УДК 004.891

ФОРМАЛЬНЫЕ МОДЕЛИ СТРУКТУРНЫХ ОШИБОК В БАЗАХ ЗНАНИЙ ИНТЕЛЛЕКТУАЛЬНЫХ СИСТЕМ

¹Долинина О.Н., ¹Сучкова Н.К., ²Резчиков А.Ф.

¹ФГБОУ ВО «Саратовский государственный технический университет имени Гагарина Ю.А.»,
Саратов, e-mail: odolinina09@gmail.com, rinoa_27@mail.ru;

²Институт проблем точной механики и управления РАН, Саратов, e-mail: rw4cy@mail.ru

В статье описана первая стадия отладки интеллектуальных систем – статический анализ; обобщается информация о структурных ошибках в продукционных базах знаний интеллектуальных систем. Приводится классификация структурных ошибок и их формализация на основе теории графов. Выделяется 3 класса структурных ошибок: ошибки избыточности, ошибки неполноты и противоречия – и приводятся рекомендации по их устранению. Для каждого класса ошибок приводится детальный анализ наличия ошибочных продукций. Описываемый подход к отладке интеллектуальных систем относится к методам статического анализа, не требующим запуска системы на выполнение. Устранение данных ошибок позволяет привести продукционную базу знаний в состояние статической корректности. Этап статического анализа является необходимым этапом отладки баз знаний, но не гарантирует отсутствия ошибок в знаниях за счет противоречивости самой предметной области.

Ключевые слова: отладка интеллектуальных систем принятия решений, статический анализ, структурные ошибки, продукционные базы знаний

FORMAL MODELS OF STRUCTURAL ERRORS IN KNOWLEDGE BASES OF INTELLECTUAL DECISION MAKING SYSTEMS

¹Dolinina O.N., ¹Suchkova N.K., ²Rezchikov A.F.

¹Federal State Educational Institution of Higher Education Yuri Gagarin State Technical University
of Saratov, Saratov, e-mail: odolinina09@gmail.com, rinoa_27@mail.ru;

²Institute of Precision Mechanics and Control of Russian Academy of Sciences, Saratov, e-mail: rw4cy@mail.ru

The paper describes the first stage of the intellectual systems debugging – static analysis. The paper accumulates information on structural errors in the rule-based knowledge bases of intellectual systems. It provides classification of structural errors on the base of the graph theory. There are 3 types of structural errors identified: redundancy errors, incompleteness errors and inconsistency. There are suggested the ways of deleting of the errors in the knowledge bases. For each class of the error there is described the detailed description of the incorrect rules. The described approach to the debugging of the intellectual system is considered as the method of static analysis which does not demand the running of the system. Deleting of these types of errors allows to transfer the rule-based knowledge base to the state of the static correctness. The stage of static analysis is a necessary stage of the knowledge base debugging but does not guarantee the absence of the errors in knowledge because of the inconsistency of the knowledge field itself.

Keywords: debugging of the intellectual decision making systems, static analysis, structural errors, rule-based systems

Интеллектуальные системы (ИС) принятия решений находят применение во множестве областей: промышленности, медицине, научно-исследовательской деятельности, образовании. В настоящее время подобные системы всё чаще используются для решения задач в критических областях, что влечет за собой повышение требований к их надёжности. Особую роль при этом играет обеспечение качества баз знаний интеллектуальных систем, являющихся центральным звеном ИС, в то же время способы их отладки на сегодняшний день всё ещё не формализованы. В большинстве случаев отладкой занимаются эксперты, что влияет на увеличение стоимости разработки и не гарантирует отсутствия ошибок в базах знаний.

В литературе описаны различные подходы для отладки баз знаний проверки

корректности и полноты баз знаний [2–4]. Некоторые ошибки в знаниях могут быть выявлены только в процессе тестирования. К таким относятся ошибки, связанные с противоречивостью самой предметной области, например ошибки вида «забывание об исключении» [1]. В то же время ряд ошибок, связанных со структурой знаний, так называемые структурные ошибки, могут быть обнаружены и исправлены на стадии статической отладки базы знаний.

Отсутствие структурных ошибок не гарантирует отсутствия ошибок в знаниях, однако повышает эффективность принятия решений за счет приведения базы знаний в состояние статической корректности [1] и соответствующего уменьшения времени на принятие решения. Поэтому первым этапом отладки базы знаний должна быть статическая отладка.

В работах [7–11] приведены некоторые структурные ошибки в базах знаний и описаны алгоритмы их обнаружения, однако формализация ошибок имеется лишь частично. Среди наиболее известных автоматизированных средств верификации необходимо отметить программное обеспечение NuSMV, прошедшее в своем развитии несколько этапов, позволяющее выявлять ряд структурных ошибок для определенного класса продукционных баз знаний и имеющее в своем составе NuSMV язык программирования [6].

