METODE KLASIFIKASI OPTIMUM UNTUK DESCRIPTION LOGIC BASED-HIERARCHICAL STUCTURED-KNOWLEDGE BASE

Kurnia Muludi[(1)] dan **Kuspriyanto**[(2)]

[(1)] Universitas Lampung
kurnia@unila.ac.id

[(2)] STEI ITB Bandung
kuspriyanto@yahoo.com

Abstract

Since W3C has recommended OWL-DL, many legacy ontologies are now being translated into Description Logic (DL) based ontology languages in order to take advantage of DL based tools and reasoning services. However the resulting Knowledge Bases (KBs) are typically of large size, but have consist mainly of shallow taxonomies. In this paper we propose an optimisation which speeds-up classification for such KBs.

Keyword*: Description Logic, Classification, Information Retrieval, Hierarchical.Data, Ontology.*

1. PENDAHULUAN

Dengan adanya rekomendasi W3C tentang OWL-DL dan tersedianya tool, banyak ontology berbagai institusi diterjemahkan ke dalam bentuk Description Logic (DL) menggunakan OWL DL ontology language [2]. Bentuk *Knowledge Base* (KB) ini biasanya berukuran besar dan banyak yang mempunyai struktur yang sangat sederhana, terutama hirarki sub-class menyerupai taxonomy yang sangat lebar dan dangkal.

Beberapa algoritma telah dibuat para penelitii yang diimplementasikan pada *reasoner* seperti FACT [8] dan RACER [4] namun tidak dapat bekerja dengan baik pada *Knowledge Base* (KB) seperti tersebut di atas. Hal ini terutama disebabkan beberapa kelas dalam hirarki memiliki jumlah sub-class yang sangat besar [7]. Untuk Knowledge bases yang seperti itu fase *top-down* pada algoritma klasifikasi standar [1] akan melakukan *subsumption tests* yang sangat banyak dan hampir semuanya akan gagal (yaitu, menyimpulkan tidak terdapatnya hubungan *subsumption*). Walaupun struktur KB pada kebanyakan kasus di atas sangat sederhana, namun jumlah tes yang sangat besar akan menyebabkan masalah pada performa.

Dalam paper ini akan dibahas masalah optimasi *classification* sehingga komputasi *concept* dalam hirarki dapat meminimumkan *negative subsumption test*.

.

2. PENELITIAN TERKAIT

Liu et. al [6] melakukan percobaan pada data Yahoo! Directory yang bertujuan diantaranya mendapatkan ambang *tuning algorithm* dalam hal *time complexity* dan pengaruhnya pada akurasi klasifikasi menggunakan algoritma Support Vector Machine (SVM). Hasil penelitian menunjukkan: dalam hal skalabiliti SVM cukup efisien untuk klasifikasi yang sangat besar. Distribusi data yang *skewed* pada Yahoo! Directory membuat kinerja algorirtma SVM masih belum memuaskan sehingga penelitian lebih lanjut sangat dibutuhkan.

Haarslev dan Moller [4] melakukan experiment pada KB yang berisi lebih dari 160.000 *concept* dengan menggunakan teknik optimasi melalui *topology sorting*, teknik *bucket-clustering* dan teknik *caching*. Hasil Implementasi Common Lisp pada sistem ini (RACE) menghasilkan algoritma yang kurang optimal.

Mensiasati masalah klasifikasi dapat dilakukan dengan menggabungkan antara *knowledge base* untuk menyimpan *concept* (Tbox) dan database untuk menyimpan *instant* (Abox) seperti pada sistem yang diberi nama Instance Store [5]. Dengan menggunakan teknik ini didapat hasil yang cukup baik namun terkendala masalah komunikasi antara Intstant Store dengan DL Reasoner yang memakan waktu 50% dari total waktu yang dibutuhkan menjawab query.

Dengan meminimalkan proses test pada *subsumee* dalam proses klasifikasi KB yang besar tetapi berhirarki dangkal, [7] dapat mempercepat proses klasifikasi pada KB yang berkonsep primitif tetapi tidak signifikan pada KB yang banyak mengandung *concept* non-primitif.

