РЕФЕРАТ / ABSTRACT

Пояснювальна записка до дипломної роботи:

Метою роботи є вивчення певних теоретичних засад побудови штучних імунних алгоритмів.

Методи розробки базуються на технології С++ без використання сторонніх засобів та бібліотек.

Об’єктом розробки є штучна імунна система, як засіб боротьби з комп’ютерними вірусами.

Результатом роботи є програмна реалізація штучної імунної системи для виявлення та нейтралізації шкідливих програм.

Розроблена система детально протестована на операційній системі Windows засобами QtTests та VSTests.

ШТУЧНА ІМУННА СИСТЕМА, ІМУННИЙ ДЕТЕКТОР, ІМУННИЙ АЛГОРИТМ, КОМП’ЮТЕРНІ ВІРУСИ, АЛГОРИТМ КЛОНАЛЬНОЇ СЕЛЕКЦІЇ.

The aim of the work is to study certain theoretical principles of constructing artificial immune algorithms.

Development methods are based on C ++ and Assembler technologies.

The object of development is an artificial immune system as a means of combating computer viruses.

The result is a software implementation of an artificial immune system to detect and neutralize malware.

The developed system is tested in detail on the Windows operating system by means of QtTests and VSTests.

ARTIFICIAL IMMUNE SYSTEM, IMMUNE DETECTOR, IMMUNE ALGORITHM, COMPUTER VIRUSES, CLONAL SELECTION ALGORITHM.

ЗМІСТ

[РЕФЕРАТ / ABSTRACT 3](#_Toc41684449)

[ЗМІСТ 4](#_Toc41684450)

[ВСТУП 5](#_Toc41684451)

[1 АНАЛІТИЧНИЙ ОГЛЯД СУЧАСНИХ АНТИВІРУСНИХ ПРОГРАМ 8](#_Toc41684452)

[1.1 Аналіз популярних антивірусних програм 8](#_Toc41684453)

[1.2 Недоліки та переваги популярних антивірусних програм 10](#_Toc41684454)

[1.3 Статистика популярних антивірусних рішень 11](#_Toc41684455)

[1.4 Висновки щодо проведення аналітичного огляду сучасних антивірусних засобів 12](#_Toc41684456)

[2 ВІДОМОСТІ ПРО ОБ’ЄКТ РОЗРОБКИ 13](#_Toc41684457)

[2.1 Поняття імунної системи (ІС) та штучної імунної системи (ШІС) 13](#_Toc41684458)

[2.2 Основні поняття 13](#_Toc41684459)

[2.3 Історія розвитку 13](#_Toc41684460)

[2.4 Принцип роботи 14](#_Toc41684461)

[2.5 Методи 15](#_Toc41684462)

[2.6 Моделі 16](#_Toc41684463)

[3 ОПИС МЕТОДІВ ТА ЗАСОБІВ ВИРІШЕННЯ ЗАДАЧІ 19](#_Toc41684464)

[3.1 Опис засобів програмної реалізації 19](#_Toc41684465)

[3.1.1 Опис мови 19](#_Toc41684466)

[3.1.2 Опис стандартної бібліотеки 20](#_Toc41684467)

[3.1.3 Переваги мови 21](#_Toc41684468)

[3.1.4 Недоліки мови 21](#_Toc41684469)

[3.1.5 Висновки щодо обраних засобів реалізації 22](#_Toc41684470)

[3.2 Опис методів реалізації 23](#_Toc41684471)

[3.2.1 Загальний алгоритм функціонування штучної імунної системи 23](#_Toc41684472)

[3.2.2 Структура та алгоритм навчання нейромережевого імунного детектора 26](#_Toc41684473)

[3.2.3 Алгоритм функціонування нейромережевого імунного детектора 32](#_Toc41684474)

[3.3 Архітектура програмного засобу 32](#_Toc41684475)

