УДК 004(063)

# МОДЕЛЬ ПРЕДСТАВЛЕНИЯ ПРОДУКЦИОННЫХ БАЗ ЗНАНИЙ НА ЭВМ

**А.С. Иванов**

Саратовский государственный университет,
кафедра математической кибернетики и компьютерных наук
E-mail: ac_ivanov@mail.ru

Предлагается модель представления продукционных баз знаний на ЭВМ в виде мультиграфа специального вида. Для этой модели разработан алгоритм проведения логического вывода. Приводятся экспериментальные данные временных затрат, подтверждающие эффективность предложенного алгоритма.

**One Model for Representation of Production Knowledge Base**

**A.S. Ivanov**

We offer a new approach to structure of production knowledge bases. In our approach the knowledge base is modeled by multigraph. We offer a deduction algorithm for such a model. The experimental results show the effectiveness of proposed methods.

В экспертных системах (ЭС) одной из основных моделей представления знаний является продукционная модель. Достоинства продукций отмечались многими авторами, такими как Д.А. Поспелов [1], Э.В. Попов [2], A. Newell [3] и др. Близость продукций к логическим импликациям позволяет легко реализовать логический вывод. Кроме того, продукционные эвристики близки стилю мышления человека-эксперта, знания которого хранятся в базе знаний (БЗ). Тем не менее при работе с продукционными БЗ приходится сталкиваться с синтаксическими и семантическими проблемами. Проверка реальной базы знаний, хранящей сотни и тысячи правил, на правильность заполнения, на полноту и непротиворечивость, структуризация базы, оценка времени, затрачиваемого на проведение логического вывода, представляет собой ряд непростых задач.

Продукционные правила имеют вид **ЕСЛИ — ТО**. В части «**ЕСЛИ**» описывается условие применимости правила, в части «**ТО**» — заключение, которое делается в случае применения правила.

Продукционная БЗ представляет собой набор продукционных правил, содержащий необходимые для проведения логического вывода знания. Обычно известные данные представляются в виде «объект – атрибут – значение». Под объектом понимается некоторый объект предметной области, под атрибутом — некоторая его характеристика. В дальнейшем в статье объект БЗ интерпретируется как «объект – атрибут» предметной области. Кроме того, каждый объект в БЗ снабжается списком разрешенных значений.

Логический вывод (ЛВ) представляет собой поиск ответа на вопрос пользователя, заданный экспертной системе. Этот ответ будет отыскиваться ЭС на основе анализа знаний, накопленных в БЗ к этому моменту. Ответ является одним из элементов списка разрешенных значений. Если используемая БЗ не дает возможности найти ответ на заданный вопрос, то информация об этом должна быть сообщена пользователю. Таким образом, результатом ЛВ всегда является либо конкретное значение из списка разрешенных значений, либо сообщение о невозможности получить ответ. Последнее формально можно рассматривать как некоторое специфическое значение (например, «неизвестно») и добавить его в список разрешенных значений. Присвоение объекту некоторого значения из расширенного списка разрешенных значений назовем *установлением значения*.

Логический вывод, реализуемый экспертной системой, проводится следующим образом. Вначале у пользователя запрашивается имя объекта, для которого ему необходимо установить значение, а также известные пользователю знания о предметной области, в дополнение к тем, что заложены в БЗ. Затем ЭС из знаний, хранящихся в БЗ, используя полученные от пользователя дополнительные сведения, выводит новые знания. Далее, используя предыдущие и вновь полученные знания, ЭС получает новые знания. Этот процесс продолжается аналогичным образом до тех пор, пока ЭС не сможет ответить на вопрос пользователя, т.е. присвоить объекту какое-либо из разрешенных значений или значение «неизвестно». Сеанс работы ЭС называется консультацией.

