Министерство образования и науки Российской Федерации

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ Н. Г. ЧЕРНЫШЕВСКОГО»

Кафедра математической
кибернетики и компьютерных наук

РЕАЛИЗАЦИЯ И АНАЛИЗ ЭФФЕКТИВНОСТИ МЕХАНИЗМОВ ЛОГИЧЕСКОГО ВЫВОДА В ЭКСПЕРТНОЙ СИСТЕМЕ

АВТОРЕФЕРАТ БАКАЛАВРСКОЙ РАБОТЫ

студента 4 курса 451 группы
направления 09.03.04 — Программная инженерия
факультета КНиИТ
Емельянова Михаила Алексеевича

Научный руководитель
доцент, к. ф.-м. н. _____ А. С. Иванов

Заведующий кафедрой
к. ф.-м. н. _____ С. В. Миронов

Саратов 2018

ВВЕДЕНИЕ

На сегодняшний день термин «экспертные системы» широко распространен и известен широкому кругу лиц, имеющим интересы в сфере информационных технологий. Экспертная система является представителем программных продуктов из области исследований искусственного интеллекта.

В настоящее время системы, основанные на знаниях, становятся распространенным коммерческим продуктом на рынке информационных технологий. Многие компании, включая «Pegasystems», «Progress Software», «FICO» и «IBM», занимаются разработкой экспертных систем и оптимизацией их работы. [1–4] Таким образом, актуальной является задача увеличения производительности экспертных систем.

Цель бакалаврской работы — реализация механизмов логического вывода и оценка их эффективности на примере экспертной системы продукционного типа.

Поставленная цель определила **следующие задачи:**

1. Реализовать механизм логического вывода, основанный на «наивном» алгоритме.
2. Реализовать механизм логического вывода, основанный на Rete сети.
3. Реализовать механизм логического вывода, основанный на миварной сети.
4. Реализовать экспертную систему продукционного типа.
5. Создать проверочные наборы входных данных и баз знаний.
6. Проверить корректность работы приложения.
7. Экспериментальным путем выяснить эффективность работы механизмов логического вывода на примере созданной экспертной системы и проверочных данных.

Методологические основы разработки экспертных систем представлены в работах К. Т. Леондеса [5], Д. А. Ватермана [6], Ф. Хайеса-Рота [7], П. Джексона [8], Д. И. Муромцева [9], А. П. Частикова [10], О. В. Германа [11], Э. В. Поспелова [12].

Структура и объем работы. Бакалаврская работа состоит из введения, 3-х разделов, заключения, списка использованных источников и 1-го приложения. Общий объем работы — 71 страница, из них 40 страниц — основное содержание, включая 21 рисунок и 1 таблицу, список использованных источ-

ников информации — 22 наименования.

Первый раздел **«Принципы функционирования экспертных систем»** посвящен описанию принципов функционирования экспертных систем в целом. В нем дается описание основных понятий, рассматривается структура экспертных систем, модели представления знаний, механизм логического вывода, а также обозначается проблема производительности.

Второй раздел **«Описание разработанного приложения»** посвящен реализации приложения и предложенных механизмов логического вывода. В нем дано полное описание реализованных структур, проверяется корректность их работы.

Третий раздел **«Оценка быстродействия интерпретаторов»** посвящен экспериментальным запускам приложения и вычислению показателей эффективности. Представлены графики, отображающие полученные зависимости.

1 Принципы функционирования экспертных систем

1.1 Экспертная система

Экспертная система — это программная система, знания и умения которой сравнимы с умениями и знаниями специалистов в какой-нибудь специальной области знаний. В качестве источника информации, которая определяет принятие решений, используют знания, представимые в некоторой форме. Общие принципы разработки экспертных систем описаны как в иностранной литературе [5–8], так и в отечественной [9–12].

1.2 Структура экспертных систем

Типичная статическая экспертная система состоит из следующих основных компонентов:

— рабочей памяти, называемой также базой данных (БД);
— базы знаний (БЗ) или базы правил;
— интерпретатора;
— интерфейса взаимодействия с экспертной системой.

База данных, также называемая рабочей памятью системы, предназначена для хранения начальных и промежуточных данных решаемой в текущий момент задачи.

База знаний в экспертной системе предназначена для хранения правил, описывающих целесообразные преобразования данных этой области.

Интерпретатор использует начальные данные из рабочей памяти и знания из базы знаний для формирования такой последовательности продукций, применение которых к начальным формирует последовательность правил, которые, будучи примененными к исходным данным, приводят к решению задачи. [13]

1.3 Модели представления знаний

Под моделью представления знаний понимают особый способ организации информации, решающий следующие задачи:

— определение состава представляемых знаний;
— определение структуры представляемых знаний;
— определение способа представить знания в выбранном формализме. [14]

С точки зрения представления знаний существуют два основных типа моделей:

— эмпирические модели — основанные на моделировании механизмов решения задач человеком,
— теоретические модели — гарантирующие правильность решений.

