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

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

Кафедра ИТиС

«Поисковая система на основе фреймовой модели представления знаний»

Лабораторная работа №3

По дисциплине: Системы искусственного интеллекта

Отчет

Выполнил: студент гр. 8091

\_\_\_\_\_\_\_ \_\_\_\_Шаклеин В. В.

Проверил: Преподаватель

\_\_\_\_\_\_\_\_\_\_\_Михайлов Д.В.

Великий Новгород

2021

1. **Формулировка цели и задач**

Рассмотреть представление знаний фреймами и соответствующие способы управления выводом.

1. **Модель представления знаний**

Предметная область компьютерные вирусы. Их я разбил на локальные и удалённые.

Фреймом называется структура для описания понятия или ситуации, состоящая из характеристик этой ситуации и их значений.

Особенность фреймовых моделей представления знаний состоит в том, что все понятия, описываемые в каждом из узлов модели, определяются набором атрибутов и их значениями, которые содержатся в слотах фрейма.

Слот — это атрибут, который связывается с узлом в системе, основанной на фреймах. Он является основной составляющей фрейма. Он имеет имя слота и значение слота.

С каждым слотом может быть связана одна или несколько процедур, которые выполняются, когда изменяются значения слотов. Чаще всего со слотами связываются процедуры:

ЕСЛИ-ДОБАВЛЕНО (IF-ADDED) — выполняется в том случае, когда новая информация помещается в слот;

Эта процедура следит за изменением информации, приписываемой данному узлу, и проверяет, что при изменении тех или иных значений производятся соответствующие действия.

Фреймовые системы позволяют использовать объектно-ориентированный подход к представлению знаний.

Таблица 1. Фрейм "Компьютерный вирус"

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя слота | Указатель наследования | Тип данных | Значение слота | Демон |
| Тип | U(Unique) | FRAME  LIST | «Local», «Remote» |  |
| Название | U(Unique) | Text |  |  |
| ОС | U(Unique) | Text | “Windows”, “OSX”, “Linux” |  |
| Год | U(Unique) | Integer |  | Проверка на адекватность путём сравнения с годом выпуска ОС и текущим годом |
| Тип вируса | U(Unique) | Text |  |  |

Таблица 2. Фрейм "Локальный вирус"

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя слота | Указатель наследования | Тип данных | Значение слота | Демон |
| Тип | S(Same) | FRAME | Компьютерный вирус |  |
| Присоединённость к другой программе | U(Unique) | Bool |  |  |
| Простота обнаружения | U(Unique) | Bool |  |  |

Таблица 3. Фрейм "Удалённый вирус"

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя слота | Указатель наследования | Тип данных | Значение слота | Демон |
| Тип | S(Same) | FRAME | Компьютерный вирус |  |
| Вектор атаки | U(Unique) | Text | “Network”, “Web” |  |

Таблица 4. Фрейм Локальный вирус Shlayer

|  |  |  |  |
| --- | --- | --- | --- |
| Имя слота | Указатель наследования | Тип данных | Значение слота |
| Тип | S(Same) | FRAME | Локальный вирус |
| Название | U(Unique) | Text | Shlayer |
| ОС | U(Unique) | Text | OSX |
| Год | U(Unique) | Integer | 2019 |
| Тип вируса | U(Unique) | Text | Trojan |
| Присоединённость к другой программе | U(Unique) | Bool | Да |
| Простота обнаружения | U(Unique) | Bool | Да |

Таблица 5. Фрейм Удалённый вирус SQL Slammer

|  |  |  |  |
| --- | --- | --- | --- |
| Имя слота | Указатель наследования | Тип данных | Значение слота |
| Тип | S(Same) | FRAME | Удалённый вирус |
| Название | U(Unique) | Text | SQL Slammer |
| ОС | U(Unique) | Text | Linux |
| Год | U(Unique) | Integer | 2003 |
| Тип вируса | U(Unique) | Text | Exploit |
| Вектор атаки | U(Unique) | Text | Network |

1. **Механизм управления вводом**

Фреймовые системы позволяют использовать объектно-ориентированный подход к представлению знаний. Эта структура базируется на отношениях IS-А между фреймами, описывающими некий вирус.

