УДК 004.891.2

DOI: 10.15827/0236-235X.123.537-542

Дата подачи статьи: 25.04.18

2018. Т. 31. № 3. С. 537–542

Повышение скорости логического вывода продукционных экспертных систем путем использования аспектно-ориентированного подхода

А.А. Гончаров¹, аспирант, andre_goncharov@hotmail.comН.А. Семенов¹, д.т.н., профессор, is@tstu.tver.ru¹ Тверской государственный технический университет, г. Тверь, 170026, Россия

Каждая экспертная система базируется на модели представления знаний. К наиболее общим моделям относятся продукционная модель, семантические сети и фреймы. Наиболее распространено применение продукционной модели.

В статье рассматривается актуальный недостаток продукционных систем, связанный с низкой эффективностью процесса логического вывода, по сравнению с другими моделями представления знаний.

Описан предлагаемый способ повышения эффективности процесса логического вывода в продукционных системах, основанный на использовании аспектно-ориентированного подхода.

Аспектно-ориентированный подход позволяет выявлять пересекающиеся функциональные элементы и обеспечивать их консолидацию в ходе создания архитектуры и реализации системы. Впервые представленный в 1997 году, данный подход популярен и в настоящее время.

В качестве примера приводится набор из продукционных правил экспертной системы выбора требований, предъявляемых к заданному уровню контроля в соответствии с требованиями руководящих документов. В данном наборе продукционных правил фактами являются значения уровней контроля, а в виде действий представлены требования, предъявляемые к выбранному уровню контроля.

Предложенный аспектно-ориентированный подход к организации продукционных систем позволил повысить скорость логического вывода в экспертных системах. Сокращение количества операций при поиске решения и избавление от перебора фактов и действий стало возможным благодаря выделению в аспекты пересекающихся фактов и действий из набора продукционных правил.

Ключевые слова: аспектно-ориентированное программирование, АОП, экспертная система, продукционное правило, искусственный интеллект.

Поиск аналогии решения интеллектуальных задач экспертами нашел свое отражение в *экспертных системах* (ЭС).

ЭС – направление исследований в области искусственного интеллекта по созданию вычислительных систем, умеющих принимать решения, схожие с решениями экспертов (высококвалифицированных специалистов) в заданной предметной области [1].

Результат интереса к ЭС, продиктованный их умением решать задачи интерпретации данных, диагностики, мониторинга, проектирования, прогнозирования, планирования и обучения, проявился в виде активного применения ЭС в различных отраслях промышленности. Насчитываются тысячи ЭС, успешно решающих задачи определенного узкого класса, и это позволяет говорить о том, что ЭС сейчас составляют мощную ветвь в индустрии программных средств [2].

Каждая ЭС базируется на модели представления знаний. К наиболее общим моделям относятся продукционная модель (или система, основанная на правилах), семантические сети и фреймы. В некоторых случаях прибегают к использованию нечетких правил и нейронных сетей.

Наиболее распространенной моделью представления знаний для ЭС является продукционная модель. Этому способствуют модульная организация, наличие средств объяснения и аналогии с познавательным процессом человека [3].

Одним из актуальных недостатков продукционных систем, согласно [4], является более низкая эффективность логического вывода по сравнению с другими моделями представления знаний. Это происходит из-за того, что значительная часть времени при логическом выводе затрачивается на непроизводительную проверку применимости правил.

Авторы статьи предлагают повысить эффективность процесса логического вывода в продукционных системах путем использования аспектно-ориентированного подхода.

Продукционная модель представления знаний

Впервые системы, основанные на продукционных правилах (production rule), были предложены в 1943 году Э. Постом.

Продукционные системы представляют знания о решении задач в виде правил продукции – выражений вида «Если [условие], то [действие]». К продукционным правилам применимы логические операции И, ИЛИ, НЕ.

В продукционном правиле антецедентом (Antecedent) называется условная часть, консеквентом (Consequent) – часть действия. Для сопоставления антецедентов и выполнения консеквентов предназначен интерпретатор правил.

В наиболее абстрактном виде БД продукционной системы схематично представлена на рисунке 1.