Отметим, что реальные базы знаний могут насчитывать сотни правил. Они хранятся в базе в хаотичном порядке, т.е. для нахождения нужного правила приходится просматривать всю БЗ, что снижает скорость работы продукционных экспертных систем. Очевидно, что если бы можно было упорядочить правила в БЗ таким образом, чтобы при установлении нового факта пришлось бы просматривать не всю базу знаний, а только часть ее, то время проведения ЛВ сократилось бы.

Ниже предлагается способ представления продукционной БЗ в виде мультиграфа специального вида, который позволит повысить эффективность работы ЭС.

Модель продукционной БЗ предлагается представить в виде конечного мультиграфа $G = (V, E)$, где $V$ — множество вершин, а $E$ — множество дуг. Множество $V$ является объединением двух непересекающихся множеств: множества О вершин-объектов и множества В вершин-ветвлений. Таким образом, $V = O \bigcup B$. Каждая вершина-объект $o_i$ соответствует конкретному объекту предметной области, для которой создана используемая БЗ.

Остановимся подробнее на том, каким образом продукционная БЗ будет представлена мультиграфом $G$. Пусть объект $o_1$ имеет разрешенные значения $o_{1,1}$, $o_{1,2}$, : $o_{1,n_1}$, $o_2$ — $o_{2,1}$, $o_{2,2}$, $o_{2,n_2}$, и так далее. В дальнейшем множество разрешенных значений для $o_k$ будем обозначать через *leg(ok)*.

Пусть продукция $P$ имеет вид

$$
\begin{array}{lll}
\textbf{ЕСЛИ} & o_1 = l_{1,i_1} & \textbf{И} \\
 & o_2 = l_{2,i_2} & \textbf{И} \\
 & \cdots & \\
 & o_k = l_{k,i_k} & \textbf{ИЛИ} \\
 & o'_1 = l'_{1,j_1} & \textbf{И} \\
 & \cdots & \\
 & o'_m = l'_{m,i_m} & \textbf{ИЛИ} \\
 & \cdots & \\
\textbf{ТО} & o_t = l_{t,i_t} & .
\end{array}
$$

Из структуры продукции видно, что она является объединением нескольких **ИЛИ**-компонент, каждая из которых представлена набором пар $(o_t, l_{t,i_t})$, соединенных логической связкой **И**. Каждый набор пар, составляющий **ИЛИ**-компоненту, далее интерпретируется как вершина-ветвление $b$. Множество всех вершин-ветвлений и есть упомянутое выше множество $B$.

Опишем теперь тип вершин мультиграфа $G$, составляющих множество О. Вершина-объект $o_k \in O$ является иерархической. Сама она считается вершиной нулевого уровня. В ее состав входит конечное число вершин 1-го уровня $l_{k,1}, l_{k,2}, \ldots, l_{k,n_k}$, взаимно однозначно соответствующих возможным значениям объекта $o_k$.

Приведенной выше продукции $P$ в мультиграфе $G = (V, E)$ будет соответствовать подграф, конструируемый следующим образом.

Вначале среди вершин множества О выбирается вершина $o_r$, которая фигурирует в части «**ТО**» продукции $P$. Из вершины 1-го уровня этой вершины-объекта $l_{r,i_r}$ проводятся дуги во все вершины-ветвления, порожденные продукцией $P$.

Пусть $b = \{(o_1 = l_{1,i_1}), \cdots, (o_k = l_{k,i_k})\}$ является одной из упомянутых вершин-ветвлений. Для каждой пары $(o_j = l_{j,i_j})$, где $j = 1, \cdots, k$, из этой вершины проводится дуга в вершину 1-го уровня $l_{j,i_j}$ вершины-объекта $o_j$. Если в объекте $o_j$ значение $l_{j,i_j}$ отсутствует, то это свидетельствует об ошибке в описании продукции $P$. Аналогичные построения проводятся для всех остальных вершин-ветвлений.

Для рассматриваемой в качестве примера продукции $P$ соответствующий ей подграф мультиграфа $G = (V, E)$ изображен на рис. 1.

