Distributed scalable abstract reasoning based on dl-lite

Bin Xia¹

¹ 1st School of Computer Science and Technology Wuhan University of Science and Technology Wuhan, China

Abstract. In recent years, researchs have shown that the ontology of a large number of data sets can be inferred effectively by abstract techniques. Distributed environment can further improve the performance of abstract reasoning. Firstly, abstract technology refers to a technology that divides ontology into equivalence classes and uses reasoning tools to reason abstract ontology. Then, The newly generated entities and relationships are added to the original ontology data set through materialization techniques. Finally, in order to ensure the consistency of ontology, the consistency check of ontology is needed later. We use the benchmark to obtain preliminary experimental results. The final results show that this abstract method can deal with a large number of ontologies effectively. Compared with the original reasoning method, the abstract technique proposed in this paper can effectively improve the reasoning efficiency and reduce the memory consumption time.

1. Introduction

With the development of ontology, ontology reasoning has become a hot topic. With the development of research, it is possible to deduce a large number of ontology sets. Some existing reasoning tools such as konclude¹, hermit can deal with a large number of ontologies, but the reasoning efficiency is not very good. This paper introduces an ontology abstract materialization technique called ron-s². The ron-s reasoning’s rules include some steps. First, abstracting the original Abox. Then, it divides the same type of data into equivalent classes to achieve the effect of compression (the abstract data may not be smaller than the original data). Finally the ron-s writes the newly generated data back to Abox. Abstract inference technology can greatly reduce the volume of ontologies that need to be inferred. The ron-s technology can also optimize the reasoning efficiency of existing reasoning tools³. In the distributed case, the performance of abstract refinement can be further optimized and the fault tolerance of the system can be increased. In this paper, an ontology reasoning optimization method is proposed. The ontology format required for reasoning needs to conform to dl-lite description logic. The whole system runs in a multi-node distributed environment⁴. The experimental results show that the efficiency of the abstract reasoning method proposed in this paper is obviously better than that of reasoning only by means of tools.

2. Abstract Refinement

2.1. Abstract process

In the concept of ontology language, the data of ontology is divided into Abox and Tbox⁵. Tbox is a summary of the concept, and Abox is a concrete implementation of Tbox. For the convenience of understanding and research, Abox and Tbox files were clearly separated in this experiment. ALCHOI ontology is a member of ontology family and is a special ontology⁶. In previous studies, some articles have implemented abstract materialization technology based on ALCHOI ontology. ALCHOI
ontology is composed of infinite disjoint countable concept names, noun names, relationship names and individual names\[7\]. Concept for each assertion D (a), and each principle C ⊆ D, C and D should meet the following syntax definition:

\[
C ::= \bot | A | O | C1 \cap C2 | C1 \cup C2 | \exists R.C
\]

\[
D ::= \bot | A | O | D1 \cap D2 | \exists R.D | \forall R.D | \neg C.
\]

We call the abstract result of Abox abstract B. The abstract process is the division of similar concepts and relationships in Abox into a broad category such as B1,B2 and so on. The result of abstracting B is that B=B1∪B2∪... . We use reasoning tools to reason the abstracted data\[8\], materialize the ontology to get new concepts and write them back to Abox. The above process may not have inferred all the data, so we need to refine the ontology. The process is to continue calculating the type A and then generate B inference until no new assertion is generated. In general, volume of the abstract Abox is less than original Abox, especially when the number of individual types with large amounts of data is highly repetitive.

2.2. Refinement process
1. First, we get the corresponding abstract B based on input of Abox
2. Then, we use reasoning tools to materialize and abstract B based on the materialization formula B∪T
3. Adding the new assertion inferred from the previous step to Abox and recalculate the new abstract B
4. Repeat steps 2 and 3 until no new assertion is made. Then the reasoning ends

The specific process of materialization is as follows: if B is the abstraction of A, there is A corresponding Tbox T for each relational triple, and for each individual A and each concept M, there is the following relationship:

1. B∪T:=M(xtp) can be concluded that A∪T:=M(A);
2. B∪T:=M(ytp) and R(a, b) in A can be concluded that A∪T:=M(B);
3. B∪T:=M(ztp) and R(c,a) in A can be concluded that A∪T:=M(c)

The process of abstract refinement\[9\] of reasoning is mainly abox in the reasoning ontology and Tbox is ignored. We classify Abox as an equivalent class by type to get an abstract B. Then reason the abstract B. And then we can write the individual mapping back to Abox using the above formula. Iterate until no new assertion is made.