На данной схеме набор правил (RULE) образует базу знаний продукционной системы, набор исходных фактов составляет базу фактов (FACT), набор действий представляется в виде базы действий (ACTION), факты составляют базу антецедентов (ANTECEDENT), а действия образуют базу консеквентов (CONSEQUENT).

Рассмотрим продукционную систему, применяемую в ЭС выбора требований, предъявляемых к заданному уровню контроля в соответствии с требованиями руководящих документов. Представим фрагмент набора продукционных правил:

Правило 1
Если
 [Выбран 4-й уровень контроля]
то
 [Выполнить контроль состава и содержания документа Спецификация (ГОСТ 19.202-78)]

Правило 2
Если
 [Выбран 4-й уровень контроля]
то
 [Выполнить контроль состава и содержания документа Описание программы (ГОСТ 19.402-78)]

Правило 3
Если
 [Выбран 4-й уровень контроля]
то
 [Выполнить контроль состава и содержания документа Описание применения (ГОСТ 19.502-78)]

...

Правило 19
Если
 [Выбран 3-й уровень контроля]
то
 [Выполнить контроль информационных объектов]

Правило 20
Если
 [Выбран 3-й уровень контроля]

то
 [Выполнить формирование перечня маршрутов выполнения функциональных объектов]

...

Правило 39

Если
 [Выбран 2-й уровень контроля]
то
 [Выполнить контроль выполнения функциональных объектов]

Правило 40

Если
 [Выбран 2-й уровень контроля]
то
 [Выполнить сопоставление фактических маршрутов выполнения функциональных объектов и маршрутов, построенных в процессе проведения статического анализа]

...

Правило 57

Если
 [Выбран 1-й уровень контроля]
то
 [Выполнить контроль выполнения функциональных объектов]

Правило 58

Если
 [Выбран 1-й уровень контроля]
то
 [Выполнить сопоставление фактических маршрутов выполнения функциональных объектов и маршрутов, построенных в процессе проведения статического анализа]

Правило 59

Если
 [Выбран 1-й уровень контроля]
то
 [Оформить протокол испытаний]

В данном наборе продукционных правил фактом является значение выбранного уровня контроля. Требования, предъявляемые к выбранному уровню контроля, представлены в виде действий.

На рисунке 2 приведены значения скорости логического вывода в базе знаний ЭС из 59 правил до применения *аспектно-ориентированного подхода* (АО-подхода): в первом случае без объяснения решения, во втором – с объяснением решения. На оси у отмечено время выполнения поиска решения в микросекундах, на оси x отмечены номера проведенных тестов.

Для повышения эффективности процесса логического вывода в продукционных системах предлагается использовать АО-подход.

АО-подход – техника, позволяющая выявлять пересекающиеся (спутывающиеся) функциональные элементы и обеспечивать их консолидацию в ходе создания архитектуры и реализации системы. Под системой в данном случае понимается любая информационная система, через которую реализована некоторая предметная область в виде функций бизнес-логики.

Впервые данный подход был представлен в 1997 году Г. Кичалесом в работе [5].

В настоящее время сформировался подход AOSD (Aspect-oriented software development) – комплексный АО-подход к сбору требований, построению архитектуры и дизайну систем [6]. Данный подход формируют:

- аспектно-ориентированная инженерия требований;
- аспектно-ориентированное управление бизнес-процессами;
- аспектно-ориентированная системная архитектура;
- аспектно-ориентированное моделирование и дизайн;
- аспектно-ориентированное программирование (АОП).

АОП – наиболее известная область применения АО-подхода. Согласно данным Google Trends, представленным на рисунке 3, популярность данной техники в последние 5 лет неизменно находится на самом высоком уровне.

АО-подход как метод повышения эффективности процесса логического вывода в продукционных системах ранее не использовался.

При использовании АО-подхода в отдельные сущности, называемые аспектами, инкапсулируется сквозная функциональность.

Сквозная функциональность (Cross-cutting concern) – это функциональность, рассеянная по всему исходному коду ПО, систематически не зависящая от предметной области [7].

Для внедрения АО-подхода необходимо выполнить декомпозицию системы на модули, а затем с использованием концепций АО-подхода идентифицировать общность модулей, выделив пересечение сквозной функциональности в отдельные аспекты.

