すると，OWL オントロジーに対する改変を 0 と 1 で表現することができる．このように遺伝的アルゴリズムを OWL オントロジーに対して適用した場合，遺伝的アルゴリズムを適応した特定のオントロジーに対する改変方法を計算することはできるが，その改変方法を他の OWL オントロジーに対して適用してその効果を得ることは難しい．

本研究の遺伝的アルゴリズムでは，OWL オントロジーに対する改変ルールを遺伝子構造として表現する．OWL オントロジーに対する操作としては，特定のオントロジーに依存するものではなく，不特定多数のオントロジーに対して適用できるものとする．具体的には，ある条件を満たしているクラスやプロパティに対して，そのクラスやプロパティが関連している記述の中の，ある条件を満たす公理に対して操作を行う．遺伝的アルゴリズムに制約処理や多目的最適化を取り入れることで，簡単な遺伝的アルゴリズムを適用した結果よりも早い世代で同等な結果を見つけることや，同じ世代数でより優れた解を見つけることで時間的コストの削減を試みる．

3.1 多目的最適化

多目的最適化では，トレードオフとなる複数の目的に対して解 A が n 個の目的関数に対して得られる評価値を n 次元のベクトル $\bar{a}$ として表現し，優越関係を示すために $\succeq$ 記号を用いると，$A \succeq B$ を数式 (1) のように定義できる [Eiben 15]．多目的最適化ではこのように優越関係を計算し，どの個体にも優越されない非劣解に対する選択圧を強めるなどして非劣解の集合を求めることを目的とする [Eiben 15]．

$$\begin{aligned} A \succeq B \Leftrightarrow & a_i \geq b_i (\forall i \in \{1, \cdots, n\}) \text{ かつ} \\ & a_i > b_i (\exists i \in \{1, \cdots, n\}) \end{aligned} \tag{1}$$

3.2 制約処理

遺伝的アルゴリズムにおける制約処理には直接的なアプローチと間接的なアプローチがある [Eiben 15]．間接的なアプローチとしては，侵害された制約の数に比例して適応度が減少するように設計したペナルティ関数を用いて目的関数を最適化するというものがある [Eiben 15]．直接的なアプローチとしては，交叉などの遺伝的アルゴリズムでの遺伝子に対する操作を制約を侵害しないように設計することや，制約を侵害する個体を可能な限り元の個体と変わりないように制約を侵害しない個体に変える方法などがある [Eiben 15]．

3.3 制約付き多目的遺伝的アルゴリズム

本研究で適用する遺伝的アルゴリズムでは，OWL オントロジーに対する改変ルールを遺伝子構造として表現する．OWL オントロジーに対する操作としては，一度計算した OWL オントロジー改変方法を他の OWL オントロジーにも適用できるようにするため，特定のオントロジーに依存するものではなく，不特定多数のオントロジーに対して適用できるようにしている．具体的には,「対象となる OWL オントロジー内の subClassOf の関係を平均より多く持つクラスの中の someValuesFrom の制約を用いた subClassOf の公理を削除する」というような，特定のオントロジーに対して依存しない改変規則を生成するものを扱う．

本研究では，制約処理と多目的最適化を遺伝的アルゴリズムに組み込むことで，簡単な遺伝的アルゴリズムを適用した結果よりも早い世代で同等な結果を見つけることや，同じ世代数でより優れた解を見つけることで時間的コストの削減を試みる．多目的最適化として，推論付き SPARQL クエリひとつひとつ

に対する評価関数を用意するという方法が考えられる．それらの評価関数から得られる評価値を用いて個体間の優越関係を計算し，どの個体にも優越されない非劣解に対する選択圧を強めることで，個体の多様性を保てるようにする．制約処理としては文献 [Eiben 15] における直接的なアプローチを用いて，最適化を行う OWL オントロジー内の特定の記述に対して指定された操作を行えないような制約をつける．制約のつけられた記述に対して設定された操作が行われるような遺伝子が生まれた場合は，制約を侵害しないように遺伝子を強制的に書き換えるような操作を行う．

4. 議論

本提案手法の潜在的な性能向上度を明らかにするために，本アプローチに基づく手法を具体的なデータに適用し，その限界を明らかにすることを試みる．ここでの適合度の計算に用いるエンドポイントの構築には Joseki(v3.4.4)，TDB(v0.8.10)，Pellet(v2.3.0) を用いており，Pellet による推論をサポートした SPARQL クエリを受け付けることができるようにしている．

4.1 使用するデータセット

データセットとして，Ontology Alignment Evaluation Initiative (OAEI) 2013*6 の Conference track で用いられた，Linklings オントロジーを一部分解し，各クラスにインスタンスを 10 個ずつ追加したオントロジーを用いる．使用する SPARQL クエリとしては以下のような，推論を用いてサブクラスのインスタンスを取得してくるような単純なものを使用する．

```
SELECT ?x WHERE {
?x rdfs:type <http://linklings#Content>
}
```

Listing 1: クエリ A

```
SELECT ?x WHERE {
?x rdfs:type <http://linklings#Event>
}
```