ВСТУП

Штучні імунні системи беруть свій початок з середини 1980-х років в дослідженнях Фармера, Пакарда та Перелсона у 1986-му, а також в дослідженнях Варела в 1990 р.В 1994 роціФоррест і Ке-пхарт опублікували свої перші наукові результати стосовно побудови штучних імунних систем. В тому ж році Дасгупта опублікував результати досліджень стосовно алгоритму негативного відбору (NegativeSelectionAlgorithm), який моделює один з механізмів дії природної імунної системи. Теоретичними засадами побудови і функціонування штучних імунних систем займались відомі вчені: Хант, Кук, Тімміс, Ніл, Де Кастро, фон Зубен, Нікосія,Кутелло та інші. Штучна імунна система є також адаптивною обчислювальною системою, що використовує моделі, принципи, механізми та функції, описані в теоретичній імунології. З класичної імунології відомо, що при попаданні антигену в організм дуже мала частина клітин імунної системи здатна до його розпізнавання. Таке розпізнавання стимулює процеси розмноження і диференціювання лімфоцитів, що призводять до утворення клонів ідентичних клітин (антитіл). Цей процес розмноження клону формує численну популяцію специфічних до антигену антитіло-продукуючих клітин. Розмноження клону імунокомпетентних клітин призводить до руйнування або нейтралізації антигену. Частина новостворених клітин зберігається для імунної пам'яті. В результаті, подальший вплив схожого антигену призводить до більш швидкої імунної реакції (вторинної відповіді). Процес циркуляції В-і Т-лімфоцитів в первинних і вторинних лімфоїдних органах ретельно контролюється, що забезпечує попадання відповідних клітинних популяцій, які називають наївними, тобто «ненавченими», ефекторних клітин, а також клітин пам'яті, в різні місця призначення. Вибіркова міграція лімфоцитів в різні органи і тканини має назву хоумінга (самонаведення). У цих органах є спеціалізоване середовище для підтримки процесів розмноження активованих антигеном лімфоцитів та їх диференціювання в ефекторні клітини і клі-тини пам'яті.

Сучасні антивірусні продукти являють собою складні програмні модулі, тісно інтегровані в ядро операційної системи і працюючі з нею як одне ціле. Це не тільки сканери, виконуючі простий пошук вірусних програм по сигнатурі, але і евристичні аналізатори, мережеві екрани, ревізори та ін.

З моменту появи нового вірусу до появи відповідної реакції на цей вірус з боку антивірусної індустрії може пройти деякий, іноді довгий час. Як показала практика, за це час вірус в змозі заразити сотні тисяч комп’ютерів, викликати справжню вірусну епідемію і принести великі втрати, як інформації так і коштів. Сучасні дослідження в галузі захисту інформації направлені на створення таких методів і алгоритмів захисту, які будуть здатні виявляти і нейтралізовувати невідомі віруси.

Методи штучних імунних систем дозволяють побудувати таку систему виявлення вірусів, яка буде здатна виявляти невідомі шкідливі програми. Основними елементами такої штучної імунної системи є детектори, які буду виконувати фунції виявлення та нейтралізації.

Антивірусні програми працюють за добре відомими ознаками шкідливих програм – сигнатур, які зберігаються в базі антивірусних сигнатур як еталони з якими в здійснюється порівняння для подальшої виявлення. Якщо в даний момент часу сигнатура тої чи іншої шкідливої програми відсутня в базі сигнатур, то антивірусна програма її не виявить і вона може довгий час існувати в системі.

Процедура отримання сигнатури вірусу потребує затрат часу і ретельного аналізу досліджуваної програми з ціллю отримання її ознак, а також додаванням її в базу сигнатур. Сьогодні з розвитком мережі Інтернет швидкість поширення різної інформації і програмного забезпечення значно збільшилася, отже, нові або модифіковані шкідливі програми можуть поширитися за невелику кількість часу на безліч комп’ютерів і завдати величезної шкоди.

Евристичні методи дозволяють отримувати сигнатури вірусів яких ще немає в базі сигнатур. Для використання евристичного підходу виявлення вірусів необхідно організувати збір та аналіз великої кількості інформації яка описує стан компонентів системи яку потрібно захистити. Проте через те, що при евристичному шукаються не шкідливі програми як такі, а об’єкти схожі на них, можливі помилкові спрацьовування. Проте саме таким шляхом можна визначити ще невідомі та поліморфні вірусні програми.