Dalam paper ini akan diusulkan suatu alternatif metode klasisifikasi pada *knowledge base* yang berbentuk hirarki (*taxonomy*).

3. OVERVIEW SISTEM

Description Logics merupakan *concept* (*class*) *based knowledge representation systems*. Adanya spesifikasi formal dan *decidability* memungkinkan DL digunakan dalam *decision procedures* (*sound*, *complete* dan *terminating algorithm*) untuk *reasoning task* seperti *concept subsumption*. *DL Knowledge Base* terdiri dari dua bagian Tbox dan Abox. Tbox terdiri dari himpunan aksioma yang mendeskripsikan *constraints* pada *instant* suatu *concept* (setara dengan skema konseptual pada database). Sedangkan Abox terdiri dari aksioma yang mendeskripsikan hubungan-hubungan antara individu dan *concepts* (setara dengan data pada database). Aksioma Tbox dalam bentuk $C \sqsubseteq D$ atau $C \vee D$, dimana C dan D adalah *concepts*. Jika C adalah nama *concept*, maka aksioma tersebut sering disebut definisi (dari C), dan jika suatu definisi dalam bentuk $C \sqsubseteq D$ maka disebut *primitive concept*.

Proses klasifikasi Tbox merupakan proses *computing* dan *caching* hirarki dari *concept* untuk semua *concept* yang bernama yang ada dalam Tbox, atau juga merupakan proses *computing* pengurutan *parsial subsumption* dari *concept* bernama. Proses klasifikasi Tbox merupakan *basic reasoning task* pada *DL reasoners* (dan dapat juga digunakan untuk proses *query answering*).

Sebelum membahas metode yang diusulkan, berikut disajikan prosedur standar *computing* hirarki *concept* yang dioptimalkan yang diusulkan [1]. Dalam prosedur ini nama-nama *concept* diurut dalam urutan definisi, yaitu jika nama *concept* D ada dalam definisi nama *concept* C, maka $D \leq C$. Hirarki *concept* diinisialisasi dengan memasukkan dua *concept* $\top$ dan $\bot$ (dimana $\top$ merupakan *super concept* dari $\bot$) kemudian *concept* diklasifikasikan (ditambahkan ke hirarki pada posisi yang sesuai) satu persatu dalam urutan menurut definisi.

Terdapat dua fase dalam pengkelasan *concept*: *top-down phase* dimana induk (*direct subsumers*) diproses, dan *bottom-up phase* dimana anak (*direct subsumees*) diproses. Dalam penambahan *concepts* menurut urutan definisi dapat mengabaikan *bottom-up phase* (karena ketika *concept* baru diklaskan, ia hanya memiliki anak $\bot$).

Berbagai optimasi digunakan untuk meminimalkan jumlah pengujian *subsumption* yang dibutuhkan pada tiap *phase*. Sebagai contoh, pada saat menambahkan *concept* C ke hirarki, *top-down breadth first traversal* dilakukan hanya untuk memeriksa apakah D merupakan induk dari C. Tentu saja subsumer (D) harus diklasifikasikan terlebih dahulu sebelum mengklasifikasikan C.

Dalam paper ini proses optimasi diusulkan dilakukan melalui pemanfaatan *external ontology* sebagai bahan pertimbangan bagi algoritma sebelum melalukan klasifikasi. Ontology ini berisi hubungan antar *concept* penting (penulis sebut *root concept*) dalam hal ini merupakan *concept* yang mempunyai *subsumee* yang banyak. Melalui informasi ini algoritma akan melakukan *breadth treversal test* terbatas sehingga diharapkan dapat mengoptimum proses klasifikasi.