Рассмотрим подробнее три наиболее распространенные модели представления знаний, используемых в экспертных системах:

— продукционные модели — модели основанные на правилах, позволяют представить знание в виде предложений типа: «ЕСЛИ условие, ТО действие». Продукционные модели обладает тем недостатком, что при накоплении достаточно большого числа правил, они начинают противоречить друг другу;
— сетевые модели или семантические сети — как правило, это граф, отображающий смысл целостного образа. Узлы графа соответствуют понятиям и объектам, а дуги — отношениям между объектами;
— фреймовые модели — основывается на таком понятии как фрейм (англ. frame — рамка, каркас). Фрейм — структура данных для представления некоторого концептуального объекта. Информация, относящаяся к фрейму, содержится в составляющих его слотах. Слоты могут быть терминальными либо являться сами фреймами, т. о. образуя целую иерархическую сеть. [15]

Реализованная экспертная система использует в качестве модели представления знаний продукционную.

1.4 Механизм логического вывода

Логический вывод в продукционной системе реализуется инетрпретатором, формирующим результат из базы данных (входных данных), согласно правилам. Такой интерпретатор также можно назвать механизмом или машиной логического вывода.

Существует два способа организации порядка вывода:

— прямой вывод;
— обратный вывод.

Прямой порядок — от фактов к заключениям. В экспертных системах с прямыми выводами по известным фактам отыскивается заключение, которое из этих фактов следует. Если такое заключение удается найти, оно заносится в рабочую память. Обратный порядок вывода — от заключений к фактам. В системах с обратным выводом вначале выдвигается некоторая гипотеза о ко-

нечном суждении, а затем механизм вывода пытается найти в рабочей памяти факты, которые могли бы подтвердить или опровергнуть выдвинутую гипотезу. Обратные выводы управляются целями. [16]

В реализованных механизмах логического вывода используется прямой порядок вывода.

1.5 Проблема производительности

Процесс логического вывода данных из базы знаний связан с сопоставлением правил с известными фактами в рабочей памяти и формированием конфликтного набора. Этап сопоставления требует проведения значительного объема операций, т. к. для конфликтного набора следует проверить все условия активных правил на всех сочетаниях активных элементов рабочей памяти.

Неэффективность простейших алгоритмов сопоставления является следствием того, что в каждом цикле работы интерпретатора для получения конфликтного набора заново производится просмотр рабочей памяти и массива правил.

В качестве альтернативы в данной работе предполагается использовать алгоритм Rete и сеть, формируемую с его помощью. Данный механизм предлагает создание специальной структуры, позволяющей проводить поиск решений в ней, основываясь на фактах в рабочей памяти.

Другой альтернативой является использование миварной сети, структурно являющейся ориентированным двудольным графом. Такая структура также помогает унифицировать представление знаний и упростить модификацию базы знаний добавлением новых продукций или удалением имеющихся.

2 Описание разработанного приложения

2.1 Программные средства реализации экспертной системы

Экспертная система реализована на языке программирования python 3.6. В качестве среды разработки была использована свободно распространяемая интегрированная среда разработки PyCharm. Также в процессе создания приложения была использована система контроля версий Git.

2.2 Функционал экспертной системы

В разработанной экспертной системе реализовано:

— обработка входных файлов правил и фактов формата txt;
— механизм логического вывода, основанный на «наивном» алгоритме;
— механизм логического вывода, основанный на Rete сети;
— механизм логического вывода, основанный на миварной сети;
— графический интерфейс.

2.3 Обработка входных файлов

Входные данные в экспертную систему загружаются из файлов формата txt. Входные факты представимы в виде « факт1 AND факт2 ». Допустимо использовать выражения присваивания с помощью зарезервированного символа « = », в правой части которого должно быть число. Продукции представимы в виде выражений « IF антецедент THEN консеквент. ».

Антецедент правила представим в виде условий для отдельных атрибутов, соединенных зарезервированным словом «AND», являющимся логическим И. В случае, если правила должны содержать логическое ИЛИ, то такие продукции на этапе разработки базы знаний с помощью декомпозиции преобразуют в два и более правил без использования логического ИЛИ. Для условий атрибутов допустимы простая проверка истинности факта, возможно, с использованием логического отрицания в виде зарезервированного слова «NOT» перед фактом. Также допустима проверка численного значения в форме « атрибут == число ». В качестве бинарных операторов допускается использовать зарезервированные символы: « > », « < », « >= », « <= », « == », « != ».

Консеквент правила выражается в виде утверждениях истинности фактов, соединенных зарезервированным словом «AND», возможно, с применением логического отрицания с помощью зарезервированного слова «NOT» перед

фактом, в форме « факт1 AND NOT факт2 ». Допускается присваивание численных значений с помощью зарезервированного символа « = », представимое в форме « факт = число ». В качестве правой части выражения присваивания допустимо использовать арифметические выражения в виде « факт1 = факт2 * 4 », возможно, с применением имен атрибутов, имеющих численные значения, используя зарезервированные символы « + », « - », « * », « / ».

Для последующей обработки и хранения промежуточной формы базы правил перед преобразованием в структуру соответствующего механизма логического вывода используется файл формата `json`. В случае повторного запуска программы этот файл позволяет программе автоматически загружать использованную в предыдущем сеансе базу знаний, пока не будет перезаписан загрузкой новых правил из файла формата `txt`.

За обработку входных файлов отвечает модуль `filehandler.py`. Был использован подключаемый модуль `nltk`, позволяющий обрабатывать строковые переменные, описанные на естественном языке. [17]

2.4 Механизм логического вывода, основанный на «наивном» алгоритме

«Наивный» алгоритм подразумевает проверку применимости каждой продукции к каждому факту из рабочей памяти. Механизм логического вывода в данном случае преобразовывает конечную форму внутреннего представления правил в список правил, оптимально использующих проверку антецедентов и применение консеквентов. Используемая динамическая структура: список, элементами которого являются словари.