Моніторинг змін є одним з найдавніших методів, за яким відстежуються зміни параметрів об’єктів в системі. При цьому знову потрібен попередній збір даних про незаражену систему, що значно знижує швидкодію комп’ютера. Помилкові спрацьовування неминучі.

Моніторинг поведінки. За цим методом виявляють невідому або поліморфні віруси, визначаючи їх за шкідливими діями. Негативними рисами такого підходу є помилкові спрацьовування на не шкідливі програми та підвищені вимоги до ресурсів комп’ютера.

Як би добре і швидко не працювали антивірусні програми, поліфаги та сканери. Якими б великими і частооновлюваними не були їх бази даних з відомими вірусами, розробниками просто не встигнути повністю обробити та внести до бази сигнатури, це обумовлено великою кількістю та швидкістю створення шкідливого програмного забезпечення. Також проблема ускладнюється поліморфністю та інтелектуальністю нових вірусних розробок. Наявність спеціальних генераторів вірусів з доступним будь-кому інтерфейсом також сприяє тому, що за створення вірусів беруться численні зловмисники.

Одним з перспективних сучасних рішень задачі оперативного виявлення комп’ютерних вірусів можна вважати побудову інтелектуальних імунних систем. Ідея виникнення таких систем з’явилась в результаті вивчення процесів біологічного імунітету, котрий захищає організм від вірусів та шкідливих бактерій біологічного характеру. Імунна реакція заключається в стимуляції різних механізмів при виявленні шкідливої бактерії. Слід зазначити, що імунна система здатна не тільки розпізнавати вже відомі їй бактерії та віруси, але також і невідомі їй.

У зв'язку з цим має актуальність розробка системи виявлення шкідливих програм на основі інтелектуальних імунних методів, яка на різних етапах аналізує поведінку шкідливої ​​програми і виявляє її, що дозволяє значно підвищити ефективність виявлення шкідливих програм і їх модифікацій, адаптувати додаткові засоби захисту інформації своєчасно локалізувати загрозу, тим самим підвищивши рівень захищеності в цілому.

Метою роботи є розробка штучної імунної системи для виявлення шкідливих програм на персональному комп’ютері.

Мета обумовила вибір наступних завдань:

* дослідити сутність штучних імунних систем та алгоритмів та їх шляхи реалізації;
* провести аналіз методів та технологій розробки штучних імунних систем;
* обрати технологію розробки для створення штучної імунної системи;
* сформувати функціональну структуру системи для виявлення та знешкодження

шкідливих програм на комп’ютері;

* розробити програмний продукт та провести його тестування.

1. АНАЛІТИЧНИЙ ОГЛЯД СУЧАСНИХ АНТИВІРУСНИХ ПРОГРАМ
   1. Аналіз популярних антивірусних програм

Для захисту і боротьби з вірусами застосовуються антивірусні програми, які традиційно поділяють на кілька видів:

Програми-ревізори — це найбільш надійні засоби захисту від вірусів. За ними спочатку запам'ятовуються відомості про стан програм, каталогів і системних ділянок дисків коли комп'ютер не заражений вірусом, а потім проводиться порівняння поточного стану з вихідним. Виявлені зміни виводяться на екран монітора. Як правило, порівняння станів відбувається відразу після завантаження операційної системи. При порівнянні перевіряються довжина файлу, код циклічного контролю (контрольна сума файлу), дата і час модифікації, інші параметри. В програмах-ревізорах реалізуються досить розвинені алгоритми, за якими виявляються стелс-віруси і можна навіть відмінити зміни в програмі, спричинені вірусом. Нажаль деякі види шкідливих програм, наприклад, зломщики паролів, клавіатурні шпигуни, за допомогою ревізора виявити неможливо. Програми-ревізори — ADinf, DLI, The Nicks Ghost Buster, UNA34. Серед них хочеться відокремити ADinf.