3. Remove the refinement
We have noticed that the above method may be more troublesome in reasoning. Abstract refinement techniques may require many cycles to complete. In real life, the general abstraction refinement process is within 10 cycles. We removed the detailed steps in further research\[10\]. Ontological reasoning only requires abstract materialization. Since the new technology reduces the reasoning time, the process becomes simpler and the consistency of the reasoning ontology is guaranteed. The idea of new abstraction techniques is suitable for a large number of ontologies\[11\]. Abstract techniques are used to reduce the volume of ontologies that need to be computed. In general, the more ontologies there are, the better the abstract effect will be. Of course, the ontology still needs to satisfy the dl-lite logical description. Among the various logical descriptions, dl-lite is part of a family of languages designed specifically for ontologies\[12\]. Dl-lite follows the basic logical description rules shown in fig 1.

3.1. Some Definitions
Definition 1: The abstract B is a union of the concepts and relationships of each type in Abox. Defining Abox triple relationship tp={A,R,R} where the first R represents the left-to-right relationship and the second R represents the right-to-left relationship\[13\].
Definition 2: The abox and its abstract B are type mappings. Abstract B can be type-mapped to A type and may correspond to multiple instances in Abox.

| Roles:               | Syntax | Semantics             |
|----------------------|--------|-----------------------|
| atomic role          | 𝑅      | 𝑅′ ⊆ Δ × Δ′           |
| inverse role         | 𝑅−    | {⟨e,d⟩ | ⟨d,e⟩ ∈ R′}       |

| Concepts:            | Syntax | Semantics             |
|----------------------|--------|-----------------------|
| atomic concept       | 𝑨      | 𝑨′ ⊆ Δ′               |
| nominal              | 𝑪      | 𝑪′ ⊆ Δ′, ||𝐶′|| = 1   |
| top                  | 𝑇      | ∅                     |
| bottom               | ⊥      | Δ                     |
| negation             | ¬𝐶     | Δ′ \ C′                |
| conjunction          | 𝐶 ∧ 𝐷  | 𝐶′ ∧ 𝐷′               |
| disjunction          | 𝐶 ∨ 𝐷  | 𝐶′ ∨ 𝐷′               |
| existential restriction | ∃ 𝑅   | {⟨d | ∃e ∈ Δ′ : ⟨d,e⟩ ∈ R′} |

| Axioms:              | Syntax | Semantics             |
|----------------------|--------|-----------------------|
| concept inclusion    | 𝐶 ⊆ 𝐷 | 𝐶′ ⊆ Δ′               |
| role inclusion       | 𝑅 ⊆ 𝑆 | 𝑅′ ⊆ Δ′               |
| concept assertion    | 𝐴(𝑎)   | 𝑎′ ∈ Δ′               |
| role assertion       | 𝑅(𝑎,𝑏) | ⟨𝑎′,𝑏′⟩ ∈ R′          |

Figure 1. Dl-lite syntax

3.2. Abstract Materialization

For an ontology O = A ∪ T

1: Calculating the abstract B of A.
2: Reasoning B based on formula B ∪ T.
3: Mapping the abstract B to Abox. If the corresponding data does not exist in Abox, add the data to Abox.
4: It returns ontology O at last. Finally the reasoning ends.

The ontological reasoning is not over. Because ontology inconsistency may occur in the inferred data. The new Abox still needs to check the ontology consistency.

Ontology consistency check steps:

1: O = A ∪ T
2: Checking whether ontology O meets the consistency requirement. If it meets the requirement, the program returns true directly. Otherwise it returns false and continues to check.
3: Checking whether Abox's abstract B satisfies the consistency requirement. If it satisfies the requirement, the program returns true. If it does not satisfy the requirement, the program return false directly.

