МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ» (ТУСУР)

Кафедра автоматизации обработки информации (АОИ)

СИСТЕМЫ ИСКУССТВЕННОГО ИНТЕЛЛЕКТА

Методические указания к практическим занятиям для студентов направления «Программная инженерия»
(уровень бакалавриата)

2018

Замятин Николай Владимирович
СИСТЕМЫ ИСКУССТВЕННОГО ИНТЕЛЛЕКТА: Методические указания к практическим занятиям для студентов направления подготовки “Программная инженерия” (уровень бакалавриата) / Н.В. Замятин. – Томск, 2018 - 17 с.

Содержание

1. Введение

Цель дисциплины — изучение теоретических основ построения систем искусственного интеллекта как совокупности формализованных знаний об определенной предметной области, представленных в виде фактов, правил, фреймов, онтологий, семантических сетей, онтологий. В рамках изучения дисциплины осуществляется знакомство с понятиями и видами искусственного интеллекта, функциями и средствами описания систем искусственного интеллекта, спецификой предоставления, а также методами построения систем искусственного интеллекта.

Для достижения перечисленных целей при изучении дисциплины ставятся следующие задачи:

- развитие у студентов системного видения организации систем искусственного интеллекта;
- формирование навыков выявления и представления систем искусственного интеллекта;
- выработка практических навыков разработки систем искусственного интеллекта.

Искусственный интеллект в настоящее время находятся в авангарде научно-технического прогресса и ставит рекорды по темпам развития и по количеству практических приложений в самых разных областях человеческой деятельности.

Существует два направления в развитии искусственного интеллекта и систем на нем основанных.

Первое направление ставит перед собой цель разобраться: как устроен человеческий мозг, в чем смысл жизни человека, каково его предназначение, как и откуда он появился на планете Земля и что с ним будет дальше.

Для второго направления характерен прагматизм для создания компьютеров и программного обеспечения, применяющих методы искусственного интеллекта в практических целях, чтобы добиться высокой эффективности создаваемой ими продукции путем копирования, моделирования, имитации структуры и принципов деятельности человеческого мозга.

В учебной дисциплине изучаются три основных стратегических подхода к созданию систем искусственного интеллекта: логический, нейробионический, теории нечетких множеств.

2. Рекомендации по выполнению практических занятий

Практическое занятие – эта форма систематических учебных занятий, с помощью которых студенты изучают тот или иной раздел определенной учебной дисциплины, входящей в состав учебного плана.

Для того чтобы практические занятия приносили максимальную пользу,

необходимо при подготовке к практическим занятиям использовать материал лекций, который должен закрепляться на практических занятиях как в результате обсуждения и анализа лекционного материала, так и с помощью решения проблемных ситуаций и задач. При подготовке к практическим занятиям следует использовать основную литературу из представленного списка, а также руководствоваться данными указаниями и рекомендациями.

Рекомендуется следующая схема подготовки к практическому занятию:

- открыть методические указания по практическим работам к данной дисциплине
- ознакомиться с целью практического занятия
- просмотреть необходимый теоретический материал из методических указаний
- просмотреть материал из рекомендуемых источников по данной теме практического занятия
- ознакомится с вариантами заданий для данного практического занятия

Практическое заанятие 1. Классификация знаний**.**

Цель работы. Изучить заданную предметную область и построить модель знаний в виде графа.

Методические указания. Для построения модели представления знаний в виде графа необходимо выполнить следующие шаги:

1) Определить целевые действия задачи (являющиеся решениями).

2) Определить промежуточные действия или цепочку действий, между начальным состоянием и конечным (между тем, что имеется, и целевым действием).

3) Опередить условия для каждого действия, при котором его целесообразно и возможно выполнить. Определить порядок выполнения действий.

4) Добавить конкретные факты , исходя из поставленной задачи.

5) Преобразовать полученный порядок действий и соответствующие им факты, условия и действия.