Listing 2: クエリ B

```
SELECT ?x WHERE {
?x rdfs:type <http://linklings#Person>
}
```

Listing 3: クエリ C

```
SELECT ?x WHERE {
?x rdfs:type <http://linklings#Role>
}
```

Listing 4: クエリ D

```
SELECT ?x WHERE {
?x rdfs:type <http://linklings#Settings>
}
```

Listing 5: クエリ E

*6 http://oaei.ontologymatching.org/2013/conference/index.html

```
SELECT ?x WHERE {
?x rdfs:type <http://linklings#Settings>
}
```

Listing 6: クエリ F

```
SELECT ?x WHERE {
?x rdfs:type <http://linklings#SubmissionType>
}
```

Listing 7: クエリ G

4.2 遺伝的アルゴリズム適用時の性能限界の計測

ここでは，より汎用性の高い遺伝子定義を持った遺伝的アルゴリズムを適用する前に，より単純な遺伝子に基づく手法を適用することで，その性能の限界値を考察する．ここでの遺伝子の定義方法は，ある公理に対応する値が 0 の場合はその公理を OWL オントロジー内から削除し，1 の場合は何もせずそのままにすると定義するものである．この OWL オントロジーに対する改変を 0 と 1 で表現したものを遺伝子として遺伝的アルゴリズムを適用した場合の最良改変方法を適用した結果を表 1 に示す．交叉は一様交叉を用い，個体数は 20，世代数は 50，Listing1 から Listing7 のクエリを使用し，個体の適合度はそれぞれのクエリ実行結果に対して式 (2) を計算した値の平均値を用いる．F_q は OWL オントロジーの改変前と改変後の SPARQL クエリ実行結果を比較した F 値を表す．T_Q は OWL オントロジー改変前の SPARQL クエリ実行時間を表し，T_q はは OWL オントロジー改変後の SPARQL クエリ実行時間を表す．今回の実験では $\alpha = 0.7$，$\beta = 0.3$ とする．SPARQL クエリによっては改変前と同じ実行結果を維持しつつ，推論付き SPARQL クエリの実行速度を削減できているものもあるが，推論付き SPARQL クエリ実行結果の再現性を大きく損なうものもある．遺伝的アルゴリズムの結果として算出された最終世代の最も優れた個体は同世代の他の個体に対して優越関係にあり，最終世代の個体を用いてエンドポイントをクラスタリングしたとしてもクエリ実行結果を改善することができないものであった．今回は，限られた種類のクエリを扱っているため必ずしも断定的なことはいえないが，複数固体に基づくクラスタリング機構を用意しなくともクエリの実行性能を改善することが可能な高品質なオントロジー書き換え規則を生成することが可能な場合があることが考えられる．

$$V_q = \alpha F_q + \beta \frac{T_Q - Tq}{T_Q} \text{ かつ } \alpha + \beta = 1 \qquad (2)$$

表 1: 遺伝的アルゴリズムから生成した最良改変方法の適用例

	改変前の実行速度	改変後の実行速度	Precision	Recall	F-measure
クエリ A	4261ms	596ms	1.0	1.0	1.0
クエリ B	5242ms	591ms	1.0	0.67	0.80
クエリ C	6823ms	606ms	1.0	0.60	0.75
クエリ D	7068ms	589ms	1.0	0.88	0.93
クエリ E	5007ms	580ms	1.0	1.0	1.0
クエリ F	5272ms	578ms	1.0	1.0	1.0
クエリ G	4513ms	613ms	1.0	1.0	1.0

5. おわりに

本研究では，制約付き多目的 GA を用いた OWL オントロジーの最適化について述べた．本研究では，最適化の指標とし

て推論付き SPARQL クエリの実行結果を利用しており，OWL オントロジーを推論付き LOD エンドポイントで用いた場合の推論付き SPARQL クエリ実行速度や，実効結果の質を GA における評価関数として用いた．GA における個体の評価は，複数の SPARQL クエリの実効結果を用いて行い，それぞれの SPARQL クエリの実効結果に対する評価値を 1 つの目的としてとらえ，それぞれの SPARQL クエリ実効結果に対して多目的最適化をはかった．また，個体の多様性を保持することで，GA の適用結果として得られるオントロジーをそれぞれ推論付き SPARQL エンドポイントにおいて利用し，それらをクラスタ化して利用した場合，通常の GA を適用した場合に比べて優れた結果が得られることが期待される．

今後の課題としては，GA を適用することで得られた具体的な改変方法を一般化し，特定の OWL オントロジーに対してではなく，複数の OWL オントロジーに対して共通して適用できるような改変方法を導くこと [山田 17] や，より多くの複雑な推論付き SPARQL クエリに対して本研究のアルゴリズムを適用することを考えている．推論付き SPARQL クエリ実行時間を削減できるように OWL オントロジーの改変を支援するシステムの試作を行っているので，それに対して本研究の遺伝的アルゴリズムを組み込むことも今後の課題である [Yamada 16a].