Модель БЗ в целом представляет собой объединение всех подграфов описанного вида, каждый из которых соответствует некоторой продукции, входящей в состав БЗ.

Как видно из изложенного, в построенном подграфе имеется два типа дуг. К первому типу относятся дуги, ведущие из вершины-объекта в вершину-ветвление, ко второму — дуги, ведущие из вершины-ветвления в вершину-объект. Условимся дуги первого типа обозначать в виде $((o_k, l_{k,i_k}), b_m)$, где $o_k$ — вершина-объект, $l_{k,i_k}$ конкретное значение из множества *leg(*$o_k$*)*, $b_m$ — вершина-ветвление, являющаяся одним из **ИЛИ**-компонент в продукции $P$. Отметим, что начало этой дуги однозначно определяется $(o_k, l_{k,i_k})$. Далее, дуги второго типа условимся обозначать $(b_m, (o_k, l_{k,i_k}))$, где $b_m$ — вершина-ветвление, из которой исходит дуга, $o_k$ — вершина-объект и входящая в ее состав вершина 1-го уровня $l_{k,i_k}$, в которую дуга заходит. При этом должно удовлетворяться следующее условие: среди всех пар, составляющих вершину-ветвление $b_m$, есть пара $(o_k, l_{k,i_k})$.

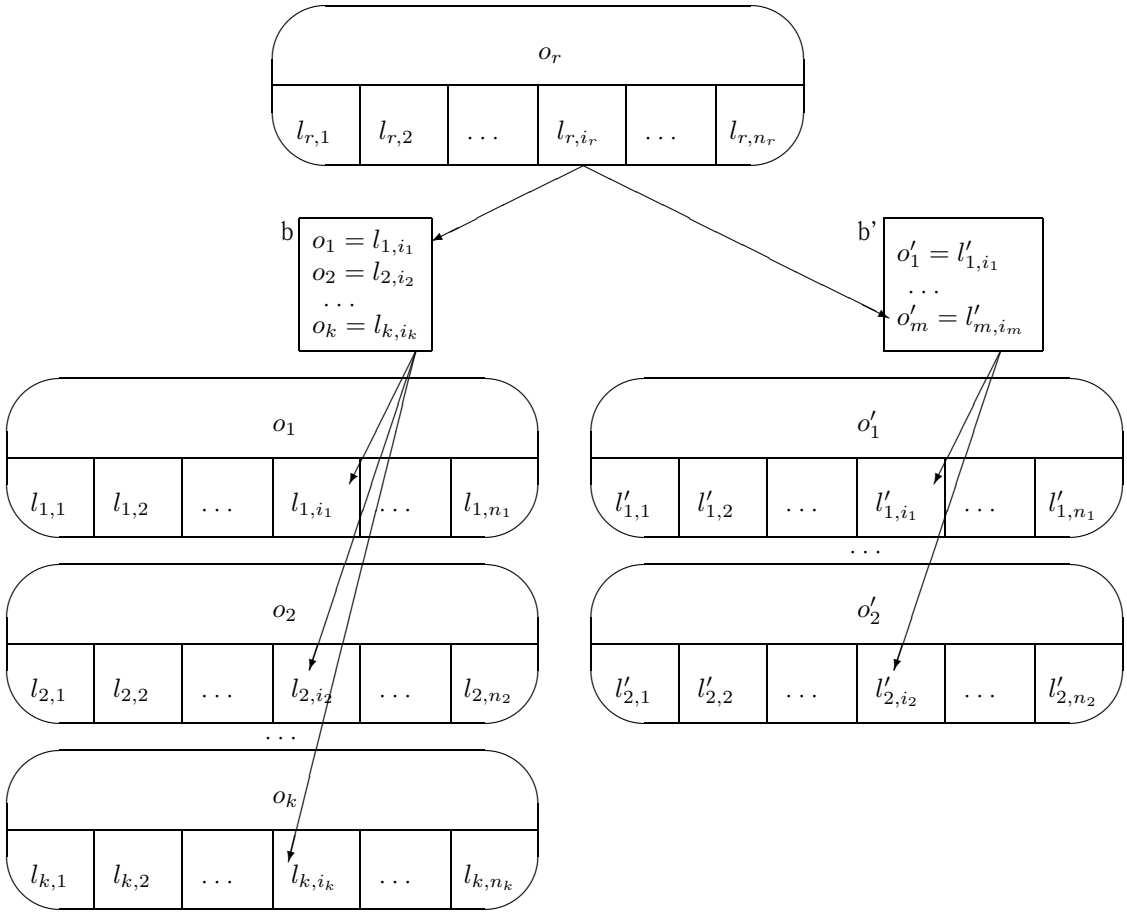


Рис.1. Представление продукции в виде графа

Введем следующие определения.

Вершину-объект о назовем невыводимой, если из нее не исходит дуг первого типа. Все остальные вершины-объекты будем считать выводимыми.

Каждой вершине $o_k$ поставим в соответствие множество $VL(o_k) \in leg(o_k)$, которое формируется в процессе проведения ЛВ и представляет собой множество установленных значений объекта $o_k$.

Отметим, что для выводимых вершин имеет место следующее утверждение:
$(l_{k,i_k} \in VL(o_k)) \longleftrightarrow \exists\ ((o_k, l_{k,i_k}), b_j)\ \forall\ (b_j, (o_p, l_{p,i_p}))(l_{p,i_p} \in VL(o_p))$.

