

ЭФФЕКТИВНЫЙ ЛОГИЧЕСКИЙ АНАЛИЗ БОЛЬШИХ ОНТОЛОГИЙ ЗА ПОЛИНОМИАЛЬНОЕ ВРЕМЯ

В статье рассматривается проблема вычислительной сложности логического анализа онтологий. Рассмотрены дескриптивные логики \mathcal{EL} и \mathcal{EL}^{++} для которых задача логического анализа имеет полиномиальную сложность. Предложено ряд улучшений для логического анализатора ELK с целью поддержки конкретных доменов (\mathcal{D}) и утверждений о экземплярах (ABox).

This paper overviews the problem of computational complexity of reasoning with ontologies. The description logic \mathcal{EL} and \mathcal{EL}^{++} is considered where logical entailment could be accomplished in polynomial time. We propose few modification to new and very promising ELK reasoner to support concrete domains (\mathcal{D}) and ABox assertions.

Вступление

Исследования в области представления знаний в значительной степени основаны на методах описания окружающего мира, его объектов и свойств в форме пригодной для использования в интеллектуальных системах. В данном контексте под понятием интеллектуальной системы подразумевается система, способная извлечь неявно указанную информацию из множества явно заданных утверждений.

Методы представления знаний можно условно разделить на две категории. В первую категорию входят методы основанные на формальной логике, а в другую – неформальные методы. В отличие от формальных моделей, в основе которых лежит строгая математическая теория, неформальные методы чаще всего используют когнитивные приёмы для представления знаний в виде специальных структур, например таких как семантические сети или фреймы. При этом, несмотря на наглядность неформальных методов, отсутствие чётких универсальных правил их интерпретации позволяло эффективно применять такие методы лишь для некоторых конкретных задач. Формальные же методы представления знаний изначально развивались на основе механизмов логики предикатов первого порядка, что обусловило их универсальность.

В последствии желание внедрить семантику в интерпретацию семантических сетей и фреймов привело к появлению нового семейства языков представления знаний – дескриптивной логики [1] (ранее известная как терминологическая система, а потом логика концептов). В этом новом формализме были скомбинированы фреймовые структуры с фрагментами логики

первого порядка, что позволило создать гибкий и выразительный язык представления знаний с хорошо изученными свойствами. При этом вычислительная сложность языка на прямую зависит от того, какие именно фрагменты логики первого порядка мы выбираем.

На практике одним из самых известных воплощений дескриптивной логики является онтология – формализация некоторой области знаний в виде множества понятий и отношений между ними. Онтологии нашли широкое применение в информатике, в частности в задачах накопления и обработки данных с учётом их семантики, при построении "порталов знаний", моделировании, а также в качестве посредника между пользователем и информационной системой или другими пользователями.

Отдельно стоит упомянуть, что онтологиям выделена центральная роль в планах развития Всемирной паутины. Современные методы автоматической обработки данных, доступных в Интернете, как правило, основаны на частотном и лексическом анализе текстового содержимого, которое, прежде всего, предназначено для восприятия человеком. Применение же метаданных и онтологий позволило бы совершать машинную обработку этих колоссальных объёмов информации, что вызвало б революцию в этой области. Консорциум Всемирной паутины (W3C) утвердил и активно развивает язык описания онтологий OWL, вокруг которого сегодня сконцентрировано значительную часть всех исследований в области представления знаний, онтологического инжиниринга и практического применения дескриптивной логики.

Однако массовому внедрению онтологий и использованию интеллектуальных систем в целом препятствует тот факт, что алгоритмы ло-

гического анализа в таких системах имеют чрезвычайно высокую вычислительную сложность. Начало исследованию вычислительной сложности дескриптивной логики положила статья [2] в которой была затронута эта проблема. В ней рассматривалась оценка вычислительной сложности логического анализа и те факторы, которые на неё влияют. Оказалось, что различные комбинации разрешённых конструкций языка порождали языки описания с совершенно разными вычислительными свойствами. Авторами был исследован один из таких языков – \mathcal{FL}^- для которого алгоритм логического анализа сохранял полиномиальную сложность. Впоследствии был изучен ещё один язык – \mathcal{EL} [3] и \mathcal{EL}^{++} [4], который сохранял разрешимость за полиномиальное время и при этом был практичнее.