Основные концепции АО-подхода [8]:

- аспект (модульный элемент);
- совет (конструкция изменения поведения системы);
- точка соединения (точка вызова);
- срез (конструкция отбора точек соединения);
- цель;
- компоновщик (аспектный интегратор).

Взаимосвязь концепций АО-подхода в упрощенном виде представлена на рисунке 4.

Для связывания аспектов с системой предназначен компоновщик. Результатом связывания является целевой объект (цель). В общем случае целевой объект складывается из бизнес-логики системы и сквозной функциональности, представленной аспектом.

Совокупность правил вида *Условие* $\Rightarrow$ *Действие*, где *Условие* специфицирует разрез системы, а *Действие* задает фрагмент кода, активизируемый в системе при выполнении данного условия, по сути дела формирует аспект [9].

Рассмотрим АО-поход как метод повышения эффективности процесса логического вывода в продукционных системах.

Метод повышения эффективности процесса логического вывода в продукционных системах

Пусть $F = \{f_1, f_2, \dots, f_n\}$ – множество фактов в наборе правил ЭС, где $n \in N$ – общее количество фактов. Пусть $A = \{a_1, a_2, \dots, a_m\}$ – множество действий в наборе правил ЭС, где $m \in N$ – общее количество действий. Каждый антецедент включает известный факт из БД ЭС, следовательно, $Ant \subseteq F$, где Ant – множество антецедентов. Каждый консеквент включает известное действие из БД ЭС, следовательно, $Con \subseteq A$, где Con – множество консеквентов. Правило можно представить парой $R_k = \langle ant_k \in Ant, con_k \in Con \rangle$, где $k \in N$ – общее количество правил в базе знаний ЭС.

Аспектом (Aspect) Asp будем называть пересечение отдельных фактов в антецедентах или действий в консеквентах в наборе правил ЭС. Каждый аспект состоит из среза (Pointcut) фактов или действий. Тогда аспектом, являющимся описанием уровней контроля, будет $Asp_F \in F$, а аспектами, представляющими требования контроля состава и содержания документации, – $Asp_{A1} \in A$, контроля исходного состояния ПО – $Asp_{A2} \in A$, статического анализа исходных текстов программ – $Asp_{A3} \in A$, динамического анализа исходных текстов программ – $Asp_{A4} \in A$, требования к отчетности – $Asp_{A5} \in A$.

Заменяя в антецеденте и консеквенте конкретный факт или действие выделенным аспектом, получаем представление антецедента в виде $Ant \in Asp_F$, представление консеквента в виде $Con \subseteq \langle Asp_{A1} \vee Asp_{A2} \vee Asp_{A3} \vee Asp_{A4} \vee Asp_{A5} \rangle$. При этом избавляемся от повторяющихся фактов и действий.

Выявленные пересечения фактов или действий в наборе правил образуют аспект. Множество фактов или действий, выделенных в аспект, образуют срез. Аспект внедряется в правило через антецедент или консеквент. Роль совета в данном подходе играют факт или действие, которые будут заменены элементом аспекта. Правило, в котором факт или действие будет заменено элементом аспекта, обозначается целью. Интерпретатор правил выполняет роль компоновщика.

Представим результат анализа составляющих продукционных систем и основных концепций АО-похода:

аспект – выявленные пересечения фактов или действий в наборе правил;

срез – множество фактов или действий, принадлежащих аспекту;

точка соединения – место внедрения элементов аспекта, представленное антецедентом или консеквентом;

совет – факт или действие, которые будут заменены элементом аспекта;

цель – правило, в котором факт или действие будет заменено элементом аспекта;

компоновщик – интерпретатор правил.

Приведем фрагмент набора продукционных правил после применения АО-подхода:

Правило 1

Если

<Уровень контроля → «4»>

то

<Контроль состава и содержания документации → «Выполнить контроль состава и содержания документа Спецификация (ГОСТ 19.202-78)»>

Правило 2

Если

<Уровень контроля → «4»>

то

<Контроль состава и содержания документации → «Выполнить контроль состава и содержания документа Описание программы (ГОСТ 19.402-78)»>

Правило 3

Если

<Уровень контроля → «4»>

то

<Контроль состава и содержания документации → «Выполнить контроль состава и содержания документа Описание применения (ГОСТ 19.502-78)»>

...