В данной работе обобщается информация о структурных ошибках и предлагаются формальные модели, которые могут быть использованы для автоматической верификации структур баз знаний.

Формализация структурных ошибок в продукционных базах знаний

Наиболее распространенный способ представления баз знаний интеллектуальных систем – продукция. Продукционную базу знаний можно представить в следующем виде:

$$P = (F, R, G, C, I), \quad (1)$$

где F – множество фактов о решаемой проблеме; R – множество правил вида

$$r_m: \text{если } f_i \text{ и } f_j \dots \text{ и } f_n \text{ то } f_k, \quad (2)$$

G – множество целей; C – множество разрешенных комбинаций фактов; I – интерпретатор правил, реализующий вывод.

Пусть S – множество входных фактов, т.е. фактов, устанавливаемых пользователем в интеллектуальной системе. $S \subset F$.

База знаний может быть представлена в виде И/ИЛИ графа. Например, пусть база знаний содержит следующие правила:

$$\begin{aligned} r_1: & \text{если } s_1 \text{ и } s_2, \text{ то } f_1; \\ r_2: & \text{если } s_2 \text{ и } s_3, \text{ то } f_2; \\ r_3: & \text{если } s_3 \text{ и } s_4 \text{ и } s_5, \text{ то } f_3; \\ r_4: & \text{если } f_1, \text{ то } g_1; \\ r_5: & \text{если } f_2 \text{ и } f_3, \text{ то } g_2. \end{aligned}$$

Тогда приведенным правилам соответствует И/ИЛИ граф на рис. 1.

Правило r_i вида

$$\text{если } f_{r_1} \text{ и } f_{r_2} \dots f_{r_n}, \text{ то } f_{r_m} \quad (3)$$

можно представить в виде пары:

$$r_i = (D_{r_i}; Q_{r_i}),$$

где $D_{r_i} = \{f_{r_1}, f_{r_2}, \dots, f_{r_n}\}$ и $Q_{r_i} = \{f_{r_m}\}$.

Как видно из представления, Q_{r_i} состоит из единственного элемента, далее обозначаемого как q_{r_i} .

Пусть L – множество цепочек вывода.

Рис. 1. И/ИЛИ граф базы знаний

Определение 1. Цепочка вывода l_i – последовательность правил $(r_{l_1}, r_{l_2}, \dots, r_{l_n})$ такая, что $\forall r_{l_k}, r_{l_{k+1}}, q_{r_{l_k}} \in D_{r_{l_{k+1}}}$ при $k = 2, \dots, (n - 1)$.

Тогда для графа, приведенного на рис. 1, $L = \{l_1, l_2, l_3, l_4, l_5, l_6, l_7, l_8\}$, где $l_1 = (r_1)$; $l_2 = (r_2)$; $l_3 = (r_3)$; $l_4 = (r_4)$; $l_5 = (r_5)$; $l_6 = (r_1, r_4)$; $l_7 = (r_2, r_5)$; $l_8 = (r_3, r_5)$.

Определение 2. Начало цепочки вывода l_i вида $(r_{l_1}, r_{l_2}, \dots, r_{l_n})$ – множество фактов в условии выполнения первого правила, $D_{l_i} = D_{r_{l_1}}$.

Определение 3. Конец цепочки вывода l_i вида $(r_{l_1}, r_{l_2}, \dots, r_{l_n})$ – следствие последнего правила, $Q_{l_i} = Q_{r_{l_n}}$.

Определение 4. Структурной ошибкой в продукционной базе знаний называется ошибка, обнаруживаемая в ходе анализа И/ИЛИ графа. Базы знаний, в которых отсутствуют структурные ошибки, являются статически корректными. Классификация структурных ошибок приведена на рис. 2.

Рассмотрим подробнее формальные модели ошибок класса «избыточность».

Определение 5. Правила r_i и r_j называются дубликатами, если $D_{r_i} \cap D_{r_j} \neq \emptyset$ и $q_{r_i} = q_{r_j}$.

Выделяют 3 вида дубликатов:

- включающие дубликаты;
- полные дубликаты;
- неполные дубликаты.

Определение 6. Правила r_i и r_j называются включающими дубликатами, если $D_{r_i} \subset D_{r_j}$, при этом $D_{r_i} \neq D_{r_j}$ и $q_{r_i} = q_{r_j}$. В этом случае правило r_i называется включаемым.

Для исправления подобной ошибки требуется удалить все правила-дубликаты, кроме включаемого.