Логический вывод заключается в проверке истинности антецедента, согласно имеющимся фактам в рабочей памяти. Применимые правила составляют список, называемый «повесткой дня». Выбранные продукции обрабатываются в порядке их поступления. Если правило все еще применимо — оно удаляется из базы знаний и активируется его консеквент. Таким образом реализован механизм разрешения конфликтов. На следующей итерации действия повторяются, пока не останется применимых правил.

Реализованный механизм логического вывода, основанный на «наивном» алгоритме представлен модулем `naive.py`.

2.5 Механизм логического вывода, основанный на Rete сети

Rete алгоритм можно описать как схему индексирования, которая не требует шагов интерпретации. Индексирующая функция представляется в виде сети простых распознавателей. Это представление относится к графам представлений так называемых структурированных шаблонов. [18]

Алгоритм подразумевает использование заранее выделенной памяти, организованной специальным образом соответствуя правилам вывода, чтобы ускорить процесс сопоставления входных данных и антецедентов продукций. Составить Rete сеть достаточно один раз при инициализации базы правил. [19]

Сеть можно разделить на две основные части: альфа-сеть и бета-сеть. Альфа-сеть представляет собой узлы, характерищующие части антецедентов правил. Бета-сеть является набором узлов, собирающих посылки правил и передающих управление узлам-исполнителям. Таким образом, производится декомпозиция продукций на элементы, которые находят представление в виде узлов Rete сети. Весь путь от начальных узлов-объектов до узла-исполнителя является правилом. С помощью данной структуры логический вывод производится путем изменения значений в узлах-объектах согласно входным фактам, которое вызывает проверку у смежных узлов. При активации узла-исполнителя выполняются некоторые действия, описанные внутри него, и правило считается выполненным.

Применимость Rete алгоритма можно считать недооцененной. Скорее всего, причиной является его наиболее частое использование в продукционных системах. Алгоритм сопоставления с шаблоном, такой как Rete, можно использовать в качестве «черного ящика» — модуля, служащего частью большей системы. Естественно, что характеристики и ограничения систем в целом не обязательно навязаны используемым алгоритмом, но могут возникнуть из других соображений. Например, на данный момент большинство систем, которые используют Rete алгоритм, являются продукционными. Это может накладывать ограничения в случае использования какого-либо механизма разрешения конфликтов. На каждом цикле Rete сеть используется для нахождения конфликтного набора правил, но после прерывается для определения последовательности их выполнения. [20]

Реализованный механизм логического вывода, основанный на Rete сети, представлен модулем `rete.py`. В модуле реализованы классы узлов, представ-

ляющих собой элементы сети и главный класс сети.

2.6 Механизм логического вывода, основанный на миварной сети

Основателем миварного подхода в области создания искусственного интеллекта является доктор технических наук Варламов Олег Олегович. В настоящее время занимает должность директором научно-исследовательского института в ООО «МИВАР», который продолжает исследования в этой области.

Миварный подход объединяет две основные технологии: накопления информации и ее обработки. Мивар заявлен разработчиками как технология, предназначенная для хранения информации с возможным эволюционным изменением структуры, а также обработки информации для логического вывода. Миварные сети могут быть представлены в виде двудольного графа, состоящего из объектов-переменных и правил-процедур. Для этого, прежде всего, составляются два списка, которые и образуют две непересекающиеся доли графа: список объектов и список правил. Таким образом, в продукционной модели представления знаний используются элементы объектно-ориентированных баз данных, позволяющие упростить структуру представления. [21]

Миварная сеть также представима в виде матрицы смежности вершин из множества объектов и вершин из множества правил, в строках которой обозначены правила, в столбцах — объекты, а в ячейках отмечают наличие аргумента и выходного результата для каждой из продукций.

Построение сети начинается с выделения всех возможных объектов с их атрибутами и заполнения множества объектов и множества правил. После того, как сформировались все доступные вершины, производят разбор правил для выявления смежности с объектами. Механизм логического вывода, основанный на миварной сети, при прямом выводе предполагает процесс последовательного изменения атрибутов узлов-объектов, вызывающие цепную реакцию у подчиненных узлов.

Реализованный механизм логического вывода, основанный на миварной сети представлен модулем `mivar.py`. В модуле реализованы классы узлов миварной сети, а также класс, реализующий саму миварную сеть.

2.7 Графический интерфейс

Графический интерфейс разработанной экспертной системы представлен оконным приложением, на главном экране которого доступны интерактивные

элементы:

— текстовое поле вывода событий системы;
— кнопки загрузки файлов;
— выпадающий список доступных механизмов логического вывода;
— кнопки реализации логического вывода и отображения результатов.

Эти элементы реализуют следующие функции:

— загрузка начальных значений фактов в рабочую память;
— загрузка базы знаний для преобразования в соответствующие структуры;
— выбор механизма логического вывода, который будет использован интерпретатором для получения результата вычислений;
— инициализация процесса логического вывода из базы знаний с применением выбранного механизма;
— вызов вспомогательного окна, отображающего результат вычислений.

При создании окон был использован инструментарий модуля `tkinter`, подробно описанный в официальной документации [22]. Реализованный графический интерфейс представлен модулем `UI.py`.