Правило 19

Если

<Уровень контроля → «3»>

то

<Статический анализ исходных текстов программ → «Выполнить контроль информационных объектов»>

Правило 20

Если

<Уровень контроля → «3»>

то

<Статический анализ исходных текстов программ → «Выполнить формирование перечня маршрутов выполнения функциональных объектов»>

...

Правило 39

Если

<Уровень контроля → «2»>

то

<Динамический анализ исходных текстов программ → «Выполнить контроль выполнения функциональных объектов»>

Правило 40

Если

<Уровень контроля → «2»>

то

<Динамический анализ исходных текстов программ → «Выполнить сопоставление фактических маршрутов выполнения функциональных объектов и маршрутов, построенных в процессе проведения статического анализа»>

...

Правило 57

Если

<Уровень контроля → «1»>

то

<Динамический анализ исходных текстов программ → «Выполнить контроль выполнения функциональных объектов»>

Правило 58

Если

<Уровень контроля → «1»>

то

<Динамический анализ исходных текстов программ → «Выполнить сопоставление фактических маршрутов выполнения функциональных объектов и маршрутов, построенных в процессе проведения статического анализа»>

Правило 59

Если

<Уровень контроля → «1»>

то

<Отчетность → «Оформить протокол испытаний»>

Внедрение АО-похода не требует переработки структуры базы знаний и БД ЭС. В целях совместимости предлагается использовать АО-поход в интерпретаторе правил.

На рисунке 5 приведены значения скорости логического вывода в базе знаний ЭС после применения АО-подхода.

Как видим, скорость логического вывода в базе знаний ЭС после применения АО-подхода в обоих случаях увеличилась. Выделение аспектов позволило избавиться от непроизводительной проверки применимости правил.

Заключение

Представленная множеством правил продукции ЭС организует знания в виде логической связки *Факт (Посылка) ⇒ Действие (Следствие)*. Умение ЭС решать различные задачи (интерпретация данных, диагностика, мониторинг, проектирование, прогнозирование, планирование и обучение) позволило применять ЭС в различных отраслях промышленности.

В статье был рассмотрен актуальный недостаток производственных систем, который, согласно [4], выражается в более низкой эффективности процесса логического вывода по сравнению с другими моделями представления знаний.

Предложенный АО-подход в производственных системах позволил повысить скорость логического вывода в ЭС, а выделение аспектов – сократить количество операций при поиске решения и избавиться от перебора фактов и действий, что привело к повышению скорости логического вывода.

Рис. 5. Скорость поиска решения в базе знаний ЭС после АО-подхода

Fig. 5. The speed of finding solutions in the expert system knowledge base after AO approach

Литература

1. Минин А.Я. Информационные технологии в образовании. М.: Изд-во МПГУ, 2016. 148 с.
2. Корнеев И.К., Машурцев В.А. Информационные технологии в управлении. М.: ИНФРА-М, 2001. 158 с.
3. Джарратано Дж., Райли Г. Экспертные системы. Принципы разработки и программирование; [пер. с англ.]. М.: Вильямс, 2007. 1152 с.
4. Валетов В.А., Орлова А.А., Третьяков С.Д. Интеллектуальные технологии производства приборов и систем. СПб: Изд-во СПб ГУИТМО, 2008. 134 с.
5. Kiczales G., Lamping J., Mendhekar A., Maeda C., Lopes C., Loingtier J., Irwin J. Aspect-oriented programming. Proc. Europ. Conf. on Object-Oriented Programming (ECOOP), Jyväskylä, Finland, 1997. URL: <http://people.cs.ubc.ca/~gregor/papers/kiczales-ECOOP1997-AOP.pdf> (дата обращения: 23.04.2018).
6. Rashid A. Aspect-oriented software development in practice: tales from AOSD-Europe. 2010. URL: <https://pdfs.semanticscholar.org/08d0/1e2ebec9c9904376c92d092663c2e5467ed2.pdf> (дата обращения: 15.04.2018).
7. Гончаров А.А., Семенов Н.А. Аспектно-ориентированное программирование в контексте решения вопросов повышения эффективности экономических показателей ИТ-проектов // Программные продукты и системы. 2016. № 3. С. 149–153.
8. Laddad R., Johnson R. AspectJ in action: enterprise AOP with spring applications. Manning Publ., 2009, 568 p.
9. Сафонов В.О. Aspect.NET – инструмент аспектно-ориентированного программирования для разработки надежных и безопасных программ // Компьютерные инструменты в образовании. 2007. № 5. С. 3–13.