In this experiment, the benchmarks such as LUBM and UOBM are used. Experimental tests are carried out by using one, five and ten data formats in LUBM and UOBM. Then, it checks whether the experimental results of ron-s algorithm reasoning and hermit reasoning tool alone are consistent. Finally, experimental results show that the volume of abstract Abox is exponentially lower than the volume of original Abox. In addition, the computational time complexity is significantly reduced. The ron-s algorithm can significantly improve the efficiency of reasoning tools and ensure the consistency of ontology.

14] [15]
4. distribute and expriment

4.1. Distributed
To further enhance the ability of reasoning tools to handle a large number of ontologies, we deployed ron-s algorithm to a distributed environment[16]. The ron-s is deployed in multiple nodes on different servers. In addition, we used registry and gateway reverse proxy technology. When a large amount of ontology data is input, the registry can intelligently select an optimized node for processing reasoning. This experiment is built with hermit inference (other inference engines can be built). In addition, jena is used to parse RDF, openAPI processes ontology files and ontology benchmark library is used as input. The results show that the performance of ron-s algorithm is significantly improved compared with that of the reasoner. If the node is down, the system will automatically switch to a node that is still available for reasoning after a few retries. Distributed virtual IP technology guarantees fault tolerance under distributed environment. And the ontology data is distributed and routed through gateways and eventually flows to different nodes.

4.2. Experiment
We implemented ron-s based on dl-lite. Test data is based on popular standard ontology libraries and evaluated using different methods. The benchmark ontology library includes NPD, DBPedia, IMDB and various versions of LUBM and UOBM. The NPD ontology data set is NPD Fact Pages, which contains information about oil collected on the Norwegian continental shelf. DBpedia is the data for wikipedia. MDB is data about movies. All three are real data. LUBM is the ontology benchmark of a university. For example, LUBM one represents a university and five represents five universities. UOBM data is also the ontology benchmark of the university. UOBM is an improvement on LUBM which uses ontology generators to generate owl files of different sizes. We extract the ontology that conforms to the dl-lite logical rule. The ontology that does not meet the rule will be removed. We compare the data after abstraction with the data before abstraction. Table 1 records Tbox axioms, atomic concepts, relationships, individuals and inferred assertions. Table 2 records the time required for reasoning by different inference tools. Experimental result suggests that Abox may be 10 percent or less of the original Abox. The time required for the whole reasoning process has also been significantly improved. The experimental environment is as follows: Intel(R) Core(TM) i7-6700hq CPU@2.6GHz, operating memory: 16GB, storage memory: 300GB. Ontology data is shown in table 1.

| Table 1. Some Information of Ontology |
|---------------------------------------|
| Ontology | axiom | con | Role | Indivi | assert |
|-----------|-------|-----|------|--------|--------|
| NPD       | 400   | 244 | 112  | 804322 | 1712578|
| DBPedia   | 1362  | 385 | 705  | 3236846| 21948501|
| IMDB      | 133   | 91  | 43   | 6656342| 29093482|
| LUBM 1    | 80    | 42  | 25   | 1555   | 5738   |
| LUBM 5    | 80    | 42  | 25   | 9035   | 65305  |
| LUBM 10   | 80    | 42  | 25   | 207425 | 850463 |
| UOBM 1    | 110   | 69  | 35   | 13654  | 198363 |
| UOBM 5    | 110   | 69  | 35   | 11374  | 937594 |
| UOBM 10   | 110   | 69  | 35   | 242491 | 1926897|

| Table 2. Reasoning time |
|--------------------------|
| Onto* | ron-s | konclude | hermit | ELK | Pellet |
|-------|-------|----------|-------|-----|--------|
| NPD   | 4     | 11       | 579   | 56  | 39     |
| DBPedia | 98   | 123      | 3023  | 453 | 856    |
| IMDB  | 23    | 142      | 453   | 245 | 234    |
| LU 1  | 1     | 2        | 3     | 4   | 4      |
| LU 5  | 1     | 2        | 5     | 6   | 5      |
In table 1, with the NPD, Dbpedia and IMDB increasing with the increase of ontology data volume, all data were significantly improved. However, with the increase of ontology data, the axioms, concepts and relationships of LUBM and UOBM have not changed significantly. Because the data they generate is based on the same reasoner. The conceptual materialization time is shown in table 2. In table 2, the ron-s inference algorithm takes the shortest time. The hermit inference engine is embedded in the experiment. For NPD, Dbpedia and IMDB which are based to real-world data. The ron-s comparing reasoning tool has a great optimization. The Performance has also been partially improved in the school ontology LUBM and UOBM.