2.8 Проверка корректности логического вывода

2.8.1 Работа с логическими переменными

Корректность работы интерпретатора была подтверждена тестовыми запусками на простой выборке правил, содержащих только логические переменные в антецедентах и консеквентах.

2.8.2 Работа с простыми численными выражениями

Также правильность вывода была проверена на простой выборке правил, содержащих численные переменные и простые арифметические выражения в антецедентах и консеквентах.

3 Оценка быстродействия интерпретаторов

Для оценки быстродействия механизмов логического вывода фиксировалось время, затраченное на получение результата. Характеристики системы, на базе которой проходило тестирование:

— Операционная система: Windows 8.1.
— Процессор: AMD FX(tm)-8350 Eight-Core Processor 4.00 GHz.
— Объем оперативного запоминающего устройства: 16.0 ГБ.
— Тип системы: 64-разрядная операционная система, процессор x64.

Для проверок были искусственно созданы базы знаний объемом 10 - 30000 правил. Полученные усредненные результаты экспериментальных запусков представлены на рисунке 1.

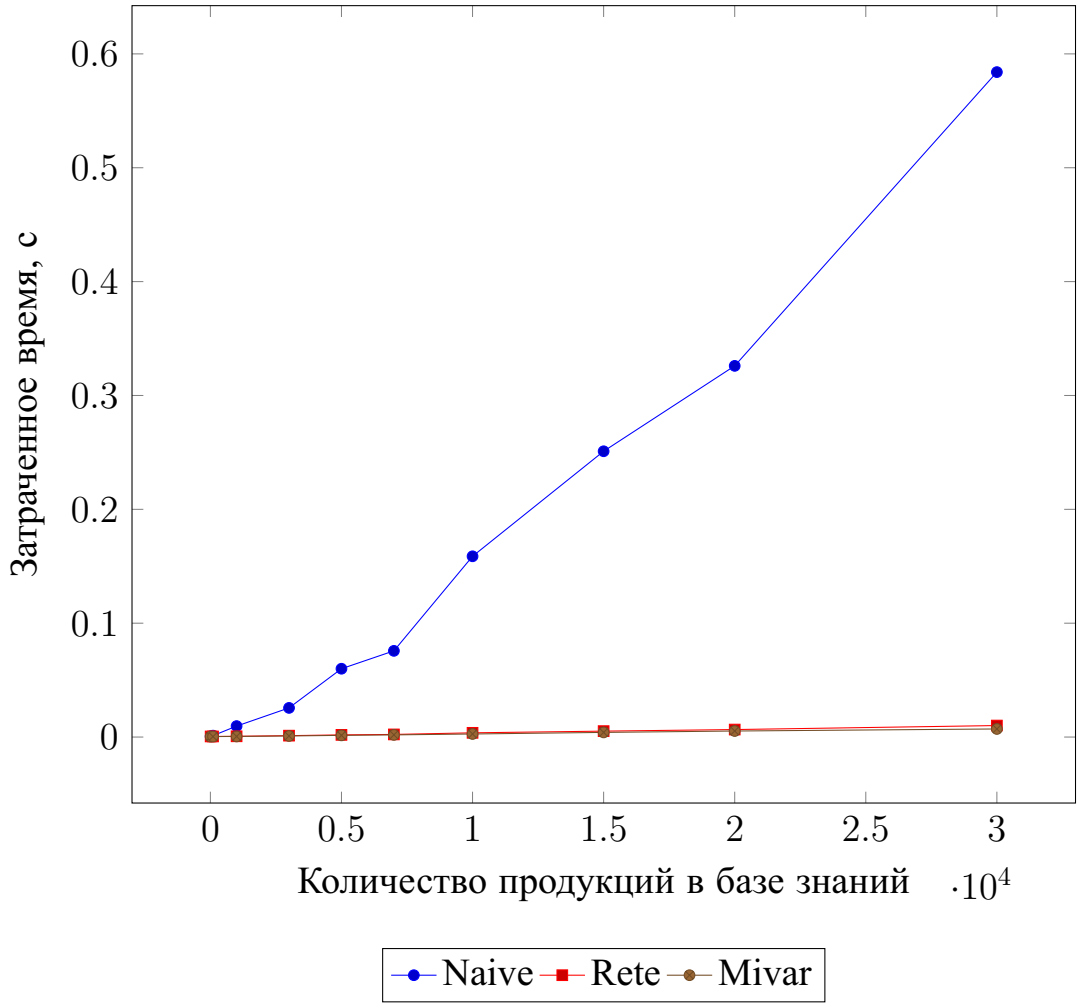

Рисунок 1 – Зависимость времени поиска решения от количества продукций

Для лучшего сравнения миварного подхода и Rete сети их показатели вынесены на отдельный график, представленный на рисунке 2.