6) Для проверки правильности построения записать цепочки, явно проследив связи между ними.Этот набор шагов предполагает движение при построениимодели от результата к начальному состоянию, но возможно и движение от начального состояния к результату (шаги 1 и 2).

7) Присвоить обозначения фактам Ф, правилам П, действиям Д.

8) Построить граф предметной области. (пример рис.1)

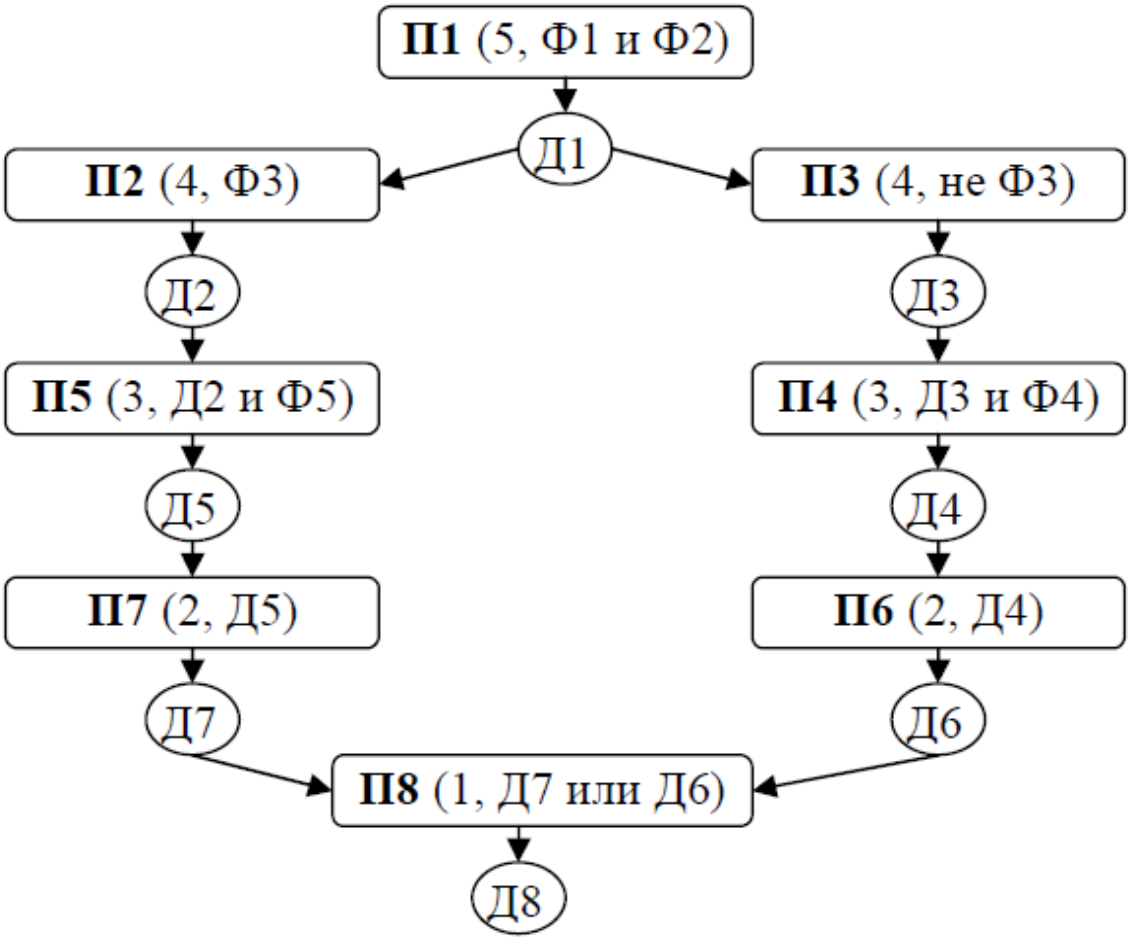

Рис. 1 – Пример графа модели знаний

Варианты заданий

1. Построить модель представления знаний в предметной области «Железная дорога» (продажа билетов).