За допомогою ревізора ADinf можна виявити наступні віруси:

* Stealth-віруси;
* віруси-мутанти;
* ще невідомі на сьогоднішній день віруси (за рахунок використання передових

досягнень в області штучних інтелектуальних систем).

У розширенні ревізора Adinf— програмі ADinf Cure Module (файл ADinfExt.exe) додатково міститься невелика база даних, в якій описуються файли, які зберігаються на диску.

Програми-доктори (фаги) - призначені для "лікування" заражених програм та дисків шляхом ліквідації віруса. На початку функціонування фагів проводиться пошук вірусів в оперативній пам'яті, знайдені віруси ліквідуються і лише потім відбувається "лікування" файлів. Відомі програми-доктори — Clean-Up, M-Disk, LD, SOS, TNTVirus, AV, Dr.Web, GriSoft, AVG AntiVirus System, Virus Cleaner та ін. Серед фагів виділяють поліфаги, тобто програми-доктори, призначені для пошуку і ліквідації великої кількості вірусів. Найбільш відомі з них: Aidstest, Scan, Norton AntiVirus, Doctor Web; Серед них можна хочеться відокремити Doctor Web та Norton AntiVirus.

За допомогою Doctor Web можна успішно позбавлятися відомих поліморфних вірусів. Крім того, використовуючи Doctor Web можна проводити евристичний аналіз файлів з метою виявлення невідомих поліморфних вірусів, вірусів, що складно шифруються, і т.ін. Успіх такого аналізу — в середньому 82%. За допомогою програми можна розпаковувати і перевіряти виконувані файли, а також файли, архівовані за допомогою архіваторів LZEXE, PKLite, Diet та ін. Характерною особливістю антивіруса Dr. Web є можливість установки на заражену машину. В процесі установки проводиться сканування пам'яті і файлів автозавантаження, перед скануванням проводиться оновлення вірусної бази. При цьому випуски оновлень вірусних баз з'являються кожні декілька годин, або й частіше.

Origins Tracing — алгоритм несигнатурного виявлення шкідливих об'єктів, який доповнює традиційні сигнатурний пошук і евристичний аналізатор, дає можливість значно підвищити рівень виявлення раніше невідомих шкідливих програм.

Dr. Web Shield — механізм боротьби з руткітами, реалізований у вигляді драйвера компонент антивірусного сканера, забезпечує доступ до вірусних об'єктів, що ховаються в глибинах операційної системи. Підтримка більшості існуючих форматів упакованих файлів і архівів, у тому числі, багатотомних архівів, що саморозпаковуються. У цей час є підтримка близько 4000 видів різних архівів і пакувальників.

Оновлення вірусних баз виробляються негайно у міру виявлення нових вірусів, до декількох разів на годину. Розробники антивірусного продукту відмовилися від випуску оновлень вірусних баз за графіком, оскільки вірусні епідемії йому не підкоряються. Компактна вірусна база і невеликий розмір оновлень. Один запис у вірусній базі дозволяє визначати десятки, а часто й тисячі подібних вірусів.

* невеликий розмір дистрибутива.
* кросплатформність — використовується єдина вірусна база і єдине ядро антивірусного сканера.
* можливість повноцінної роботи сканера без інсталяції, що дозволяє використовувати антивірус для лікування заражених систем з використанням носіїв в режимі лише для читання.
* виявлення і лікування складних поліморфних, шифрованих вірусів і руткітів

Norton Antiirus — пакет, призначений для захисту комп'ютера від вірусів під час роботи в Інтернеті, обміну файлами через мережу, завантаження файлів з дискет і CD. Тепер за допомогою програми можна автоматично сканувати вхідні поштові повідомлення, що містять різного роду прикріплені дані, в таких популярних поштових програмах, як MS Outlook, MS Outlook Express, Eudora Pro, Eudora Lite, Netscape Messenger, Netscape Mail. За програмою захищається система від небезпечних ActiveX-кодів, Java-аплетів і так званих "троянців", а також: визначаються і ліквідуються віруси зі стиснутих файлів (у тому числі і з багаторазово стиснутих файлів). Підтримувані формати MIME/UU, LHA/LZH, ARJ, CAB, PKLite, LZEXE, ZIP. За допомогою програми можна перевіряти рівень безпеки бездротових мереж (корисна фунція для користувачів, що під'єднують свої комп'ютери до мереж: через Wi-Fi).