Разрабатывая нашу интеллектуальную систему [5] управления ресурсами Грид, в которой центральную роль играют онтологии, мы столкнулись с недопустимо низкой производительностью логического анализа над базой знаний. Перепробовав различные алгоритмы и методы оптимизации мы пришли к выводу, что табличный алгоритм, который лежит в основе почти всех логических анализаторов, не может справиться с онтологиями большого размера так как изначально был рассчитан на языки с большой экспрессивностью и выполняет процесс классификации онтологии путём итеративного построения модели для каждой пары концептов и поиск противоречий в ней. Согласно [1] такая задача имеет сложность $NExpTime$, что в комбинации с очень большим размером онтологии в нашем случае исключает этот подход.

За решением проблемы мы обратились к опыту наших коллег, которые разрабатывали интеллектуальные системы на основе одних из самых больших онтологий на сегодняшний день – *SNOMED CT*, *GALEN* и *Gene Ontology*. Дабы обеспечить приемлемую скорость классификации и масштабируемость, упомянутые онтологии были построены с применением дескриптивной логики \mathcal{EL}^{++} .

Дескриптивная логика \mathcal{EL}^{++}

Словарь логики \mathcal{EL} состоит из исчислимо бесконечного множества атомарных ролей N_R , множества атомарных концептов N_C и верхнего концепта \top . Сложные концепты могут быть индуктивно построены с помощью выражений

конъюнкции ($C \sqcap D$) и квантора существования $\exists R.C$, где $R \in N_R$.

\mathcal{EL} терминология, так называемый *TBox* – это конечное множество определений типа $A \equiv C$ и $A \sqsubseteq C$ где никакое имя концепта не определяется более одного раза. Выражение $A \equiv C$ определяет \mathcal{EL} -концепт A эквивалентный концепту C , а выражение $A \sqsubseteq C$ выражает вложенность концепта A в концепте C .

Причина, по которой логика \mathcal{EL} не содержит дизъюнкции (\sqcup) и квантора всеобщности (\forall) заключается в том, что наличие этих операций в языке делает задачу логического анализа $NExpTime$ – полной. Как было показано в [1] и [3] дизъюнкция является одним из основных источников вычислительной сложности и недетерминизма логического анализа, так как требует построения модели для каждого варианта и последующей её проверки. Квантор \forall , в свою очередь, может вызывать аналогичный скачок вычислительной сложности в комбинации с экзистенциальной конструкцией $\exists R.C$.

Логика \mathcal{EL}^+ [6] расширяет \mathcal{EL} конструкцией $r \circ s \sqsubseteq t$, где $r, s, t \in N_R$ с помощью которой можно выразить такие важные выражения как иерархии ролей ($r \sqsubseteq s$), транзитивные роли ($r \circ r \sqsubseteq r$) и цепочки ролей. Последние играют особо важную роль в составлении сложных медицинских и био-онтологий.

Наконец, логика \mathcal{EL}^{++} расширяет логику \mathcal{EL}^+ номиналами, пустым концептом \perp , рефлексивными ролями, а также определением домена и диапазона ролей с некоторыми ограничениями. Наличие концепта \perp среди прочего позволяет сделать утверждение о различии концептов: $C \sqcap D \sqsubseteq \perp$. Полный синтаксис и семантика логики \mathcal{EL}^{++} представлена в Таблице 1. Тут и дальше используется семантика Тарского.

Табл. 1 Синтаксис и семантика языка \mathcal{EL}^{++}

Выражение	Семантика
\top	$\Delta^{\mathcal{I}}$
\perp	\emptyset
$\{a\}$	$\{a^{\mathcal{I}}\}$
$C \sqcap D$	$C^{\mathcal{I}} \cap D^{\mathcal{I}}$
$\exists r.C$	$x \in \Delta^{\mathcal{I}} \mid \exists y \in \Delta^{\mathcal{I}}: (x, y) \in r^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}$
$C \sqsubseteq D$	$C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$
$r_1 \circ \dots \circ r_k \sqsubseteq r$	$r_1^{\mathcal{I}} \circ \dots \circ r_k^{\mathcal{I}} \subseteq r^{\mathcal{I}}$
$\text{dom}(r) \sqsubseteq C$	$r^{\mathcal{I}} \subseteq C^{\mathcal{I}} \times \Delta^{\mathcal{I}}$
$\text{ran}(r) \sqsubseteq C$	$r^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times C^{\mathcal{I}}$
$C(a)$	$a^{\mathcal{I}} \in C^{\mathcal{I}}$
$r(a, b)$	$(a^{\mathcal{I}}, b^{\mathcal{I}}) \in r^{\mathcal{I}}$