2. Построить модель представления знаний в предметной области «Торговый центр» (организация).

3. Построить модель представления знаний в предметной области «Автозаправка» (обслуживание клиентов).

4. Построить модель представления знаний в предметной области «Компьютерные сети» (организация).

5. Построить модель представления знаний в предметной области «Университет» (учебный процесс).

6. Построить модель представления знаний в предметной области «Компьютерная безопасность» (средства и способы ее обеспечения).

7. Построить модель представления знаний в предметной области «Компьютерная безопасность» (угрозы).

8. Построить модель представления знаний в предметной области «Интернет-кафе» (организация и обслуживание).

9. Построить модель представления знаний в предметной области «Разработка информационных систем» (ведение информационного проекта).

10. Построить модель представления знаний в предметной области «Туристическое агентство» (работа с клиентами).

11. Построить модель представления знаний в предметной области «Кухня» (приготовление пищи).

12. Построить модель представления знаний в предметной области «Больница» (прием больных).

13. Построить модель представления знаний в предметной области «Кинопрокат» (ассортимент и работа с клиентами).

14. Построить модель представления знаний в предметной области «Прокат автомобилей» (ассортимент и работа с клиентами).

15. Построить модель представления знаний в предметной области «Операционные системы» (функционирование).

16. Построить модель представления знаний в предметной области «Информационные системы» (виды и функционирование).

17. Построить модель представления знаний в предметной области «Предприятие» (структура и функционирование).

Контрольные вопросы

1. Что такое факт?

2.Дайте определение данным?

3. Что такое знания?

4. Что такое поле знаний?

5. Кто такой инженер - когнитолог?

Содержание отчета по практическому занятию

- цель занятия
- краткие теоретические сведения
- описание предметной области
- граф предметной области
- ответы на вопросы

Практическое заанятие 2. Экспертные системы различных предметных областей.

Цель занятия: ознакомиться с особенностями языка CLIPS, получить практические навыки разработки советующих систем, основанных на использовании модели представления знаний.

Краткие теоретические сведения. CLIPS располагает тремя механизмами представления знаний:

процедурным, эвристическим и объектно-ориентированным.

Процедурный механизм позволяет пользователю при помощивстроенных в язык функций разрабатывать или конструировать новыефункции, выполняющие некоторые действия или возвращающие какие-либозначения. В этом смысле CLIPS напоминает такие известные языкипрограммирования, как C, C++ или Pascal. Так, для созданияпользовательских функций используется конструктор deffunction, имеющийследующий синтаксис:

(deffunction имя_функции
[необязательный комментарий]
(список формальных параметров)
(действие_1)
(действие_2)
.....
(действие_N))

Пример, определить функцию om(x,y), которая возвращает целую часть частного от деления переменной y на переменную x:

(deffunction om

(?x ?y)

(div ?y ?x))

В CLIPS имя переменной начинается с символа “ ? “, и что для вызова функции (в данном случае – встроеннойфункции деления нацело div) используется префиксная нотация, а также нато, что вся конструкция представляет собой список, состоящий из четырехполей. Этим CLIPS похож не только на C, но и на LISP.

Эвристический механизм представления знаний в CLIPS реализуетсяпри помощи правил в форме

ЕСЛИ условие_1 и ... и условие_N выполняются,ТО

ВЫПОЛНИТЬ действие_1 и ... и действие_N.

Список условий называется левой частью правила (Left-Hand Side илиLHS). Список действий называется правой частью правила (Right-Hand Sideили RHS). Возможность применить конкретное правило определяется тем,выполняются ли условия, которые сформулированы в его левой части.

Выполнение или невыполнение условий определяется в момент ихсопоставления с так называемыми фактами, которые образуют ни что иное,как базу данных.

В CLIPS такая база данных может представлять некоторую предметную область, исходное или текущее состояние какой-либо проблемы, может моделировать в пространстве или во времени поведение какой-либо системы или любой сущности, которую можно описать посредством множества записей в виде списков.