Improving logical inference speed of production expert systems using aspect-oriented approach

A.A. Goncharov¹, Postgraduate Student, andre_goncharov@hotmail.com

N.A. Semenov¹, Dr.Sc. (Engineering), Professor, is@tstu.tver.ru

¹ Tver State Technical University, Tver, 170026, Russian Federation

Abstract. The reason for active applying of expert systems in various industries is their ability to solve problems of data interpretation, diagnosis, monitoring, design, forecasting, planning and training.

Each expert system is based on a knowledge representation model. A production model, semantic networks and frames are the most common models. The production model is the one used most frequently.

The paper considers the important shortage of production systems related to low efficiency of the logical inference process compared to other knowledge representation models.

The paper describes the proposed method of increasing the efficiency of the logical inference process in production systems based on applying an aspect-oriented approach.

The aspect-oriented approach allows identifying intersecting functional elements and providing their consolidation during architecture creation and system implementation. This approach was first introduced in 1997 and it remains popular at the present time.

As the example, the article provides a set of production rules of expert system for selecting the requirements for a given level of control according to the requirements of guidance document of undocumented features. In this case, the facts are presented in the form of values of control levels and actions are presented as requirements to the selected level of control.

The proposed aspect-oriented approach to organizing production systems provided an opportunity of increasing the speed of logical inference in expert systems. The separation of intersecting facts and actions from the set of production rules into aspects has made it possible to reduce the number of operations when searching for a solution and to eliminate exhaustive search for facts and actions.

Keywords: aspect-oriented programming, aspect-oriented approach, expert system, production rule, artificial intelligence.

References

1. Minin A.Ya. *Information Technologies in Education*. Moscow, MPGU Publ., 2016, 44 p.
2. Korneev I.K., Mashurtsev V.A. *Informat Information Technologies in Management*. Moscow, INFRA-M Publ., 2001, 158 p.
3. Giarratano J.C., Riley G.D. *Expert Systems: Principles and Programming*. 4th ed. Course Technology Publ., 2004, 288 p. (Russ. ed.: Moscow, Williams Publ., 2007, 1152 p.).
4. Valetov V.A., Orlova A.A., Tretyakov S.D. *Intellectual Technologies for the Production of Devices and Systems*. St. Petersburg, SPb GUITMO Publ., 2008, 134 p.
5. Kiczales G., Lamping J., Mendhekar A., Maeda C., Lopes C., Loingtier J., Irwin J. Aspect-oriented programming. *Proc. Europ. Conf. on Object-Oriented Programming (ECOOP)*. Jyväskylä, Finland, 1997. Available at: <http://people.cs.ubc.ca/~gregor/papers/kiczales-ECOOP1997-AOP.pdf> (accessed April 23, 2018).
6. Rashid A. *Aspect-Oriented Software Development in Practice: Tales From AOSD-Europe*. 2010. Available at: <https://pdfs.semanticscholar.org/08d0/1e2ebec9c9904376c92d092663c2e5467ed2.pdf> (accessed April 15, 2018).
7. Goncharov A.A., Semenov N.A. Aspect-oriented programming as an approach to improve the efficiency of IT projects economic performance. *Software & Systems*. 2016, no. 3, pp. 149–153.
8. Laddad R., Johnson R. *AspectJ in Action: Enterprise AOP with Spring Applications*. Manning Publ., 2009, 568 p.
9. Safonov V.O. Aspect.NET as an aspect-oriented programming tool for the development of reliable and safe programs. *Computer Tools in Education*. 2007, no. 5, pp. 3–13 (in Russ.).