Условимся, что для невыводимой вершины о множество ее значений $VL(\mathrm{o})$ должно быть запрошено у пользователя до начала ЛВ.

Для проведения логического вывода предлагается следующий рекурсивный алгоритм.

На вход подается стартовая вершина-объект.

1. Если вершина $\mathrm{o}_{start}$ невыводима и $VL(\mathrm{o}_{start})$ пусто, запросить значение $VL(\mathrm{o}_{start})$ у пользователя. Перейти к шагу 2.

2. Если $VL(\mathrm{o}_{start})$ непустое, то вернуть его в качестве ответа и завершить работу алгоритма, в противном случае перейти к шагу 3.

3. Если вершина выводима и $VL(\mathrm{o}_{start})$ пусто, тогда для $k = 1, \ldots, i_{start}$ и для всех $((\mathrm{o}_{start}, l_{start,k}), b_j)$ необходимо проверить выполнение условия $\forall\ (b_j, (o_p, l_{p,i_p}))(l_p, i_p \in VL(o_p))$. Если оно выполняется, то добавить $l_{start,k}$ в $VL(o_{start})$. Если в процессе проверки какое-либо множество $VL(o_p)$ пусто, то необходимо выполнить этот алгоритм с вершиной $o_p$ в качестве стартовой. После завершения перечисленных в этом пункте действий перейти к шагу 4.

4. Вернуть $VL(o_{start})$ в качестве ответа и завершить работу алгоритма.

После окончания работы алгоритма множества $VL(o)$ будут содержать в себе результаты проведенного вывода. Если для некоторой вершины-объекта о множество $VL(\mathrm{o})$ пусто, то это означает, что значение для данного объекта при проведении консультации установить не удалось.

Теперь рассмотрим вопрос о представлении предложенной модели в компьютере. На рис. 2 представлены основные элементы модели и отношения между ними.