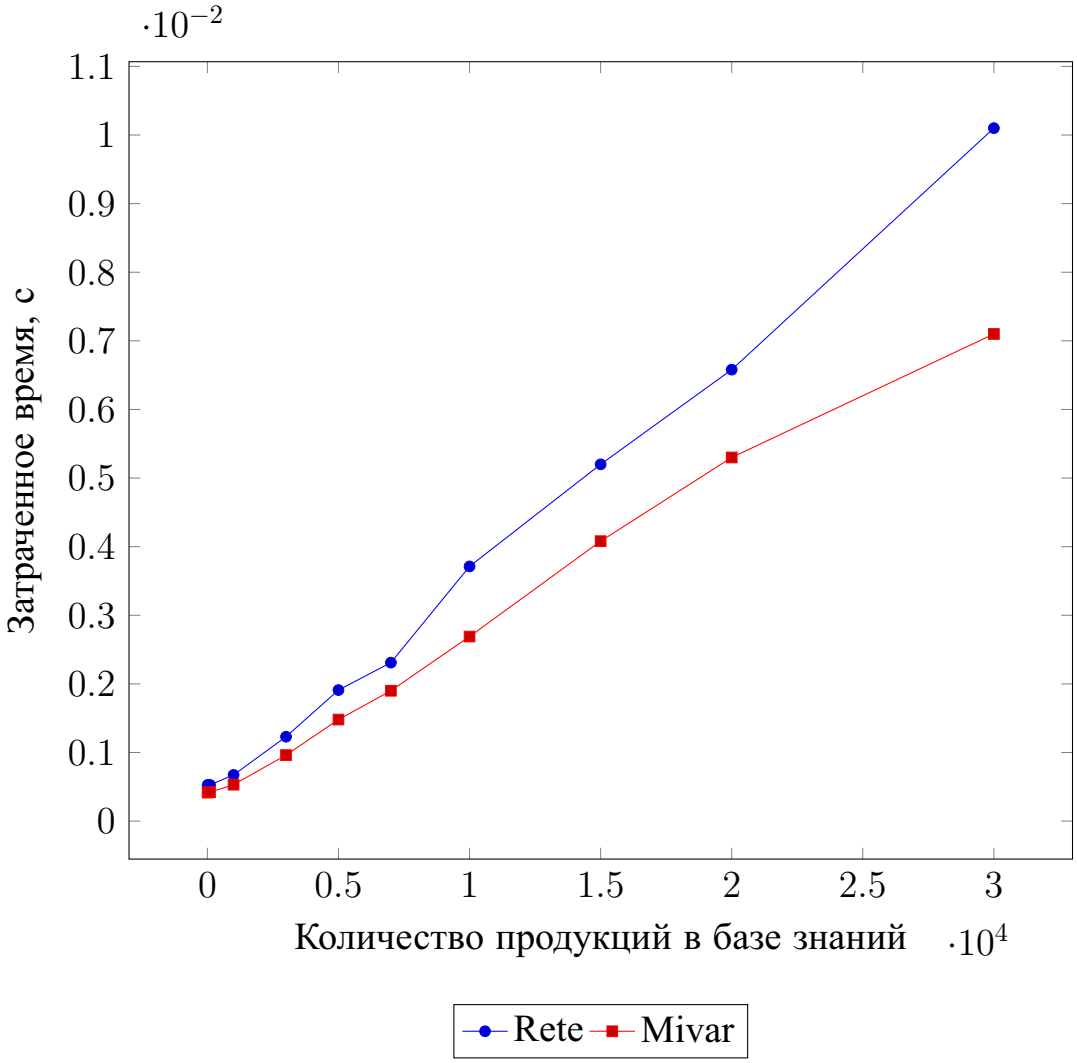

Рисунок 2 – Зависимость времени поиска решения от количества продукций

Анализируя полученные значения, можно отметить, что Rete и миварный подход предоставляют увеличение в скорости на несколько порядков относительно «наивного» алгоритма.

Также были сняты показания затраченного времени при фиксированном значении (1000) количества правил и изменяющемся количестве начальных фактов. Данные представлены на рисунке 3.