Консорциум Всемирной паутины (W3C) в рамках новой версии языка описания онтологий Web Ontology Language (OWL) утвердил профиль *OWL 2 EL* который коррелирует с логикой \mathcal{EL}^{++} .

Хорошие вычислительные свойства и достаточная экспрессивность *OWL 2 EL* были залогом стремительного развития онтологий на его основе. Особенно большую популярность такие онтологии получили в области медицины и генетики.

Вслед за онтологиями появилось множество логических анализаторов способные за приемлемое время производить их классификацию. Среди них особо стоит упомянуть CB, CEL, jCEL, Snoroket и ELK. Исчерпывающий обзор логических анализаторов \mathcal{EL}^{++} представлен в [7].

Упомянутый выше логический анализатор ELK благодаря своей продуманной масштабируемой архитектуре стал в основу нашей работы.

Логический анализатор ELK

ELK является свободно распространяемым логическим анализатором с открытым исходным кодом для *OWL EL* онтологий. По результатам тестирования он способен классифицировать онтологию SNOMED CT, содержащую более 300 000 концептов, менее чем за 5 секунд. Такой результат достигается за счёт оптимизированных алгоритмов логического анализа способных использовать преимущества многоядерных процессоров.

Логический анализ в ELK происходит путём применения к множеству исходных аксиом, содержащихся в исходной онтологии, трансформирующих правил которые порождают новые аксиомы-следствия. Список таких правил приведён в Таблице 2.

Табл. 2 Базовые правила вывода ELK

$$\begin{array}{ll}
 R_{\sqsubseteq} \frac{C \sqsubseteq D}{C \sqsubseteq E} : D \sqsubseteq E \in \mathcal{O} & R_{\sqcap}^+ \frac{C \sqsubseteq D_1 \quad C \sqsubseteq D_2}{C \sqsubseteq D_1 \sqcap D_2} : D_1 \sqcap D_2 \in \mathcal{O} \\
 R_{\sqcap}^- \frac{C \sqsubseteq D_1 \sqcap D_2}{C \sqsubseteq D_1} & R_{\exists}^+ \frac{C \sqsubseteq D}{\exists S. C \rightarrow \exists S. D} : \exists S. D \in \mathcal{O} \\
 & C \sqsubseteq D_2 \\
 R_{\exists}^- \frac{C \sqsubseteq \exists R. D}{D \sqsubseteq D} & R_{\mathcal{H}} \frac{D \sqsubseteq \exists R. C \quad \exists S. C \rightarrow E}{D \sqsubseteq E} : R \sqsubseteq_o^* S \\
 R_{\top}^+ \frac{C \sqsubseteq C}{C \sqsubseteq \top} & R_{\mathcal{T}} \frac{D \sqsubseteq \exists R. C \quad \exists S. C \rightarrow E}{\exists T. D \rightarrow E} : R \sqsubseteq_o^* T \sqsubseteq_o^* S \\
 & \text{Trans}(T) \in \mathcal{O}
 \end{array}$$

В работе [8], которая детально описывает алгоритм работы ELK, было доказано его корректность и полноту.

Для повышения производительности алгоритм был распараллелен таким образом, что бы максимизировать масштабируемость процесса логического анализа.