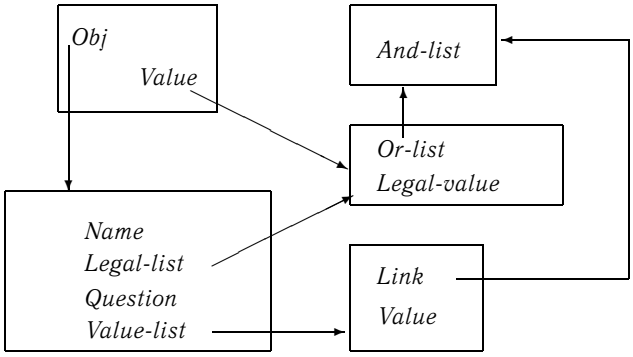


Рис. 2. Модель представления продукционной БЗ в виде списков

Множество всех объектов БЗ будем представлять в виде линейного односвязного списка *obj–list* вершин-объектов графа $G$. Каждый элемент списка взаимно однозначно соответствует вершине-объекту модели и включает в себя поля:

- *name* — название объекта;
- *question* — вопрос о значении объекта, задаваемый пользователю; — *legal–list* — ссылка на список разрешенных значений (соответствует *leg*);
- *value–list* — ссылка на список полученных в результате консультации значений (соответствует *VL*).

Перед началом консультации все списки *VL* должны быть пусты. Их заполнение будет происходить во время проведения ЛВ.

Список разрешенных значений состоит из двух составляющих:

- *legal–value* — разрешенного значения;
- *or–list* — списка ссылок на вершины-ветвления, в которые ведут дуги первого типа, начинающиеся в вершине первого уровня *legal–value* вершины–объекта *name*.

Вершина-ветвление на рис. 2 обозначена как *and–list* и представляет собой список пар вида [объект]=[значение], являющихся частью дуг второго типа, исходящих из данной вершины-ветвления. Каждая пара [объект]=[значение] из списка — это пара ссылок на соответствующую вершину-объект *obj* и соответствующее ей разрешенное значение *value*.

Очевидно, что на описанной выше модели проведение ЛВ будет осуществляться намного быстрее, чем в традиционном представлении БЗ. Причиной тому являются два обстоятельства:

1. Для установления значения объекта нет необходимости просматривать всю БЗ, поскольку каждый объект содержит ссылки на все продукции, в части «**ТО**» которых он фигурирует.

2. Происходит сокращение времени на проверку условий применимости продукции, поскольку это сводится всего лишь к установлению факта наличия требуемого значения в списке *value–list* нужного объекта. Заметим, что в традиционной продукционной БЗ это происходит путем сравнения каждого условия продукции с каждым значением, полученным в процессе консультации.

Теперь приведенный выше рекурсивный алгоритм логического вывода можно переформулировать следующим образом.

На входе задается имя объекта, значение которого необходимо установить.

1. Если в процессе работы этот объект уже получил какое-то значение, т.е. список *value–list* этого объекта не пуст, полученное значение (или значения, если их несколько) возвращается в качестве ответа и алгоритм завершается. В противном случае осуществляется переход к шагу 2.

2. Используя список разрешенных значений данного объекта, производится обращение к объектам, значения которых необходимы для проведения логического вывода. Далее этот алгоритм применяется к каждому из указанных объектов. Если для какого-либо разрешенного значения хотя бы для одного элемента списка *or–list* для каждой пары [объект]=[значение] списка *and–list* соответствие установлено, данное разрешенное значение считается установленным и тогда выполняется переход к шагу 3 алгоритма.

3. Полученное значение (или значения, если их несколько) возвращается в качестве ответа и одновременно присоединяются к списку *value–list*, после чего алгоритм завершается. При этом в список *value–list* заносится не только найденное значение, но и ссылка на правило, которое при этом использовалось. Заметим, что эта информация необходима для функционирования подсистемы объяснения.

Оценим временную и емкостную сложность приведенного алгоритма.

Пусть $n$ — количество вершин-объектов графа, т.е. количество объектов в предметной области, а $t$ — количество вершин-ветвлений, т.е. правил в БЗ. Заметим, что данный алгоритм исполняется только для тех вершин-объектов, которые еще не участвовали в логическом выводе. Из этого следует, что каждую вершину-объект алгоритм обрабатывает не более одного раза. Для вершин-объектов, чьи значения непосредственно или опосредованно не влияют на значения устанавливаемого объекта, алгоритм не выполнится ни разу. Число же операций шага 2 алгоритма будет порядка $t$, так как каждое правило используется не более одного раза. Все это дает общую сложность алгоритма $O(n+t)$.