Для создания базы данныхиспользуется конструктор deffacts. Его синтаксис:

(deffacts имя_базы_данных

[необязательный комментарий]

(факт_1)

(факт_2)

.....

(факт_N))

Каждый факт в базе данных представляет собой запись в виде списка. Список может содержать одно или несколько полей, принимающих символьные либо числовые значения. Список также может быть пустым. Если каждое условие в левой части правила находит себя среди фактов

– происходит активизация правила и выполняются ВСЕ действия,

записанные в его правой части. В противном случае правило неактивизируется.

Работа правила напоминает условный оператор if-then, присутствующий во многих процедурных языках программирования. Интерпретатор CLIPS при старте программы, содержащей множество фактов и правил, интерпретатор CLIPS запускает машину логического вывода, которая выясняет, какие из правил можно активизировать. Это выполняется циклически, причем каждый цикл состоит из трех шагов:

· сопоставление фактов и правил;

· выбор правила, подлежащего активизации;

· выполнение действий, предписанных правилом.

Таким образом, правила, взаимодействующие с базой данных в виде фактов, вносят в нее функциональность и образуют вместе с ней базу знаний. Для создания правила используется конструктор defrule, который имеет следующий синтаксис:

(defrule имя_правила

[необязательный комментарий]

[необязательное объявление]

(условие_1)

(условие_2)

.....

(условие_M)

=>

(действие_1)

(действие_2)

.....

(действие_N))

Левая часть правила отделяется от правой комбинацией символов “=>”, а количество условий и действий в правиле в общем случае не совпадает. **Пример.** Предметная область, участники некоторой конференции, приехавших из разных городов. Все участники обычно проходят регистрацию. Пусть эта процедура представляет собой ввод сведений об участниках в базу данных, вкоторой на каждого участника выделяется одна запись (факт), состоящая из списка с тремя полями. Пусть первое поле имеет символьное значение rep сокращение от representative (представитель). В общем случае это значение может быть любым, а поле может отсутствовать. Во втором поле списка хранится фамилия участника, а в третьем – город, из которого участник прибыл. Содержимое фактов базы данных с именем rep может быть,например, таким:

(deffacts rep

(rep Alejnov Odessa)

(rep Ladak Odessa)

(rep Slobodjanjuk Lvov)

(rep Klitka Lvov)

(rep Bojko Kiev)

(rep Pustovit Odessa)

(rep Spokojnij Odessa)

(rep Shamis Odessa)

(rep Lobovko Kiev)

(rep Zadorozhna Lvov)

(rep Javorskij Lvov))

Используя любой текстовый редактор, нужно создать и сохранить базу данных в виде текстового ASCII-файла с именем, повторяющим имя базыданных (то есть rep). Это позволит легко редактировать данные, независимоот каких-либо

других программных модулей, добавляя новых участников или удаляя выбывших.

После окончания конференции организаторы подводят итоги. Пусть требуется определить количество представителей от каждого города. Для каждого города задаем счетчик и последовательно просматриваем списки в записях файла rep. Если в записи третье поле списка имеет значение Kiev, то содержимое соответствующего счетчика увеличиваем на единицу. Для других городов – аналогично. Программа наязыке CLIPS, реализующая указанный алгоритм, может быть, например,такой:

```
(defglobal ?*odessa* = 0)
(defglobal ?*kiev* = 0)
(defglobal ?*lvov* = 0)
(defrule start
(initial-fact)=>(printout t crlf «REPRESENTATIVES» crlf)
(defrule odessa
(rep ? Odessa)=>(bind ?*odessa* (+ ?*odessa* 1)))
(defrule kiev
(rep ? Kiev)
=>
(bind ?*kiev* (+ ?*kiev* 1)))
(defrule lvov
(rep ? Lvov)
=>
(bind ?*lvov* (+ ?*lvov* 1)))
(defrule result
(declare (salience -1))
(initial-fact)
=>
(printout t «from Odessa: « ?*odessa* crlf)
(printout t «from Kiev: « ?*kiev* crlf)
```