Доктори-ревізори — це програми, в яких поєднуються властивості ревізорів і фагів: з їх допомогою можна знайти зміни у файлах і системних ділянках дисків та, за необхідності, у випадку патологічних змін, можна автоматично повернути файл до вихідного стану. Приклади: InoculateIT Personal Edition, McAfee VirusScan, Panda Antivirus Platinum;

В антивірусний сканер McAfee VirusScan нових версій включено функцію автоматичного оновлення вірусних баз даних через Інтернет, вбудовано мережевий екран і нову функцію Hostile Activity Watch Kernel (HAWK), за якою користувача попереджають про спроби масового розсилання електронної пошти з його комп'ютера і про одержання в листі приєднаного файлу з подвійним розширенням, що є явною ознакою вірусів. А за допомогою вбудованого мережевого екрану проводиться фільтрація вхідного Інтернет-трафіка і виявляються спроби вторгнення.

* 1. Недоліки та переваги популярних антивірусних програм

Багато безкоштовних антивірусних пакетів мають функціонал, який мало чим поступається їх платним конкурентам. Відмінності між різними антивірусами полягають в ціні програми, комплектації пакета, функціональні можливості програми та системних вимогах.

Основними недоліками відомих безкоштовних антивірусних програм є:

* збільшення зростання споживання системних ресурсів при виставленні максимальних

налаштувань безпеки;

* сигнатурні бази малого об’єму;
* сканер антивірусу не працює із запакованими і виконавчими файлами;
* для роботи зі статистикою додатки потрібно встановити Flash Player, який дуже

активно використовують для атак на браузер, що, звичайно ж, робить систему більш вразливою;

* помилкові спрацьовування;
* слабкі евристичні алгоритми.

Проте безкоштовні версії мають і сильні сторони:

* швидко працюючі резидентні частини;
* простий та зрозумілий інтерфейс;
* можливість гнучкого налаштування системи;
* захист від багатьох шкідливих програм.
  1. Статистика популярних антивірусних рішень

Рейтинг найнадійніших антивірусів для домашнього використання на Windows. Рейтинг сформувала незалежна лабораторія AV-TEST.

Так, найкращим антивірусом експерти назвали вбудований у Windows продукт Microsoft Defender. Він сумарно набрав максимальні 18 балів – найвищі оцінки експерти віддали за рівень захисту, продуктивність та інтерфейс.

А от найгіршим антивірусом експерти назвали Webroot SecureAnywhere. Він набрав всього 11,5 балів. Антивірус отримав 2 бали за захист, 5,5 балів за продуктивність та 4 бали за інтерфейс.