4. DESAIN SISTEM DAN ANALISIS

Pada Description Logic *reasoning* dasar untuk *concept* dalam Tbox terdiri dari [9]:

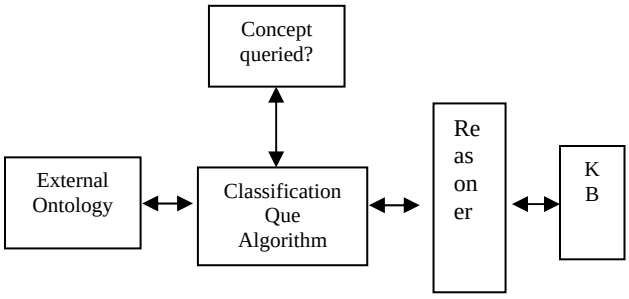
- Konsistensi KB : suatu Tbox $\mathcal{T}$ konsisten jika dan hanya jika ia *satisfiable*, yaitu setidaknya terdapat suatu model untuk $\mathcal{T}$. Suatu interpretasi pada $\mathcal{I}$ merupakan suatu model untuk $\mathcal{T}$ jika ia *satifiifies* untuk tiap *assertion* pada $\mathcal{T}$.
- *Satisfiability* : suatu concept C *satisfiable* dalam hal $\mathcal{T}$ jika terdapat suatu model $\mathcal{I}$ dari $\mathcal{T}$.
- *Subsumption*: suatu concept D mengindukii (*sumbsumes*) suatu concept C dalam hal $\mathcal{T}$ ($C \sqsubseteq_T D$ atau $\mathcal{T} \models C \sqsubseteq D$) Jika $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ untuk tiap model $\mathcal{I}$ dari $\mathcal{T}$.
- *Equivalence*: dua concept C dan D *equivalent* dalam hal $\mathcal{T}$ jika $C^{\mathcal{I}} = D^{\mathcal{I}}$ untuk tiap model $\mathcal{I}$ dari $\mathcal{T}$.
- *Disjointness*: dua *concept* C dan D *disjoint* dalam hal T, jika untuk tiap model $\mathcal{I}$ dari $\mathcal{T}$, $C^{\mathcal{I}} \cap D^{\mathcal{I}} = 0$.

Sedangkan *reasoning* untuk Abox terdiri dari [9]:

- *Instantiation test/instance check*: menentukan apakah suatu *assertion* di-*entailed* oleh Abox $\mathcal{A}$ atau tidak. *Assertion* di-*entailed* oleh Abox ($\mathcal{A} \models \mathcal{C}(a)$) jika tiap interpretasi yang *satisfies* $\mathcal{A}$ (tiap model $\mathcal{A}$), juga *satisfies* $\mathcal{C}(a)$.
- *Realisation*: jika diketahui suatu individu a dan suatu himpunan *concept*, cari *concept* paling specifik $\mathcal{C}$ dari himpunan tersebut sehingga $\mathcal{A} \models \mathcal{C}(a)$.
- *Retrieval*: jika terdapat suatu Abox $\mathcal{A}$ dan *concept* $\mathcal{C}$, cari individu-individu sehingga $\mathcal{A} \models \mathcal{C}(a)$.
- *Abox Concistency*: Abox $\mathcal{A}$ konsisten jika dan hanya jika ia konsisten dalam hal Tbox $\mathcal{T}$.

Menurut [5] *reasoning task* seperti *classification* dan *retrieval* dapat di sederhanakan menjadi *subsumption* dan *instantiation*. Sedangkan *Realisation* dapat disederhanakan menjadi kombinasi dari *retrieval* dan *classification*. Pada proses *classification,* pengkelasan concept $\mathcal{C}^q$ dalam hirarki KB $\mathcal{K}$, dilakukan melalui *subsumption test* yang disediakan oleh reasoner seperti $\mathcal{FACT}$++ [8].

Yang menjadi isu penting adalah proses tersebut dilakukan *breadth first traversal* sehingga untuk mengoptimalkannya diperlukan strategi meminimalkan *travel*. Hal ini dapat dicapai dengan memanfaatkan informasi yang disediakan oleh *external ontology*. *External ontology* merupakan 'miniatur' dari KB $\mathcal{K}$ yang berisi *root concept.* Model sistem klasifikasi yang diusulkan disajikan pada Gambar 1.



Gambar 1. Model sistem klasifikasi yang diusulkan.