При этом наши представления и знания об объекте "Компьютерный вирус" детализируем рядом понятий, которые определим через набор атрибутов и возможными типы их значений. Все атрибуты разместим в слотах фрейма.

Таким образом, на самом верхнем уровне иерархии определен фрейм “Компьютерный вирус”. В данном случае компьютерные вирусы разделяются на локальные вирусы и удалённые вирусы, которые определены в качестве соответствующих дочерних фреймов.

Фреймы, описывающие различные объекты называются шаблонами, а фреймы верхнего уровня, использующиеся для представления этих шаблонов, называются фреймами класса.



Рисунок 1 Структура фреймов

1. **Описание механизма поиска по запросу в поисковой системе**

В этой лабораторной работе предусмотрен поиск информации об определённом вирусе, это происходит путем ввода названия вируса, и программа предоставляет основную информацию об этом вирусе.

1. **Тестовые наборы для всех функций преобразования над знаниями в системе**

**На вход подается название компьютерного вируса:**

Вводим: find-> Shlayer

Ответ:

year: 2019

OS: OSX

virus\_type: Trojan

is\_attached: True

easy\_detect: True

**Неудачный набор данных:**

Является ситуация при которой мы введем название вируса, которой нет в базе:

Вводим: find->SevaLocker

Ответ:

Entry with this name not found in DB

**Также при добавлении новой модели если она уже есть в базе.**

Вводим: add-> Shlayer

Ответ: Entry with same name is in DB already

**Проверка работы демона проверяющего адекватность года создания вируса:**

Please select action(find, add, remove): add

virus\_type(local, remote): local

name: StoneVirus

year: 1900

OS: Linux

virus type: Stone

is\_attached(y/n): n

easy\_detect(y/n): y

Year must be in range 1991<=year<=2021

1. **Вывод**

Были приобретены умения и навыки реализации пополняемой динамической базы знаний основанной на фреймовой системе представления знаний. И также был разработан редактор фреймов, с помощью которого можно добавлять и удалять фреймы.

**Приложение:**

**Код программы:**

from abc import ABC

from datetime import datetime

class Virus(ABC):

    def \_\_init\_\_(self, name, year, os, vtype):

        self.\_\_name = name

        self.os = os

        self.year = year

        self.vtype = vtype

    @property

    def name(self):

        return self.\_\_name

    @property

    def year(self):

        return self.\_\_year

    @year.setter

    def year(self, year):

        OS\_YEARS = {

            "Windows": 1985,

            "OSX": 2001,

            "Linux": 1991

        }

        first\_year = OS\_YEARS[self.os]

        curr\_year = datetime.now().year

        if year < first\_year or year > curr\_year:

            raise ValueError(

                f"Year must be in range {first\_year}<=year<={curr\_year}")

        self.\_\_year = year

    @property

    def os(self):

        return self.\_\_os

    @os.setter

    def os(self, os):

        OS = ['Windows', 'Linux', 'OSX']

        if os not in OS:

            raise ValueError(f"Only viruses for {', '.join(OS)} are supported")

        self.\_\_os = os

    @property

    def vtype(self):

        return self.\_\_vtype

    @vtype.setter

    def vtype(self, vtype):

        self.\_\_vtype = vtype

    def deserialize(virus\_str):

        str\_parts = virus\_str.split(',')

        if str\_parts[0] == 'L':

            return LocalVirus(str\_parts[1], int(str\_parts[2]), str\_parts[3], str\_parts[4], bool(str\_parts[5]), bool(str\_parts[6]))

        else:

            return RemoteVirus(str\_parts[1], int(str\_parts[2]), str\_parts[3], str\_parts[4], str\_parts[5])

def serialize\_bool(b):

    return '1' if b else ""

class LocalVirus(Virus):

    def \_\_str\_\_(self):

        return 'L,' + ','.join([self.name, str(self.year), self.os, self.vtype, serialize\_bool(self.is\_attached), serialize\_bool(self.easy\_detect)])

    def \_\_init\_\_(self, name, year, OS, vtype, is\_attached, easy\_detect):