В статье [9] были представлены правила вывода для работы с номиналами – концептами содержащими всего один элемент. Для их реализации необходимо было ввести понятие достижимости \rightsquigarrow которое интерпретируется следующим образом: $I \models C \rightsquigarrow D$ если $C^I \neq \emptyset$ и $D^I \neq \emptyset$

Табл. 3 Правила вывода для номиналов

$$\begin{array}{ll}
 R_{\rightsquigarrow}^- \frac{G \rightsquigarrow D}{G : D \sqsubseteq D} & R_{\rightsquigarrow}^+ \frac{G \rightsquigarrow C \quad G : C \sqsubseteq \exists R. D}{G \rightsquigarrow D} \\
 R_{\emptyset} \frac{G : C \sqsubseteq \{o\} \quad G : D \sqsubseteq \{o\} \quad G \rightsquigarrow C \quad G \rightsquigarrow D}{G : C \sqsubseteq D}
 \end{array}$$

Таким образом, возможности ELK покрывают значительную часть профиля OWL EL. Однако, для практических целей, в том числе и для нашей системы этого было недостаточно. Так ELK не поддерживает логический анализ над типами данных и рассуждения с экземплярами концептов (ABox).

Поддержку типизированных ролей (конкретных доменов) а так же разрешённых выражений с их участием существенно ограничили в профиле языка OWL EL. Единственным разрешённым выражением является равенство ($=$), так как его присутствие в онтологии почти не влияло на сложность её логического анализа. В то же время, некоторые не очевидные комбинации других операторов могли симулировать поведение дизъюнктивного выражения в онтологии, что является основной причиной значительного увеличения вычислительной сложности. В качестве примера рассмотрим такой набор аксиом: $A \sqsubseteq \exists R. (\leq 5)$, $\exists R. (\leq 2) \sqsubseteq B$ и $\exists R. (\geq 2) \sqsubseteq C$. Не сложно заметить, что таким образом мы неявно выразили дизъюнкцию вида $A \sqsubseteq B \sqcup C$.

За решением этой проблемы мы обратились к работе [10], целью которой было найти такие комбинации выражений с типизированными ролями, при которых сохранялось бы свойство разрешимости логического анализа за полиномиальное время. Авторами статьи были определены все возможные ограничения использования типизированных ролей при построении

концептов для множества натуральных, целых, вещественных и рациональных чисел. На основе представленных доказательств можно сделать вывод, что логика \mathcal{EL} может быть расширена подобными конструкциями, а имеющиеся ограничения не столь существенны и позволяют выражать большинство необходимых выражений.

Для работы с типизированными ролями было предложено внедрить в правила вывода ELK выражения, указанные в Таблице 4, где $A, B \in N_C$, $r \in N_R$, $F \in N_F$ и $F^J \subseteq \Delta^J \times \mathcal{D}$, а выражение $r_+ \rightarrow_D \perp$ означает, что из ограничения следует пустое множество, в то время как выражение $r_+ \rightarrow_D r_-$ означает, что ограничение справа от \sqsubseteq удовлетворяет ограничению слева.

Табл. 4 Правила вывода для типизированных ролей

$\frac{}{A \sqsubseteq \perp}$	$A \sqsubseteq \exists F.r_+ \in \mathcal{O}, r_+ \rightarrow_D \perp$
$\frac{A \sqsubseteq \exists F.r_+}{A \sqsubseteq B}$	$\exists F.r_- \sqsubseteq B \in \mathcal{O}, r_+ \rightarrow_D r_-$
$\frac{A \sqsubseteq B}{A \sqsubseteq \exists F.r_+}$	$B \sqsubseteq \exists F.r_+ \in \mathcal{O}$

Следующим значительным недостатком ELK было отсутствие поддержки утверждений об индивидах (*ABox*). Возможность логического анализа с большим количеством индивидов постепенно приобретает всё большее значение для интеллектуальных систем, выдвигая новые требования к их реализации, так как размеры *ABox* зачастую многократно превышают размеры *TBox*.

Ведётся активная работа по реализации этого функционала и вскоре ELK будет поддерживать работу с экземплярами в рамках профиля OWL EL.

Тем не менее, существует способ симулировать поддержку логического анализа над индивидами в логическом анализаторе без поддержки *ABox*. Для этого необходимо совершить серию трансформаций:

1. Заменить все утверждения о экземплярах вида $C(a)$ на $A^* \sqsubseteq C$, где A^* – новый концепт репрезентирующий a . Для онтологий, составленных с помощью функционального синтаксиса OWL это означает замену аксиомы типа *ClassAssertion*(*prefix:C prefix:a*) на аксиомы типа *SubClassOf*(*prefix:a prefix:C*). Для того что бы различать терминологические кон-

цепты от концептов-индивидов, имя последних формируется специальным образом либо аннотируется необходимыми метаданными.

