**Министерство науки и высшего образования Российской Федерации Федеральное государственное бюджетное образовательное учреждение высшего образования** **«Новгородский государственный университет имени Ярослава Мудрого»**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Кафедра ИТиС

«Продукционная модель представления знаний»

Лабораторная работа №2

По дисциплине: Системы искусственного интеллекта

Отчет

Выполнил: студент гр. 8091

\_\_\_\_\_\_\_ \_\_\_\_Кудряшов И. С.

Проверил: Преподаватель

\_\_\_\_\_\_\_\_\_\_\_Михайлов Д. В.

Великий Новгород

2021

1. **Формулировка цели и задач**

Целью данной работы является приобретение студентами умений и навыков реализации пополняемой динамической базы знаний, не включаемой непосредственно в текст программы.

Необходимые для достижения поставленной цели задачи состоят в следующем:

· научиться оценивать возможности применения современных языков высокого уровня для реализации баз знаний экспертных систем (ЭС);

· изучение технических аспектов реализации продукционной модели представления знаний.

1. **Конфигурация системы продукций**

Предметная область для ЭС – Виды спорта

Виды спорта – разбиты на группы, в зависимости от травмоопасности.

Таблица 1. Описание предметной области

|  |  |  |
| --- | --- | --- |
| **Группа** | **Название** | **Характеристики** |
| Низкий | Бег | Уровень развития силы 1  Уровень развития выносливости 4  Внешний вид 3  Энергия в обычной жизни 3  Энергия во время тренировки 4  Интерес 4  Самочувствие во время тренировки 3  Самочувствие после тренировки 5  Самочувствие на следующий день 4 |
| Плавание | Уровень развития силы 2  Уровень развития выносливости 4  Внешний вид 1  Энергия в обычной жизни 3  Энергия во время тренировки 3  Интерес 1  Самочувствие во время тренировки 5  Самочувствие после тренировки 2  Самочувствие на следующий день 3 |
| Велосипед | Уровень развития силы 1  Уровень развития выносливости 3  Внешний вид 2  Энергия в обычной жизни 1  Энергия во время тренировки 5  Интерес 5  Самочувствие во время тренировки 6  Самочувствие после тренировки 2  Самочувствие на следующий день 1 |
| Средний | Тренажёрный зал | Уровень развития силы 5  Уровень развития выносливости 1  Внешний вид 5  Энергия в обычной жизни 2  Энергия во время тренировки 1  Интерес 3  Самочувствие во время тренировки 4  Самочувствие после тренировки 3  Самочувствие на следующий день 2 |
| Стрит воркаут | Уровень развития силы 4  Уровень развития выносливости 2  Внешний вид 4  Энергия в обычной жизни 2  Энергия во время тренировки 2  Интерес 2  Самочувствие во время тренировки 4  Самочувствие после тренировки 2  Самочувствие на следующий день 4 |
| Высокий | Кроссфит | Уровень развития силы 4  Уровень развития выносливости 6  Внешний вид 4  Энергия в обычной жизни 5  Энергия во время тренировки 5  Интерес 5  Самочувствие во время тренировки 1  Самочувствие после тренировки 4  Самочувствие на следующий день 5 |
| Единоборства | Уровень развития силы 3  Уровень развития выносливости 5  Внешний вид 4  Энергия в обычной жизни 4  Энергия во время тренировки 6  Интерес 5  Самочувствие во время тренировки 2  Самочувствие после тренировки 1  Самочувствие на следующий день 6 |

Для описания объекта или его отдельных свойств выбираются некоторые характеристики – величины, которые могут принимать либо количественные, либо качественные значения.

В свою очередь, совокупность всех характеристик некоторого объекта образует так называемый список разрешенных характеристик данного объекта. Списки разрешенных характеристик и разрешенных значений этих характеристик охватывают множество всех имеющихся фактов, подлежащих хранению в базе знаний экспертной системы.

В экспертных системах правила, по которым решаются проблемы в конкретной предметной области, хранятся в базе знаний. *Проблемная область ЭС* описывается посредством фактов и правил. Проблемы ставятся перед системой в виде совокупности фактов, описывающих некоторую ситуацию, и система с помощью базы знаний пытается вывести заключение из этих фактов.

Для данной лабораторной работы в качестве базы знаний был выбран текстовый файл.

В начале файла описываются все атрибуты и их значения. 1 атрибут на 1 строке.