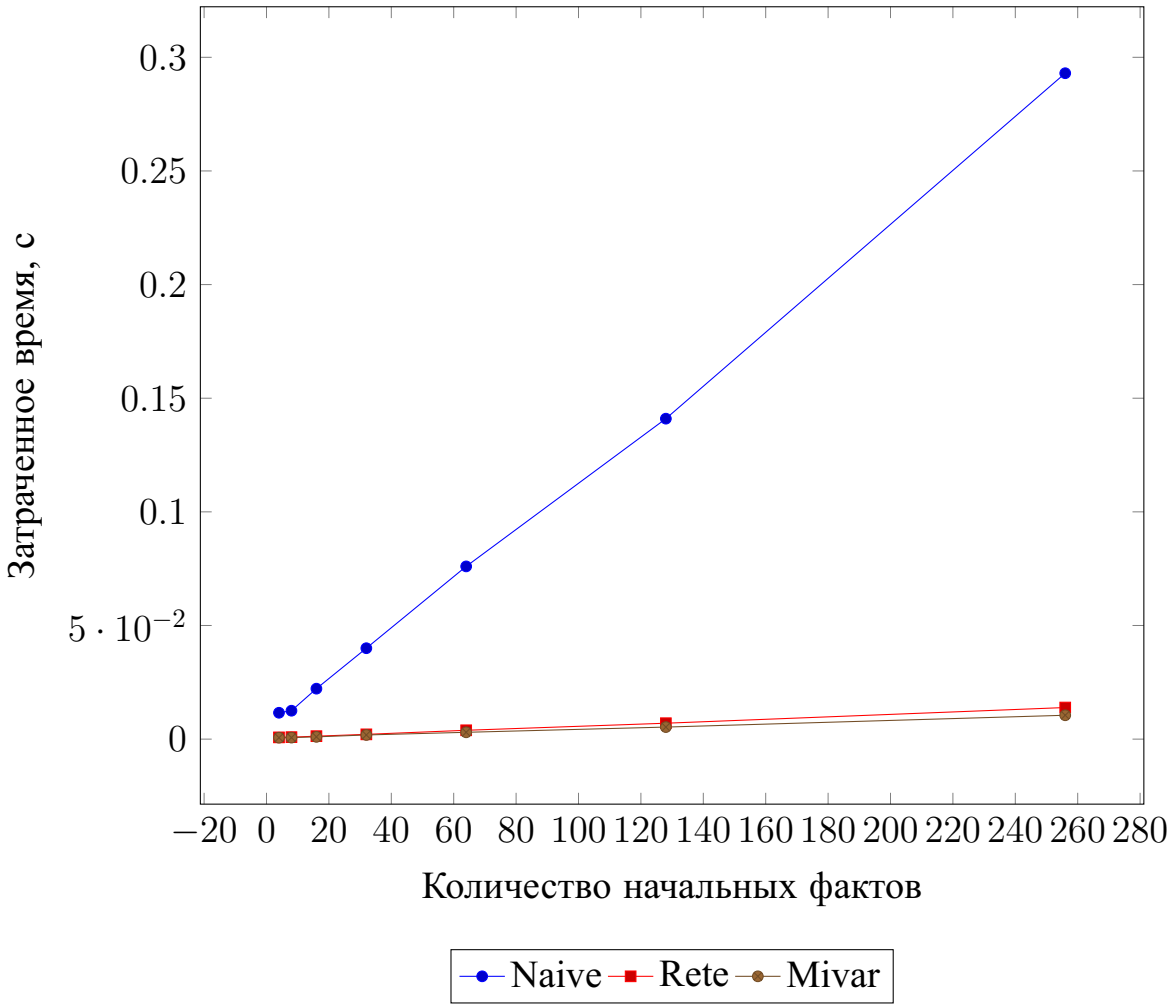

Рисунок 3 – Зависимость времени поиска решения от количества фактов

Для лучшего сравнения миварного подхода и Rete сети их показатели вынесены на отдельный график, представленный на рисунке 4.