6. 謝辞

本研究の一部は，JST CREST の支援を受けたものである．

参考文献

[Antoniou 09] Antoniou, G. and Van Harmelen, F.: Web Ontology Language: OWL, in Staab, S. and Studer, R. eds., *Handbook on ontologies*, pp. 91–110, Springer (2009)

[Baader 08] Baader, F. and Suntisrivaraporn, B.: Debugging SNOMED CT Using Axiom Pinpointing in the Description Logic $\mathcal{EL}^+$, in *Proceedings of the 3rd International Conference on Knowledge Representation in Medicine KR-MED 2008*, Vol. 410, pp. 1–7, CEUR-WS (2008)

[Baader 09] Baader, F., Horrocks, I., and Sattler, U.: Description Logics, in *Handbook on Ontologies*, pp. 21–43, Springer (2009)

[Baumgartner 96] Baumgartner, P., Furbach, U., and Niemelä, I.: Hyper Tableaux, in Alferes, J. J., Pereira, L. M., and Orlowska, E. eds., *Logics in Artificial Intelligence. LNCS*, Vol. 1126, pp. 1–17, Springer-Verlag (1996)

[Eiben 15] Eiben, A. E. and Smith, J. E.: *Introduction to Evolutionary Computing*, pp. 195–213, Springer Publishing Company, Incorporated, 2nd edition (2015)

[Gonçalves 12] Gonçalves, R. S., Parsia, B., and Sattler, U.: Performance Heterogeneity and Approximate Reasoning in Description Logic Ontologies, in Cudré-Mauroux, P., et al. eds., *The Semantic Web–ISWC 2012 Part I. LNCS*, Vol. 7649, pp. 82–98, Springer-Verlag (2012)

[Grau 08] Grau, B. C., Horrocks, I., Motik, B., Parsia, B., Patel-Schneider, P., and Sattler, U.: OWL 2: The next step for OWL, *Web Semantics: Science, Services and Agents on the World Wide Web*, Vol. 6, No. 4, pp. 309–322 (2008)

[Hall 09] Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., and Witten, I. H.: The WEKA Data Mining Software: An Update, *ACM SIGKDD explorations newsletter*, Vol. 11, No. 1, pp. 10–18 (2009)

[Kang 12] Kang, Y.-B., Li, Y.-F., and Krishnaswamy, S.: Predicting Reasoning Performance Using Ontology Metrics, in Cudré-Mauroux, P., et al. eds., *The Semantic Web–ISWC 2012 Part I. LNCS*, Vol. 7649, pp. 198–214, Springer-Verlag (2012)

[Kazakov 14] Kazakov, Y. and Klinov, P.: Goal-Directed Tracing of Inferences in EL Ontologies, in Mika, P., et al. eds., *The Semantic Web–ISWC 2014 Part II. LNCS*, Vol. 8797, pp. 196–211, Springer-Verlag (2014)

[Motik 09] Motik, B., Shearer, R., and Horrocks, I.: Hypertableau Reasoning for Description Logics, *Journal of Artificial Intelligence Research*, Vol. 36, pp. 165–228 (2009)

[Romero 12] Romero, A. A., Grau, B. C., and Horrocks, I.: MORe: Modular Combination of OWL Reasoners for Ontology Classification, in Cudré-Mauroux, P., et al. eds., *The Semantic Web–ISWC 2012 Part I. LNCS*, Vol. 7649, pp. 1–16, Springer (2012)

[Sirin 07] Sirin, E., Parsia, B., Grau, B. C., Kalyanpur, A., and Katz, Y.: Pellet: A practical OWL-DL reasoner, *Journal of Web Semantics*, Vol. 5, No. 2, pp. 51 – 53 (2007)

[Yamada 16a] Yamada, N. and Fukuta, N.: Toward Performance-Oriented Ontology Debugging Support Using Heuristic Approaches and DL Reasoning, in *1st International Workshop on Platforms and Applications for Social problem Solving and Collective Reasoning (PASSCR 2016)*, pp. 88–91 (2016)

[Yamada 16b] Yamada, N., Yamagata, Y., and Fukuta, N.: Query Rewriting or Ontology Modification? Considering Reasoning Capability on LOD Endpoints, in *IEEE International Conference on Agents (IEEE ICA 2016)*, pp. 19–24 (2016)

[Yamagata 14] Yamagata, Y. and Fukuta, N.: A Dynamic Query Optimization on a SPARQL Endpoint by Approximate Inference Processing , in *Proc. of 5th International Conference on E-Service and Knowledge Management (ESKM 2014)*, pp. 161–166 (2014)

[Yamagata 15] Yamagata, Y. and Fukuta, N.: An Approach to Dynamic Query Classification and Approximation on an Inference-enabled SPARQL Endpoint, *Journal of Information Processing*, Vol. 23, No. 6, pp. 759–766 (2015)

[山田 17] 山田 直希, 福田直樹：ホーン節に基づいた論理による OWL オントロジー修正機構の試作, 第 41 回人工知能学会セマンティックウェブとオントロジー (SWO) 研究会 (2017)