Теперь перейдем к оценке емкостной сложности. Основу предложенной модели составляют ссылки. Каждая ссылка занимает фиксированный объем памяти. Поэтому размер модели напрямую зависит от общего числа ссылок в ней.

Произведем оценку количества ссылок, необходимых для введения нового правила. Каждая пара «[объект]=[значение]» из части «**ЕСЛИ**» правила требует по одной ссылке соответственно на объект и его значение и еще одну для создания списка *and–list*. Для такой же пары из части «**ТО**» потребуется только две ссылки. Одна — для элемента в списке *or–list*, вторая — на *and–list*. Таким образом, количество добавленных ссылок зависит только от количества пар «[объект]=[значение]» в частях «**ЕСЛИ**» и «**ТО**» правила и не зависит от количества правил в БЗ.

Легко видеть, что число ссылок во всей БЗ, если имеется $n$ объектов, $m$ — общее количество разрешенных значений для всех объектов, $l$ пар в части «**ЕСЛИ**» и $k$ пар в части «**ТО**», равно $O(n+m+l+k)$.

Очевидным недостатком предлагаемой модели является замедление процесса добавления новых правил в БЗ. Так, чтобы добавить новое правило в рамках данной модели необходимо для каждой пары [объект]=[значение] из частей «**ЕСЛИ**» и «**ТО**» просматривать в поисках нужного правила всю модель. Данное обстоятельство приводит к увеличению загрузки БЗ в оперативную память, но не перечеркивает преимуществ предложенной модели.

Это вытекает из следующих соображений. Широкое распространение баз данных привело к тому, что для них понадобилось проектировать не только специальное программное обеспечение, как, например СУБД, но и аппаратное обеспечение. Подобное ожидает и экспертные системы. Их дальнейшее распространение потребует разработки такого аппаратно-программного обеспечения, которое смогло бы повысить производительность этих систем по сравнению с работающими на обычных компьютерах. Это, в свою очередь, потребует разработки специальных моделей представления знаний или модификации уже имеющихся, усовершенствования механизмов логического вывода и поиска в БЗ. В этом случае поскольку необходимость добавления новых знаний в БЗ возникает значительно реже, чем реализация ЛВ, увеличение скорости проведения консультаций даст существенное увеличение общей эффективности работы ЭС, несмотря на неизбежное при этом замедление процесса пополнения и корректировки БЗ.

В табице приведены экспериментальные данные, отражающее время, затраченное на работу рассматриваемых в статье алгоритмов. Эксперимент проводился на процессоре Pentium-166 с помощью системной функции GetTickCount.

Зависимость времени работы рассмотренных алгоритмов от размеров БЗ

| Кол-во правил в базе знаний | Кол-во объектов в базе знаний | Время, затраченное на создание модели | Время проведения ЛВ на предложенной модели | Время проведения ЛВ без использования предложенной модели | Отношение времени проведения ЛВ на модели ко времени без ее использования |
|---|---|---|---|---|---|
| 5621 | 1728 | 10420 | 17 | 12420 | 731 |
| 2805 | 864 | 3807 | 11 | 2802 | 255 |
| 1397 | 432 | 1560 | 5 | 613 | 123 |
| 693 | 216 | 685 | 2 | 135 | 68 |
| 341 | 108 | 165 | 1 | 33 | 33 |
| 165 | 54 | 70 | 1 | 9 | 9 |

Время, указанное в таблице, исчисляется в тиках процессорного времени. Для сравнения приведем три характеристики:

- время, затраченное на конвертирование базы знаний из текстового файла в предложенную модель;
- время, затраченное на проведение ЛВ на предложенной модели;
- время, затраченное на проведение ЛВ без использования предложенной модели.