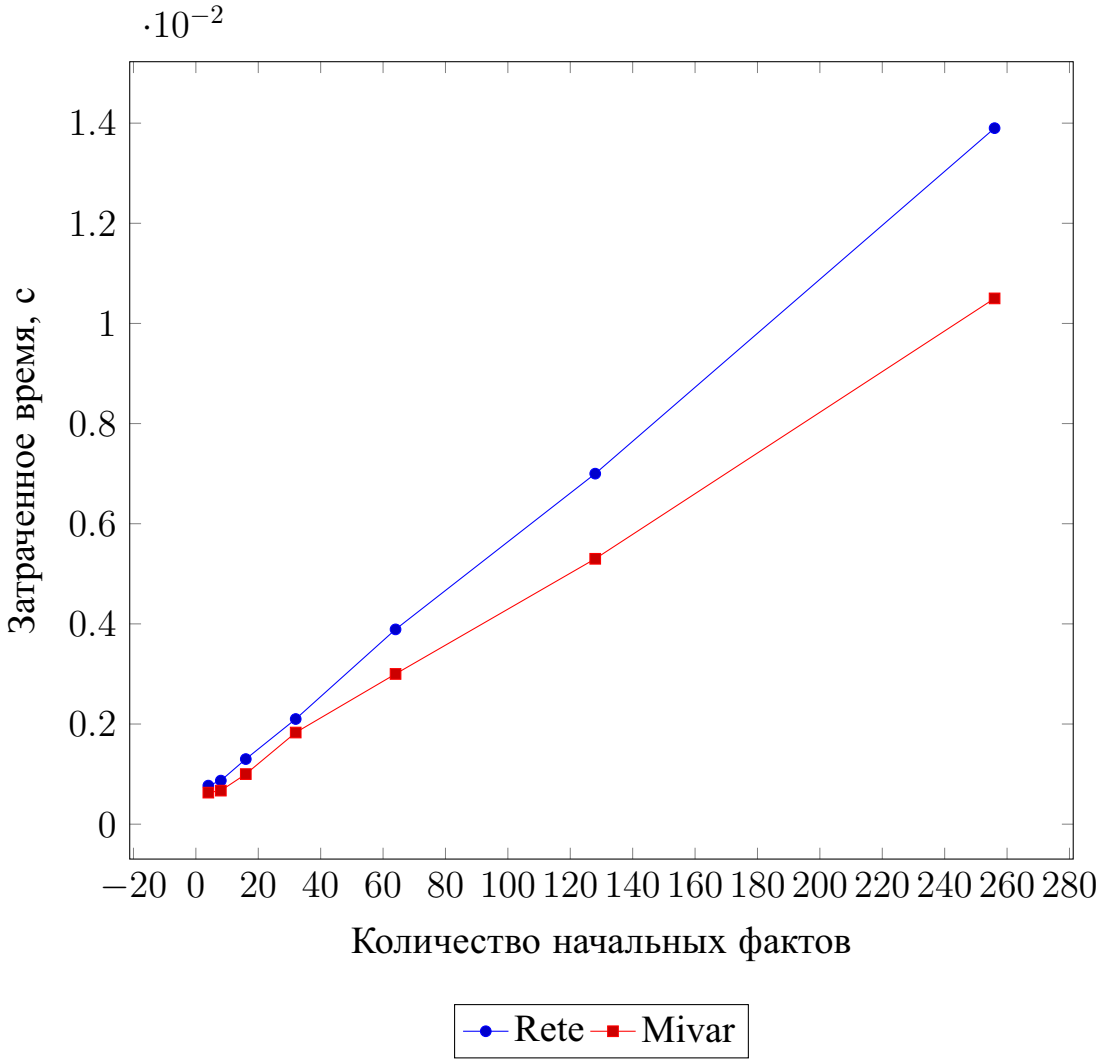

Рисунок 4 – Зависимость времени поиска решения от количества фактов

Были проведены вычисления используемой механизмами логического вывода памяти устройства. С этой целью было проведено измерение объема памяти, занимаемого структурой каждого из предложенных интерпретаторов на фиксированных наборах правил. Результаты вычислений отражены на рисунке 5.

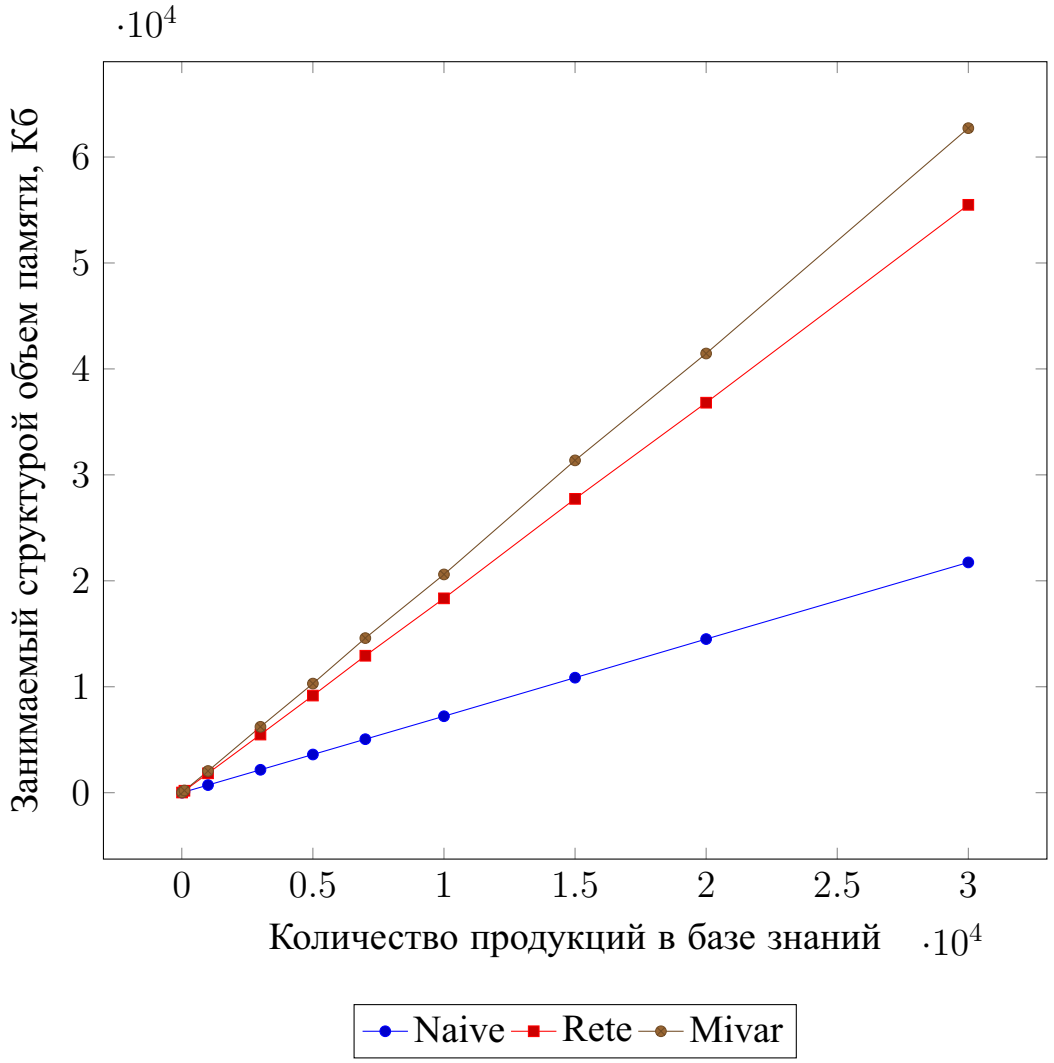

Рисунок 5 – Зависимость используемого объема памяти от количества продукций

ЗАКЛЮЧЕНИЕ

В практической части работы было реализовано:

- механизм логического вывода, основанный на «наивном» алгоритме;
- механизм логического вывода, основанный на Rete сети;
- механизм логического вывода, основанный на миварной сети;
- экспертная система продукционного типа;
- проверочные наборы входных данных.