2. Заменить все отношения (роли) между индивидами $r(a, b)$ на экзистенциальные конструкции $A^* \sqsubseteq \exists r.B^*$, где A^* и B^* – новые концепты, репрезентирующие индивидов a и b соответственно. Для OWL онтологий это означает замену выражений *ObjectPropertyAssertion*

(*prefix:OP prefix:a1 prefix:a2*) на выражение типа *SubClassOf*(*prefix:a1 ObjectSomeValuesFrom(prefix:OP prefix:a2)*).

Представленный подход может быть успешно использован в большинстве случаев после некоторой модификации интерфейса между логическим анализатором и интеллектуальной системой.

Заключение

Использование онтологий в интеллектуальных системах, несмотря на множество преимуществ, имеет один существенный недостаток, который тормозит их внедрение. Задача логического анализа в OWL DL онтологии в худшем случае имеет сложность *NExpTime*, в OWL 2 – *2NExpTime*, Horn *SHIQ* – *ExpTime* и т.д. При достаточно больших базах знаний практическая польза нивелируется чрезвычайно низкой скоростью её обработки.

По этой причине были направлены усилия на поиск логик, в которых бы сохранялась полиномиальная сложность логического анализа, а язык обладал бы достаточной экспрессивностью для практического применения. Семейство логик $\mathcal{EL} / \mathcal{EL}^{++}$ отвечает этим требованиям.

Логический анализатор ELK зарекомендовал себя эффективным и масштабируемым инструментом для работы с онтологиями профиля OWL EL основанного на логике \mathcal{EL}^{++} , однако полную поддержку всех конструкций языка ещё предстоит реализовать. В этой работе были рассмотрены некоторые расширения к ELK, которые позволяют совершать логический анализ с типизированными свойствами и множеством экземпляров (*ABox*). Представленные расширения будут интегрированы в новую версию ELK в скором времени.

Список литературы

1. Baader, F., Calvanese, D., McGuinness, D.L., Nardi, D., Patel-Schneider, P.F. The Description Logic Handbook: Theory, Implementation, and Applications. 2nd ed. Cambridge University Press, 2007
2. R. J. Brachman, H. J. Levesque. The tractability of subsumption in frame-based description languages. In Proceedings of the 4th National Conference on Artificial Intelligence (AAAI-84), pp. 34-37, 1984.
3. F. Baader, S. Brandt, and C. Lutz. Pushing the EL Envelope. In Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence, pp. 364–369, 2005.
4. F. Baader, S. Brandt, and C. Lutz. Pushing the EL Envelope Further. In Proceedings of the OWLED 2008 Workshop on OWL: Experiences and Directions, 2008.
5. А. Поспешный, С. Стиренко. GRID-DL – семантический информационный сервис ГРИД // Компьютинг, 2011. – Том 10, Выпуск 3. – С. 285 – 294.
6. F. Baader, C. Lutz, and B. Suntisrivaraporn. Is Tractable Reasoning in Extensions of the Description Logic EL Useful in Practice? In Proceedings of the Methods for Modalities Workshop, 2005.
7. K. Dentler, R. Cornet, A. Ten Teije, N. De Keizer. Comparison of reasoners for large ontologies in the OWL 2 EL profile. Semantic Web, vol. 2(2), pp. 71–87, 2011.
8. Y. Kazakov, Markus Krötzsch. Concurrent Classification of EL Ontologies. In Proceedings of the 10th International Semantic Web Conference (ISWC-11), pp. 305-320, 2011.
9. Y. Kazakov, M. Krötzsch, F. Siman. Practical Reasoning with Nominals in the EL Family of Description Logics. Artificial Intelligence, 2011.
10. D. Magka, Y. Kazakov, I. Horrocks. Tractable Extensions of the Description Logic EL with Numerical Datatypes. In Proceedings of the 5th International Joint Conference on Automated Reasoning (IJCAR 2010), 2010.