Пример атрибутов:

Уровень\_развития\_силы\* = 1 | 2 | 3 | 4 | 5

Уровень\_развития\_выносливости\* = 1 | 2 | 3 | 4 | 5 | 6

Внешний\_вид\* = 1 | 2 | 3 | 4 | 5

Энергия\_в\_обычной\_жизни\* = 1 | 2 | 3 | 4 | 5

Энергия\_во\_время\_тренировки\* = 1 | 2 | 3 | 4 | 5 | 6

Интерес\* = 1 | 2 | 3 | 4 | 5

Самочувствие\_во\_время\_тренировки\* = 1 | 2 | 3 | 4 | 5 | 6

Самочувствие\_после\_тренировки\* = 1 | 2 | 3 | 4 | 5

Самочувствие\_на\_следующий\_день\* = 1 | 2 | 3 | 4 | 5 | 6

Риск\_травм\* = Низкий | Средний | Высокий

Также в базе знаний хранятся правила, синтаксис которых:

IF A THEN S, где А – условие; S – заключение.

Пример правил:

IF Риск\_травм = Средний & Уровень\_развития\_силы = 5 & Уровень\_развития\_выносливости = 1 & Внешний\_вид = 5 & Энергия\_в\_обычной\_жизни = 2 & Энергия\_во\_время\_тренировки = 1 & Интерес = 3 & Самочувствие\_во\_время\_тренировки = 4 & Самочувствие\_после\_тренировки = 3 & Самочувствие\_на\_следующий\_день = 2 THEN Вид\_спорта = Тренажёрный\_зал

1. **Описание способа организации поиска на графе**

Поиск на графе организован с помощью последовательного отброса заведомо ложных путей и проверки доступных с помощью перебора.

1. **Описание конфликтного набора и алгоритм разрешения конфликта при логическом выводе**

В данной лабораторной работе был выбран обратный порядок вывода. В системах с обратным выводом вначале выдвигается гипотеза, а затем механизм вывода в процессе работы как бы возвращается назад, переходя от выдвинутой гипотезе к фактам и пытаясь найти среди них те, которые подтверждают эту гипотезу.

Конфликтный набор:

Система предлагает ответить на вопрос, ответами являются такие символы как y и n.

Чтобы вызвать конфликтный набор, можно на эти вопросы ответить, как приведено ниже:

Высокий->5

Ответ системы: База знаний не содержит информацию об этом виде спорта

Механизм решения конфликта представляет собой алгоритм, который при возможности применения сразу нескольких правил запрашивает уточнение одного из параметров.

Когда пользователь опровергает определенный факт, т.е. отвечает отрицательно, алгоритм прерывает проверку текущего правила и переходит к следующему по списку в базе знаний.

1. **Тестовый набор правил**

Высокий -> 3 -> 5 -> 4 -> 4 -> 6 -> 5 -> 2 -> 1 -> 6

Ответ системы: Единоборства

Низкий -> 2 -> 4 -> 1 -> 3 -> 3 -> 1 -> 5 -> 2 -> 3

Ответ системы: Плавание

**Пример конфликтного набора:**

Низкий -> 1

В данном случаи правила для бега и велосипеда могут сработать одновременно и возникает конфликтная ситуация. Для ее решения у пользователя запрашивается уточнение одного из параметров. Когда пользователь опровергает параметр, алгоритм прерывает проверку текущего правила и переходит к следующему по списку в базе знаний.

Низкий -> 1 -> 4

Или

Низкий -> 1 -> 3

1. **Вывод**

Мною была изучена реализация пополняемой динамической базы знаний, не включаемой непосредственно в текст программы.

Также я изучил реализацию продукционной модели представления знаний.

Также была построена база знаний, предметной области, виды спорта.

**Приложение:**

1. **Текст программы**

Файл Eparser.py

import copy

def parse(path):

    parameters = {}

    rules = []

    try:

        with open(path, 'r', encoding="utf-8") as f:

            lines = f.readlines()

    except:

        print("Проблема с открытием файла " + path)

        return

    for line in lines:

        if not line or line.startswith('-') or line.startswith('#'):

            continue

        elif line.startswith('IF'):

            current = {}

            sides = line.replace('IF', '').split('THEN')

            current['LHS'] = {}

            conditions = sides[0].split('&')

            for condition in conditions:

                HS = list(map(str.strip, condition.split('=', 1)))

                current['LHS'][HS[0]] = [s.strip() for s in HS[1].split('|')]

            action = sides[1]

            HS = list(map(str.strip, action.split('=')))

            current['RHS'] = {HS[0]: HS[1]}

            rules.append(copy.deepcopy(current))

        else:

            splitLine = line.split('=', 1)

            parameters[splitLine[0].strip()] = [s.strip()

                                                for s in splitLine[1].split('|')]

    return parameters, rules

Файл Shell.py

import Eparser

# returns rule in IF THEN fromat

def rule\_repr(rule):

    LHS = []

    for attr, values in rule['LHS'].items():