5. Conclusions
In this paper, an improved reasoning algorithm is presented and experiments are carried out in a distributed environment. The NPD, Dbpedia,IMDB,LUBM and UOBM ontology were used as the benchmark ontology. Experimental results show that ron-s has better performance in distributed environment. In the experiment, we checked the consistency of abstraction and ontology, which can ensure the fairness and integrity of ontology. We mainly focus on the generation of concept assertion in the experiment. The derivation of relationship assertion can be accomplished by horn ontology reasoning refinement mentioned in the front paragraph. The volume of abox after abstraction was significantly reduced exponentially compared with the original abox, and the reasoning time was significantly optimized. In the future, we plan to apply this idea to some new fields and improve the abstract thinking and distributed environment in order to make the speed of ontology reasoning faster and the system more stable.

References
[1] Glimm B, Kazakov Y, Tran T K. Scalable Reasoning by Abstraction Beyond DL-Lite[C]//International Conference on Web Reasoning and Rule Systems. Springer, Cham, 2016: 77-93.
[2] Glimm B, Kazakov Y, Liebig T, et al. Abstraction refinement for ontology materialization[C]//International Semantic Web Conference. Springer, Cham, 2014: 180-195.
[3] HermiT OWL Reasoner[J]. URL: http://www.hermiT-reasoner.com, 2016.
[4] Thukral A, Jain A, Aggarwal M, et al. Semi-automatic Ontology Builder Based on Relation Extraction from Textual Data[M]//Advanced Computational and Communication Paradigms. Springer, Singapore, 2018: 343-350.
[5] Farid H, Haarslev V. Handling Nominals and Inverse Roles using Algebraic Reasoning[J]. arXiv preprint arXiv:1810.00916, 2018.
[6] Glimm B, Kazakov Y, Tran T K. Ontology Materialization by Abstraction Refinement in Horn SHOIF[C]//AAAI. 2017: 1114-1120.
[7] Baader F, Borgwardt S, Lippmann M. Query rewriting for DL-Lite with n-ary concrete domains[C]//Proceedings IJCAI. 2017.
[8] Zheleznyakov D, Kharlamov E, Horrocks I. Trust-Sensitive Evolution of DL-Lite Knowledge Bases[C]//AAAI. 2017: 1266-1273.
[9] Mansour E, Abdelaziz I, Ouzzani M, et al. A demonstration of Lusail: Querying linked data at scale[C]//Proceedings of the 2017 ACM International Conference on Management of Data. ACM, 2017: 1603-1606.
[10] Khan S A, Qadir M A, Abbas M A, et al. OWL2 benchmarking for the evaluation of knowledge based systems[J]. PloS one, 2017, 12(6): e0179578.
[11] Glimm B, Horrocks I, Motik B, et al. HermiT: an OWL 2 reasoner[J]. Journal of Automated Reasoning, 2014, 53(3): 245-269.
[12] Uschold M. Demystifying OWL for the Enterprise[J]. *Synthesis Lectures on Semantic Web: Theory and Technology*, 2018, 8(1): i-237.

[13] Uschold M. Demystifying OWL for the Enterprise[J]. *Synthesis Lectures on Semantic Web: Theory and Technology*, 2018, 8(1): i-237.

[14] Sazonau V, Sattler U. TBox Reasoning in the Probabilistic Description Logic SHIQp[C]//*Description Logics*. 2015.

[15] Bak J, Nowak M, Jedrzejek C. RuQAR: Reasoning framework for OWL 2 RL ontologies[C]//*European Semantic Web Conference*. Springer, Cham, 2014: 195-198.

[16] Kim K H, Hwang M S, Jae E Y, et al. A Study on Message Queue Safe Proper Time for Open API Fast Identity Online Fintech Architecture[J]. *International Journal of Software Engineering and Its Applications*, 2016, 10(5): 33-44.