Berikut algoritma *concept classification* :

```
Input KB, C_Q
    /* KB=Knowledge Base,
    /* C_Q = Concept yang diquery

Cek_C_Q_pada_external_ontology(Oe, C_Q)
Jika True
   C_index = index(Oe, C_Q)
   Classify_using_standar_reasoner(KB,
     C_index)
Jika False
   do breadth_first_travel_search_
   pada_standar_reasoner (KB, C_Q)
end
```

Sedangkan algoritma pengembangan *external ontology* adalah sebagai berikut:

```
Input KB, limit
    /* limit = % subsumee pada suatu
    /*         concept terhadap semua
    /*        jumlah concept dalam KB

External_ontology := {}

For all concept C_i pada KB do
    %Subsumee_Ci = hitung % sub-concept
                   dari Ci thd KB

    If %Subsumee_Ci ≥ limit then
             External_ontology += Ci

End
```

Pada *reasoning* standar menggunakan *breadth first traversal* akan dijumpai tingkat kompleksitas $O(b^{d+1})$ dimana b merupakan *branching factor* (jumlah *concept* dalam KB) dan d merupakan kedalaman solusi terdangkal dalam *tree/taxonomy*. Dengan menggunakan algoritma yang diajukan, tingkat kompleksitas akan lebih kecil dari $O(b^{d+1})$.

5. KESIMPULAN

Adanya rekomendasi W3C tentang OWL-DL menyebabkan banyak *legacy ontology* (KB) di transfer kedalam bentuk OWL-DL. Namun pada prakteknya KB yang dihasilkan umumnya berciri bervolume besar dengan struktur hirarkis dangkal dan sangat lebar. Hal ini menyebabkan proses *reasoning* menjadi sulit. Dengan meminimalkan travel pada algoritma pengklasifikasian diharapkan proses *reasoning* dapat dilakukan lebih cepat dan optimal.

6. FUTURE WORKS

Beberapa kegiatan yang akan dilakukan di masa mendatang:

- Penyempurnaan teknik pencarian/ pengembangan *root concept* dalam pengembangan *external ontology* dari KB $\mathcal{K}$.
- Penyempurnaan *Classification Que Algorithm*.
- Melakukan eksperimen untuk menguji kinerja algoritma pada berbagai KB berbasis DL berskala besar, misal Yahoo Directory, Open Galen, dan *self-made KB*.

7. DAFTAR PUSTAKA

[1] F. Baader, E. Franconi, B. Hollunder, B. Nebel, and H.-J. Profitlich. An empirical analysis of optimization techniques for terminological representation systems or: Making KRIS get a move on. Applied Artificial Intelligence. Special Issue on Knowledge Base Management, 4:109–132, 1994

[2] S. Bechhofer, F. van Harmelen, J. Hendler, I. Horrocks, D. L. McGuinness, P. F. Patel-Schneider, and L. A. Stein. OWL web ontology language reference. W3C Recommendation, 10 February 2004. http://www.w3.org/TR/owl-ref/

[3] V. Haarslev and R. Moller. 2001. Description of the RACER System. and its Applications. Proc. Of The Int. Join Conf. on Automated Reasoning (IJCAR). Springer.

[4] V. Haarslev and R. Moller. 2001. High Performance Reasoning with Very Large

Knowledge Bases. Proc. of the 17th Int. Joint Conf. on Artificial Intelligence.

[5] I. Horrocks, L. Li, D. Turi, and S. Bechhofer. 2004. The Instance Store: Description Logic Reasoning with Large Numbers of Individuals. Int. Workshorp on Description Logic (DL 2004).

[6] Tie-Yan Liu, Yiming Yang, Hao Wan, Hua-Jun Zeng, Zheng Chen, and Wei-Ying Ma. 2005. Support Vector Machines Classification with A Very Large-scale Taxonomy. SIGKDD Explor. Newsl.

[7] D. Tsarkov and I. Horrocks, 2005. Optimised Classification for Taxonomic Knowledge Bases. Proc. of International Workshop on Description Logics. http://ceur-ws.org/Vol-147/39-TsarHorr.pdf

[8] D. Tsarkov and I. Horrocks, 2006. FaCT++ Description Logic Reasoner: System Description. Proc. Of the 3rd Int. Join Conf. on Automated Reasoning.

[9] Baader, F., Calvanese, D., McGuinness, D., Nardi, D.& Patel-Schneider, P., eds (2003), The Description Logic Handbook: Theory, Implementation, and Applications, Cambridge University Press, New York.