        LHS.append(attr + " = " + "|".join(values))

    (RHSkey, RHSvalue) = list(rule['RHS'].items())[0]

    return "IF " + " & ".join(LHS) + " THEN " + RHSkey + " = " + RHSvalue

# Prints Working Memory

def printRM():

    print("Memory:")

    for r, v in RM.items():

        print(r, " = ", v)

# Returns a list of rules, RHS = param value

def getConflictRules(rules, goal):

    ruleset = []

    for rule in rules:

        attribute = list(rule['RHS'].keys())[0]

        if attribute == goal:

            ruleset.append(rule)

    return ruleset

# The function checks if there is at least one rule, the right side of which is the goal value.

def conflictRuleExists(rules, goal):

    for rule in rules:

        attribute = list(rule['RHS'].keys())[0]

        if attribute == goal:

            return True

    return False

# The function checks if this rule works.

def ruleWorks(rule, RM):

    conditions = rule['LHS']

    for param in conditions:

        if param in RM:

            if RM[param] not in conditions[param]:

                return False

        else:

            return False

    return True

# The function for the user to enter the default parameter in the working memory

def parameterInput(param, RM):

    value = input("Enter parameter value '" + param +

                  "' " + str(parameters[param+"\*"]) + ": ")

    while(value not in parameters[param+"\*"]):

        value = input()

    RM[param] = value

# Prints attributes / params, their values and all rules in Working Memory

def printKnowledgeBase(parameters, rules):

    print('-'\*105)

    print('|' + '\t'\*6 + 'Knowledge base' + '\t'\*6 + '|')

    print('-'\*105 + '\n')

    print("Attributes:")

    for attr, value in parameters.items():

        print(attr + " = " + " | ".join(value))

    print("\nRules:")

    for i, rule in enumerate(rules):

        print(str(i+1) + ") " + rule\_repr(rule))

    print('-'\*105 + '\n')

# Parse attributes from file

parameters, rules = Eparser.parse('./Sport.txt')

# Print KB

printKnowledgeBase(parameters, rules)

# Working memory, stack with targets and a list of already verified attributes

RM = {}

goals = []

checked\_goals = []

# Asks user to enter target hypothesis

#goal = input('Please, enter "Phone": ')

goals.append('Вид\_спорта')

# Main loop

while(True):

    # # If stack is empty, quit

    if len(goals) == 0:

        break

    # Vspomog variab

    new\_goal = False

    new\_parameter = False

    # Save current target

    goal = goals[-1]

    # Create a set of conflicting rules and keep them count

    conflictRules = getConflictRules(rules, goal)

    remainingRules = len(conflictRules)

    # If no matching rule is found, stop the loop and notify the user.

    if remainingRules == 0:

        print('The knowledge base does not contain information about this phone.')

        break

    # Prints a set of rules and working memory.

    print('Rules: ')

    for cr in conflictRules:

        print(rule\_repr(cr))

    printRM()

   # The cycle goes through a set of conflicting rules.

    # If the rule works:

    # 1) delete the current target from above

    # 2) keeps its right side in working memory

    # 3) set the variable new\_goal to True and exit the loop

    for cr in conflictRules:

        if ruleWorks(cr, RM):

            (RHSkey, RHSvalue) = list(cr['RHS'].items())[0]

            RM[RHSkey] = RHSvalue

            curr\_goal = goals.pop()

            print("Goal: " + curr\_goal + " = " + RHSvalue)

            new\_goal = True

            break

# If the target is updated

    if new\_goal:

        continue

    # for each rule in the conflict rule set

    for cr in conflictRules:

        if new\_goal:

            break

        # decrease count of unchecked rules

        remainingRules -= 1

        conditions = cr['LHS']

        # for each parameter of checked rules

        for param in conditions:

           # if the parameter has already been checked and it was not possible to execute it,

            # skip rule (do not check other parameters)

            if param in checked\_goals:

                break

            # if current parameter is in working memory

            if param in RM:

               # parameter value does not match the value in working memory

                if RM[param] not in conditions[param]:

                    break

            else:  # parameter is not in memory

                # if one of the rules fulfills the current parameter, set it as a target

                if conflictRuleExists(rules, param) and param not in goals:

                    goals.append(param)

                    new\_goal = True

                    break

                elif param + "\*" in parameters:

                    # None of the rules executes the parameter

                    # if possible (the parameter name ends with '\*'),

                    # asks user to enter parameter value

                    parameterInput(param, RM)

                    new\_parameter = True

                    break

                else:

                   # parameter cannot be obtained from any rule

                    # user cannot login

                    checked\_goals.append(param)

        if new\_parameter:

            break

        # If a new target has not been established and all the rules have been verified, delete the target from above.

          # and save it to the list of goals that can not be achieved

        if remainingRules == 0 and not new\_goal:

            curr\_goal = goals.pop()

            checked\_goals.append(curr\_goal)

            print('База знаний не содержит информацию об этом виде спорта')

    print('-'\*100)

Файл Sport.txt

Уровень\_развития\_силы\* = 1 | 2 | 3 | 4 | 5

Уровень\_развития\_выносливости\* = 1 | 2 | 3 | 4 | 5 | 6

Внешний\_вид\* = 1 | 2 | 3 | 4 | 5

Энергия\_в\_обычной\_жизни\* = 1 | 2 | 3 | 4 | 5

Энергия\_во\_время\_тренировки\* = 1 | 2 | 3 | 4 | 5 | 6

Интерес\* = 1 | 2 | 3 | 4 | 5

Самочувствие\_во\_время\_тренировки\* = 1 | 2 | 3 | 4 | 5 | 6

Самочувствие\_после\_тренировки\* = 1 | 2 | 3 | 4 | 5

Самочувствие\_на\_следующий\_день\* = 1 | 2 | 3 | 4 | 5 | 6

Риск\_травм\* = Низкий | Средний | Высокий

IF Риск\_травм = Средний & Уровень\_развития\_силы = 5 & Уровень\_развития\_выносливости = 1 & Внешний\_вид = 5 & Энергия\_в\_обычной\_жизни = 2 & Энергия\_во\_время\_тренировки = 1 & Интерес = 3 & Самочувствие\_во\_время\_тренировки = 4 & Самочувствие\_после\_тренировки = 3 & Самочувствие\_на\_следующий\_день = 2 THEN Вид\_спорта = Тренажёрный\_зал

IF Риск\_травм = Низкий & Уровень\_развития\_силы = 1 & Уровень\_развития\_выносливости = 4 & Внешний\_вид = 3 & Энергия\_в\_обычной\_жизни = 3 & Энергия\_во\_время\_тренировки = 4 & Интерес = 4 & Самочувствие\_во\_время\_тренировки = 3 & Самочувствие\_после\_тренировки = 5 & Самочувствие\_на\_следующий\_день = 4 THEN Вид\_спорта = Бег

IF Риск\_травм = Высокий & Уровень\_развития\_силы = 4 & Уровень\_развития\_выносливости = 6 & Внешний\_вид = 4 & Энергия\_в\_обычной\_жизни = 5 & Энергия\_во\_время\_тренировки = 5 & Интерес = 5 & Самочувствие\_во\_время\_тренировки = 1 & Самочувствие\_после\_тренировки = 4 & Самочувствие\_на\_следующий\_день = 5 THEN Вид\_спорта = Кроссфит

IF Риск\_травм = Средний & Уровень\_развития\_силы = 4 & Уровень\_развития\_выносливости = 2 & Внешний\_вид = 4 & Энергия\_в\_обычной\_жизни = 2 & Энергия\_во\_время\_тренировки = 2 & Интерес = 2 & Самочувствие\_во\_время\_тренировки = 4 & Самочувствие\_после\_тренировки = 2 & Самочувствие\_на\_следующий\_день = 4 THEN Вид\_спорта = Стрит\_воркаут

IF Риск\_травм = Низкий & Уровень\_развития\_силы = 2 & Уровень\_развития\_выносливости = 4 & Внешний\_вид = 1 & Энергия\_в\_обычной\_жизни = 3 & Энергия\_во\_время\_тренировки = 3 & Интерес = 1 & Самочувствие\_во\_время\_тренировки = 5 & Самочувствие\_после\_тренировки = 2 & Самочувствие\_на\_следующий\_день = 3 THEN Вид\_спорта = Плавание

IF Риск\_травм = Низкий & Уровень\_развития\_силы = 1 & Уровень\_развития\_выносливости = 3 & Внешний\_вид = 2 & Энергия\_в\_обычной\_жизни = 1 & Энергия\_во\_время\_тренировки = 5 & Интерес = 5 & Самочувствие\_во\_время\_тренировки = 6 & Самочувствие\_после\_тренировки = 2 & Самочувствие\_на\_следующий\_день = 1 THEN Вид\_спорта = Велосипед

IF Риск\_травм = Высокий & Уровень\_развития\_силы = 3 & Уровень\_развития\_выносливости = 5 & Внешний\_вид = 4 & Энергия\_в\_обычной\_жизни = 4 & Энергия\_во\_время\_тренировки = 6 & Интерес = 5 & Самочувствие\_во\_время\_тренировки = 2 & Самочувствие\_после\_тренировки = 1 & Самочувствие\_на\_следующий\_день = 6 THEN Вид\_спорта = Единоборства