        super().\_\_init\_\_(name, year, OS, vtype)

        self.is\_attached = is\_attached

        self.easy\_detect = easy\_detect

    @property

    def is\_attached(self):

        return self.\_\_is\_attached

    @is\_attached.setter

    def is\_attached(self, is\_attached):

        self.\_\_is\_attached = is\_attached

    @property

    def easy\_detect(self):

        return self.\_\_easy\_detect

    @easy\_detect.setter

    def easy\_detect(self, easy\_detect):

        self.\_\_easy\_detect = easy\_detect

class RemoteVirus(Virus):

    def \_\_str\_\_(self):

        return 'R,' + ','.join([self.name, str(self.year), self.os, self.vtype, self.attack\_vector])

    def \_\_init\_\_(self, name, year, OS, vtype, attack\_vector):

        super().\_\_init\_\_(name, year, OS, vtype)

        self.attack\_vector = attack\_vector

    @property

    def attack\_vector(self):

        return self.\_\_attack\_vector

    @attack\_vector.setter

    def attack\_vector(self, attack\_vector):

        ATTACK\_VECTORS = ['Network', 'Web']

        if attack\_vector not in ATTACK\_VECTORS:

            raise ValueError(

                f"Only {', '.join(ATTACK\_VECTORS)} attack vectors are supported")

        self.\_\_attack\_vector = attack\_vector

def parse\_db():

    db = []

    with open('db.txt') as db\_file:

        for line in db\_file.readlines():

            if len(line) > 0:

                db.append(Virus.deserialize(line.strip("\r\n ")))

    return db

def save\_db():

    with open('db.txt', 'w') as db\_file:

        db\_file.write('\n'.join(map(str, db)))

def add\_entry(e):

    if find\_entry(e.name) is None:

        db.append(e)

        save\_db()

    else:

        raise ValueError(f"Entry with same name is in DB already")

def find\_entry(name):

    for e in db:

        if e.name == name:

            return e

    return None

def remove\_entry(name):

    e = find\_entry(name)

    if e is not None:

        db.remove(e)

        save\_db()

    else:

        raise ValueError(f"Entry with this name not found in DB")

db = parse\_db()

while True:

    command = None

    while command not in ['find', 'add', 'remove']:

        command = input("Please select action(find, add, remove): ")

    if command == 'find':

        e = find\_entry(input('name: '))

        if e is not None:

            print(f"year: {e.year}\nOS: {e.os}\nvirus\_type: {e.vtype}")

            if isinstance(e, LocalVirus):

                print(

                    f"is\_attached: {e.is\_attached}\neasy\_detect: {e.easy\_detect}")

            else:

                print(f"attack\_vector: {e.attack\_vector}")

        else:

            print(f"Entry with this name not found in DB")

    elif command == 'add':

        vclass = None

        while vclass not in ['local', 'remote']:

            vclass = input("virus\_type(local, remote): ").lower()

        name = input('name: ')

        year = input('year: ')

        OS = input('OS: ')

        vtype = input('virus type: ')

        if vclass == 'local':

            is\_attached = input('is\_attached(y/n): ') == 'y'

            easy\_detect = input('easy\_detect(y/n): ') == 'y'

            try:

                add\_entry(LocalVirus(name, int(year), OS,

                                     vtype, is\_attached, easy\_detect))

            except ValueError as e:

                print(e)

        else:

            attack\_vector = input('attack vector: ')

            try:

                add\_entry(RemoteVirus(name, int(year),

                                      OS, vtype, attack\_vector))

            except ValueError as e:

                print(e)

    else:

        try:

            remove\_entry(input('name: '))

        except ValueError as e:

            print(e)

**Содержимое db.txt**

L, Shlayer,2019,OSX,Trojan,1,1

L,Nimda,2001,Windows,Trojan,,1

L,ILOVEYOU,2001,OSX,Worm,1,1

R,SQL Slammer,2003,Linux,Exploit,Network

R,CryptoLocker,2018,Windows,Ransomware,Network

R,Coinhive,2020,Windows,Cryptojacker,Web