Рис. 2. Классификация структурных ошибок в продукционных базах знаний

Определение 7. Правила r_i и r_j называются полными дубликатами, если $D_{r_i} = D_{r_j}$ и $q_{r_i} = q_{r_j}$.

Решением в данном случае является удаление всех дубликатов, кроме одного.

Определение 8. Правила r_i и r_j называются неполными дубликатами, $D_{r_i} \cap D_{r_j} \neq \emptyset$ и $q_{r_i} = q_{r_j}$, при этом $D_{r_i} \setminus D_{r_j} \neq \emptyset$ и $D_{r_j} \setminus D_{r_i} \neq \emptyset$.

Для исправления ошибки вида «неполные дубликаты» не существует универсального способа, в каждом конкретном случае необходимые изменения определяются экспертом.

Определение 9. Цепочка вывода l_i называется избыточной, если $q_{l_i} \in G$ и $\neg \exists l_j$ такой, что $q_{l_j} \in D_{l_i}$ и $q_{l_j} \in G$.

На рис. 3 приведен пример избыточной цепочки вывода $l_1 = (r_1, r_2)$:

r_1 : если s_1 и s_2 , то f_2 ;

r_2 : если f_1 и f_2 , то f_3 ;

$q_{l_1} = f_3$;

$f_3 \notin G$.

Рис. 3. Избыточная цепочка вывода

Избыточные цепочки вывода могут быть удалены из базы знаний.

Определение 10. l_i вида $(r_{i1}, r_{i2}, \dots, r_{in})$ называется незначащей цепочкой вывода, если $\forall r_{i,j}, |D_{r_{i,j}}| = 1$ при $j = 1, \dots, n$.

Выделяют 2 вида незначащих цепочек вывода:

- явные;
- неявные.

Определение 11. Незначащая цепочка вывода l_i называется явной, если $\exists r_j$ такое, что $r_j = (D_{l_i}; Q_{l_i})$.

Явные незначащие цепочки вывода могут быть удалены из базы знаний.

Определение 12. Незначащая цепочка вывода l_i называется неявной, если $\neg \exists r_j$ такое, что $r_j = (D_{l_i}; Q_{l_i})$.

Неявная незначащая цепочка вывода не является критичной ошибкой, однако допускает оптимизацию за счет сведения к предшествующему или последующему правилу.

Определение 13. Сведением к предшествующему правилу r_n незначащей цепочки вывода l_i при условии, что $q_{r_n} \in D_{l_i}$, называется создание правила r_m такого, что $r_m = (D_{r_n}; Q_{l_i})$, и удаление l_i и r_n из базы знаний.

Определение 14. Сведением к следующему правилу r_n незначащей цепочки вывода l_i при условии, что $q_{l_i} \in D_{r_n}$, называется создание правила r_m такого, что $r_m = (D_{r_n} - q_{l_i}) \cup D_{l_i}; Q_{r_n}$, и удаление l_i и r_n из базы знаний.

Различают 2 типа некорректных цепочек вывода:

- цепочки, избыточные по входу;
- цепочки, избыточные по выходу.

Определение 15. Цепочка вывода l_i называется избыточной по входу, если $D_{l_i} \cap G \neq \emptyset$.

Решением в данном случае является удаление избыточной по входу цепочки.

Определение 16. Цепочка вывода l_i называется избыточной по выходу, если $q_{l_i} \in S$.

Для исправления ошибки цепочку, избыточную по выходу, требуется удалить.

Определение 17. Цепочка вывода l_i называется циклом, если $q_{l_i} \in D_{l_i}$.

Цикл является критичной ошибкой, однако некорректное правило для удаления не может быть выбрано автоматически, поэтому решение по исправлению должно приниматься экспертом. Частным случаем цикла является простой цикл.

Определение 18. Правило r_i называется простым циклом, если $q_{r_i} \in D_{r_i}$.

Простой цикл должен быть удален из базы знаний.

Рассмотрим класс ошибок «неполнота».

К данному классу ошибок относятся изолированные вершины.

Определение 19. Вершина $f_i \in F \cup G$ называется изолированной, если $\neg \exists r_j$ такое, что $f_i \in D_{r_j}$ или $f_i \in Q_{r_j}$.

Рис. 4. Изолированная вершина

Вершина g_1 на рис. 4 является изолированной, т.к. нет правил, ведущих к ней.

Способ исправления ошибки зависит от типа вершины: если изолированная вершина – входной факт или цель, то необходимо добавить правила; в противном случае вершина может быть удалена из базы знаний.

Рассмотрим класс ошибок «противоречия».

Пусть C – множество разрешенных комбинаций фактов, тогда $c_i = \{f_{c,1}, f_{c,2}, \dots, f_{c,n}\}$. Тогда определим противоречивые цепочки вывода.