(printout t «from Lvov: « ?*lvov* crlf))

В первых трех строках программы при помощи конструктора defglobal объявляются три глобальные переменные: ?*odessa*, ?*kiev* и ?*lvov*. Эти переменные являются счетчиками. В CLIPS переменная может быть и локальной – но тогда она связывается только с тем правилом, в котором объявляется. Далее следует правило с именем start, левая часть которого представляет собой запись (initial-fact). Так обозначается системный начальный факт, который создается в рабочей памяти интерпретатора CLIPS по команде (reset) до запуска программы на выполнение.

В CLIPS-программах распространенными правилами являются такие, которые добавляют факты в базу данных, либо, наоборот, удаляют их. При старте программы в базе данных нет фактов, удовлетворяющих хотя бы одному правилу. В этом случае программа ничего не выполнит. Для того чтобы начать вычисления и используется системный начальный факт, который, независимо от фактов в базе данных, активизирует некоторое правило, добавляющее такие факты, которые, в свою очередь, активизируют правила, условия которых не выполнялись в начальный момент.

В данной программе (initial-fact) запускает правило start, которое активизируется независимо от фактов в файле rep и присутствует в программе только с одной целью – вывести заголовок. Для этого в его правой части вызывается встроенная функция printout с ключом t, выводящая на стандартное устройство вывода (монитор) заголовок, заключенный в кавычки. Комбинация символов crlf является аналогом endl в C++ и служит для перевода курсора на следующую строку.

Следующие три правила с именами odessa, kiev и lvov можно назвать ядром программы. В них производится подсчет количества участников – соответственно, из Одессы, Киева и Львова.

Рассмотрим правило lvov. Оно активизируется в том случае, когда в базе данных находится факт (rep ? Lvov). Символ “ ? “ во втором поле этого списка

означает символ универсальной подстановки и заменяет собой любую фамилию. Отсюда следует, что правило lvov активизируется столько раз, сколько раз факт (rep ? Lvov) присутствует в базе данных. При этом столько же раз выполнятся действия, содержащиеся в правой части правила. Встроенная функция bind – аналог оператора присваивания. Следовательно, содержимое переменной ?*lvov* увеличивается на единицу и результат сохраняется в этой же переменной.

Аналогично работают правила odessa и kiev.Действия, которые выполняются в последнем правиле программы, отражены в его названии. Правая часть правила особых комментариев не требует, в то время как левая часть заслуживает подробного рассмотрения. В CLIPS существует несколько стратегий очередности выполнения правил, а сами правила могут иметь приоритет, который задается встроенной функцией declare с параметром salience (особенность). Этот параметр может принимать целочисленные значения от –10000 до +10000. По умолчанию для всех правил величина salience равна нулю. Если в правиле result не указать приоритет, оно будет конфликтовать с правилом start за очередность выполнения, так как у этих правил одинаковая левая часть. Для устраненияконфликта в правиле result приоритет указан явно и со знаком минус, в связи с чем это правило выполнится последним.

Используя любой текстовый редактор, наберите и сохраните текст программы в ASCII-файле со стандартным для CLIPS-программ расширением .clp и с именем represent. Командой clips вызовем интерпретатор CLIPS, командой (load имя_файла) загрузим в интерпретатор файлы rep и represent.clp, командами (reset) и (run) запустим программу represent.clp на выполнение.

```
CLIPS (V6.21 06/15/03)
CLIPS> (load rep)
.....
TRUE
CLIPS> (load represent.clp)
.....
```

TRUE

CLIPS> (reset)

CLIPS> (run)

REPRESENTATIVES

from Odessa: 5

from Kiev: 2

from Lvov: 4

CLIPS>

Сообщение интерпретатора TRUE означает, что в файле нет синтаксических ошибок и команда загрузки выполнена корректно. Многоточием представлены другие сообщения интерпретатора, которые в данном случае опущены.