Поскольку время может колебаться в зависимости от вводимых значений, алгоритм выполнялся многократно при различных входных данных. В таблице приводится среднее время. В последнем столбце приведено отношение времени проведения ЛВ на предложенной модели ко времени такого же ЛВ, но без ее использования.

На рис. 3–5 приведены графики зависимости времени, затраченного на выполнение трех описанных выше алгоритмов, от количества правил в БЗ, где на горизонтальной оси откладывается количество продукционных правил в БЗ, на вертикальной — время, затраченное на выполнение соответствующего алгоритма.

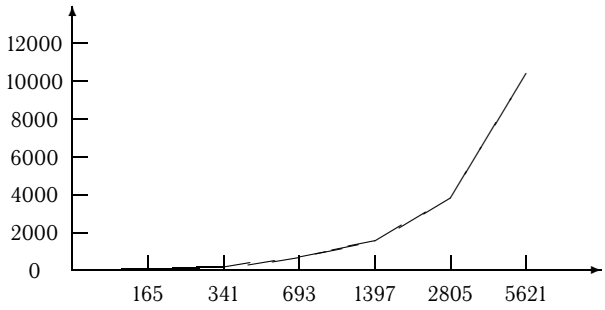


Рис. 3. Зависимость времени, затраченного на представление БЗ в виде предложенной модели от количества правил в БЗ

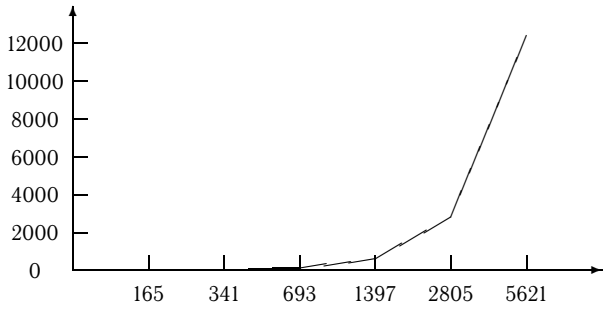


Рис. 4. Зависимость времени, затраченного на проведения одного сеанса ЛВ без предложенной модели от количества правил в БЗ

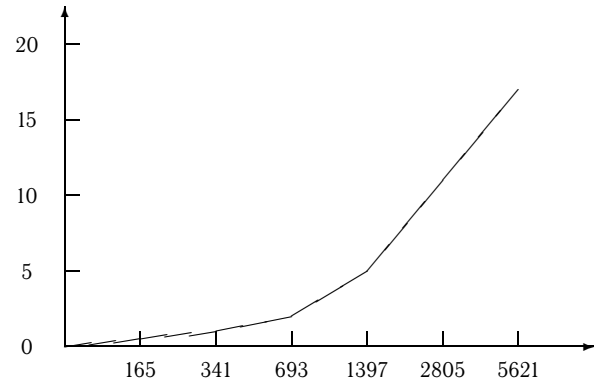


Рис. 5. Зависимость времени, затраченного на проведения одного сеанса ЛВ на предложенной модели от количества правил в БЗ

Таким образом, приведенные экспериментальные данные показывают, что время, затрачиваемое на построение модели, имеет тот же порядок, что и время, необходимое для проведения логического вывода без использования предложенной модели. Эти данные также наглядно демонстрируют, что время проведения ЛВ на предложенной модели значительно ниже, чем без ее использования. Отсюда вытекает, что применение такой модели представления продукционной БЗ на ЭВМ позволит существенно повысить эффективность логического вывода.

**Библиографический список**

1. *Поспелов Д.А.* Продукционные модели // Искусственный интеллект. М.: Радио и связь, 1990. Кн. 2.
2. *Попов Э.В.* Экспертные системы. М.: Наука, 1987.
3. *Newell A.* Producton systems: models of control structures. N.Y.: Acadmik press, 1973.