Определение 20. Цепочки вывода l_i и l_j называются противоречивыми, если $\exists f_k$ такой, что $f_k \in D_{l_i}$ и $f_k \in D_{l_j}$, и $\neg \exists c_m$ такая, что $q_{l_i} \in c_m$ и $q_{l_j} \in c_m$.

Рис. 5. Противоречивые цепочки вывода

На рис. 5 $l_1 = (r_1, r_2)$ и $l_2 = (r_3)$ являются противоречивыми:

- $D_{l_1} = \{f_1, f_2\}$;
- $q_{l_1} = \{f_7\}$;
- $D_{l_2} = \{f_2, f_3\}$;
- $q_{l_2} = \{f_6\}$;
- $D_{l_1} \cap D_{l_2} = f_2$;
- нет $c_k = (f_6, f_7)$.

Исправление ошибки вида противоречивые цепочки вывода должно производиться экспертом, либо за счет добав-

ления разрешенной комбинации фактов, либо за счет изменения или удаления правил. Отметим, что устранение описанных выше структурных ошибок приводит базу знаний в состояние статической корректности и является необходимым условием отладки, но не гарантирует ошибок иного рода, например, связанных с противоречивостью самой предметной области, обнаружение которых возможно с помощью тестирования [1, 5].

Заключение

В работе приведена классификация структурных ошибок в продукционных базах знаний интеллектуальных систем. Выделено 3 типа структурных ошибок: ошибки класса избыточности, ошибки неполноты и противоречия. К первому типу отнесены дубликаты, избыточные цепочки вывода, незначащие цепочки вывода, некорректные цепочки вывода и циклы. Ошибка вида «изолированные вершины» рассмотрена как ошибка класса неполноты. Противоречия в базах знаний представлены в работе противоречивыми цепочками вывода. Каждая рассмотренная ошибка формализована и снабжена описанием способа её исправления.

Формальная модель структурных ошибок может быть использована в качестве основы для проведения автоматической верификации структуры продукционных баз знаний на этапе статической отладки.

Список литературы

1. Долинина О.Н. Алгоритмы и методы разработки и отладки экспертных систем: монография. – Саратов: Саратовский гос. технический ун-т, 2015 – 225 с.
2. Рыбина Г.В., Смирнов В.В. Методы и алгоритмы верификации баз знаний в интегрированных экспертных системах // Известия РАИ. Теория и системы управления. – 2007. – № 4. – С. 91–102.
3. Cimatti A., Corvino R., Lazzaro A., Narasamdya I., Rizzo T., Roveri M., Sanseviero A., Tchaltev A.: Formal Verification and Validation of ERTMS Industrial Railway Train Spacing System. In Madhusudan, P., Seshia, S.A., eds.: CAV. Volume 7358 of LNCS., Springer (2012). P. 378–393.
4. Cragun B.J. Stendel H.J. A decision-table-based processor for checking completeness and consistency in rule-based expert systems // Int. J. Man-Mach. Stud. – 1987. – Vol. 26, № 5. – P. 633–648.
5. Dolinina O. Method of the Debugging of the Knowledge Bases of Intellectual Decision Making Systems / Advances in Intelligent Systems and Computing // Proc. Of the 5th Computer Science On-line Conf. 2016 (CSOC2016), 2016. – vol. 3. – P. 301–315.
6. Ferrante O., Benvenuti L., Mangeruca L., Sofronis C., Ferrari A.: Parallel NuSMV: A NuSMV Extension for the Verification of Complex Embedded Systems / Ortmeier, F., Daniel, P., eds.: SAFECOMPWorkshops. Volume 7613 of LNCS., Springer (2012). – P. 409–416.
7. Knauf R., Gonzalez A.J., Abel T. A framework for validation of rule-based systems / IEEE Trans Syst Man Cybern B Cybern. 2002; 32(3), P. 281–295.
8. Miller S.P., Whalen M.W., Cofer D.D. Software model checking takes off / Commun. ACM 53(2). (February 2010), P. 58–64.
9. Pira E., Reza Zand Miralvand M., and Soltani F. Verification of Conflicting and Unreachability in Rule-Based Expert Systems with Model Checking. URL: <https://arxiv.org/abs/1404.2768> (дата обращения: 10.04.2014).
10. Suwa H., Scott A.C., Shortliffe. An Approach to Verifying Consistency and Completeness in a Rule-Based Expert System // Rule-Based Expert Systems. – London: Addison – Wesley, 1984. – P. 159–170.
11. Xu D., Kejian Xia, Dezheng Zhang, Huangsheng Zhang Model Checking the Inconsistency and Circularity in Rule-Based Expert Systems / Computer and Information Science. – 2009. – vol. 2, № 1. – P.12–17.