Была произведена оценка быстродействия предложенных механизмов логического вывода экспериментальным путем и проведено их сравнение. При тестовых запусках на проверочных наборах данных фиксировалось время, затраченное на вывод. Было замечено, что с помощью механизмов логического вывода, основанных на Rete сети и миварной сети можно добиться прироста скорости нахождения решений в 1.5 - 43 раза при качественном увеличении числа правил в сравнении с «наивным» алгоритмом. Также выполнение логического вывода с применением миварной сети показало ускорение в 1.3 раза по сравнению с интерпретатором, основанном на Rete сети. Проверка производительности механизмов логического вывода повышением количества входных данных показала, что Rete и миварная сеть теряют эффективность в среднем в 20 раз медленнее «наивного» алгоритма. Стоит отметить, что снижение производительности решателя с использованием миварной сети происходит в 1.3 раза медленнее машины вывода с Rete сетью. Самым эффективным механизмом логического вывода относительно занимаемого объема памяти является «наивный» алгоритм. В сравнении с Rete сетью миварный подход потребляет больше ресурсов. Это объясняется повышенной сложностью внутренней организации узлов относительно узлов Rete сети.

Таким образом, наилучшими показателями быстродействия обладает интерпретатор с использованием миварной сети. Относительно используемых ресурсов памяти вычислительного устройства Rete алгоритм показал себя более эффективным, чем миварный подход.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1 Business Rules Engine [Электронный ресурс]. — URL: https://www.pega.com/business-rules-engine (Дата обращения 28.05.2018). Загл. с экр. Яз. англ.

2 Progress Corticon 5.7 Documentation: Overview of Progress Corticon [Электронный ресурс]. — URL: https://documentation.progress.com/output/ua/Corticon/#page/corticon/progress-corticon-documentation---where-and-what.html (Дата обращения 28.05.2018). Загл. с экр. Яз. англ.

3 FICO Decision Management Platform [Электронный ресурс]. — URL: http://www.fico.com/en/products/fico-decision-management-platform#overview (Дата обращения 28.05.2018). Загл. с экр. Яз. англ.

4 Operational Decision Manager Version 8.9.0 documentation [Электронный ресурс]. — URL: https://www.ibm.com/support/knowledgecenter/en/SSQP76_8.9.0/welcome/kc_welcome_odmV.html (Дата обращения 28.05.2018). Загл. с экр. Яз. англ.

5 *Leondes, C. T.* Expert Systems / C. T. Leondes. — Cambridge: Academic Press, 2001.

6 *Waterman, D. A.* A Guide to Expert Systems / D. A. Waterman. — Reading: Addison-Wesley, 1985.

7 *Hayes-Roth, F.* Building Expert Systems / F. Hayes-Roth, D. B. Lenat. — Reading: Addison-Wesley, 1983.

8 *Jackson, P.* Introduction to expert systems / P. Jackson. — Reading: Addison-Wesley, 1986.

9 *Муромцев, Д. И.* Введение в технологию экспертных систем / Д. И. Муромцев. — СПб.: СПб ГУ ИТМО, 2005.

10 *Частиков, А. П.* Разработка экспертных систем. Среда CLIPS / А. П. Частиков, Т. А. Гаврилова, Д. Л. Белов. — СПб.: БХВ-Петербург, 2003.

11 *Герман, О. В.* Введение в теорию экспертных систем и обработку знаний / О. В. Герман. — Минск: ДизайнПРО, 1979.

12 *Поспелов, Г. С.* Искусственный интеллект - основа новой информационной технологии / Г. С. Поспелов. — М.: Наука, 1988.

13 *Попов, Э. В.* Статические и динамические экспертные системы / Э. В. Попов, И. Б. Фоминых, Е. В. Кисель, М. Д. Шапот. — М.: Финансы и статистика, 1996.

14 *Морозова, В. А.* Представление знаний в экспертных системах / В. А. Морозова, В. И. Паутов. — Екатеринбург: Издательство Уральского университета, 2017.

15 Модели представления знаний [Электронный ресурс]. — URL: http://www.aiportal.ru/articles/knowledge-models/knowledge-models.html (Дата обращения 28.05.2018). Загл. с экр. Яз. рус.

16 Прямой и обратный вывод в экспертных системах продукционного типа [Электронный ресурс]. — URL: http://baumanki.net/lectures/10-informatika-i-programmirovanie/302-iskusstvennyy-intellekt/4026-62-pryamoy-i-obratnyy-vyvod.html (Дата обращения 28.05.2018). Загл. с экр. Яз. рус.

17 Natural Language Toolkit [Электронный ресурс]. — URL: https://www.nltk.org/index.html (Дата обращения 28.05.2018). Загл. с экр. Яз. англ.

18 *Forgy, C. L.* Rete: A Fast Algorithm for the Many Pattern/Many Object Pattern Match Problem / C. L. Forgy. — Pittsburgh: Carnegie-Mellon University, 1982.

19 How the Rete Algorithm Works [Электронный ресурс]. — URL: https://www.sparklinglogic.com/rete-algorithm-demystified-part-2/ (Дата обращения 28.05.2018). Загл. с экр. Яз. англ.

20 *Doorenbos, R. B.* Production Matching for Large Learning Systems / R. B. Doorenbos. — Pittsburgh: Carnegie Mellon University, 1995.

21 *Варламов, О. О.* Мивар: Линейный логический вывод / О. О. Варламов. — М.: НИИ МИВАР, 2012.

22 tkinter — Python interface to Tcl/Tk [Электронный ресурс]. — URL: https://docs.python.org/3/library/tkinter.html (Дата обращения 28.05.2018). Загл. с экр. Яз. англ.