Как следует из описанных действий, в интерпретаторе CLIPS находятся два файла. Первый, с именем rep, является базой данных. Второй, с именем represent.clp, содержит сведения (правила) о том, как эти данные могут быть использованы. Таким образом, вместе файлы образуют базу знаний, которая содержит, по крайней мере, два знания. Первое – общий состав участников конференции. Его можно посмотреть, не выходя из интерпретатора по команде (facts). Второе знание – количество участников от каждого города. В рассмотренном примере база знаний состоит из двух программных модулей.

Задание. Необходимо, изучив команды CLIPS и приведенный пример, разработать экспертную систему по заданной предметной области.

Варианты заданий:

Варианты заданий

1. Построить модель представления знаний в предметной области «Железная дорога» (продажа билетов).

2. Построить модель представления знаний в предметной области «Торговый центр» (организация).

3. Построить модель представления знаний в предметной области «Автозаправка» (обслуживание клиентов).

4. Построить модель представления знаний в предметной области «Компьютерные сети» (организация).

5. Построить модель представления знаний в предметной области «Университет» (учебный процесс).

6. Построить модель представления знаний в предметной области «Компьютерная безопасность» (средства и способы ее обеспечения).

7. Построить модель представления знаний в предметной области «Компьютерная безопасность» (угрозы).

8. Построить модель представления знаний в предметной области «Интернет-кафе» (организация и обслуживание).

9. Построить модель представления знаний в предметной области «Разработка информационных систем» (ведение информационного проекта).

10. Построить модель представления знаний в предметной области «Туристическое агентство» (работа с клиентами).

11. Построить модель представления знаний в предметной области «Кухня» (приготовление пищи).

12. Построить модель представления знаний в предметной области «Больница» (прием больных).

13. Построить модель представления знаний в предметной области «Кинопрокат» (ассортимент и работа с клиентами).

14. Построить модель представления знаний в предметной области «Прокат автомобилей» (ассортимент и работа с клиентами).

15. Построить модель представления знаний в предметной области «Операционные системы» (функционирование).

16. Построить модель представления знаний в предметной области «Информационные системы» (виды и функционирование).

17. Построить модель представления знаний в предметной области «Предприятие» (структура и функционирование).

Контрольные вопросы

1. Какая модель знаний используется в CLIPS?

2. Зачем нужна объяснительная подсистема?

3. Что хранится в базе данных?

4. Как выполняется вывод советов в экспертной системе?

5. В чем закдючается назначение базы знаний?

Содержание отчета по практическому занятию

- цель работы
- краткие теоретические сведения
- описание предметной области
- структура экспертной системы
- Листинг программы
- Ответы на вопросы

Рекомендуемые источники

1. Системы искусственного интеллекта: Учебное пособие / Н. В. Замятин - 2018. 244 с.: Научно-образовательный портал ТУСУР, https://edu.tusur.ru/publications/7269

2. Рутковская, Д. Нейронные сети, генетические алгоритмы и нечеткие системы: Пер.с польск.И.Д.Рудинского. ЭБС “Лань” [Электронный ресурс] / Д. Рутковская, М. Пилиньский, Л. Рутковский. — Электрон. дан. — М. : Горячая линия-Телеком, 2013. — 384 с. – Режим доступа http://e.lanbook.com/book/11843.

3. Замятин, Н. В. Нечеткая логика и нейронные сети: Учебное пособие [Электронный ресурс] / Замятин Н. В. — Томск: ТУСУР, 2014. — 289 с. — Режим доступа: https://edu.tusur.ru/publications/7020.

4. Цуканова, Н.И. Теория и практика логического программирования на языке Visual Prolog 7.Учебное пособие для вузов. [Электронный ресурс] : Учебные пособия / Н.И. Цуканова, Т.А. Дмитриева. — Электрон. дан. — М. : Горячая линия-Телеком, 2013. — 232 с. — Режим доступа: http://e.lanbook.com/book/11847.