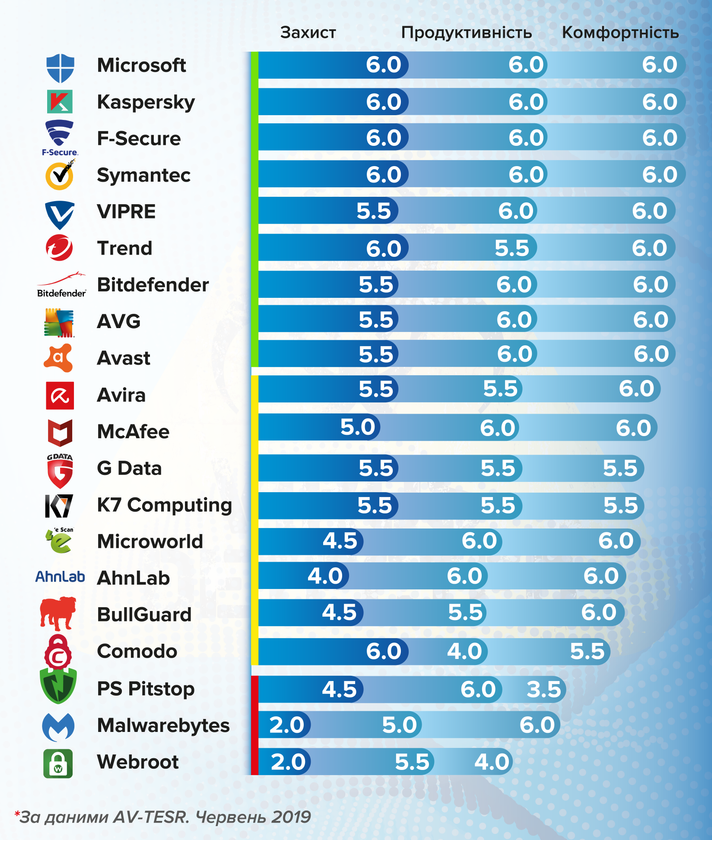


Рисунок 1.1 – Рейтинг найнадійніших антивірусів для домашнього використання на Windows

Цікавою є статистика, щодо виявлених загроз у економічно розвинених регіонах України. Так, користувачі ПК у м. Київ, Дніпропетровській, Харківській, Донецькій, Львівській областях найчастіше були атаковані троянськими програмами сімейства Trojan.Agent.Win32. Дані троянці створені для крадіжки конфіденційних даних користувачів, в тому числі паролів доступу до Інтернет ресурсів, номерів кредитних карт, іншої інформації, яку можна використати для крадіжки приватних фінансів користувачів.

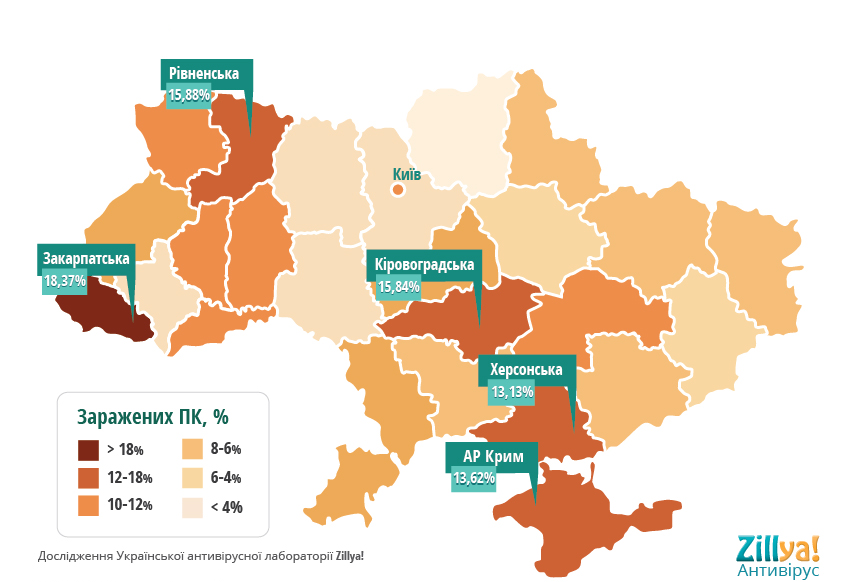


Рисунок 1.2 ­– Рівень зараження регіонів України комп’ютерними вірусами

### Висновки щодо проведення аналітичного огляду сучасних антивірусних засобів

На основі розглянутих антивірусних програм можна зробити висновок, що кожна з них має свої особливості, як переваги так і недоліки. На сьогодні існує невелика кількість обчислювальних моделей, заснованих на принципах роботи імунної системи. Це пов'язано з відсутністю однозначного погляду на основні положення побудови штучних імунних алгоритмів, а також неповною їх визначеністю. Найуживанішими на сьогодні є моделі імунної мережі, алгоритм негативного відбору і імунні мережеві алгоритми.

1. ВІДОМОСТІ ПРО ОБ’ЄКТ РОЗРОБКИ
   1. Поняття імунної системи (ІС) та штучної імунної системи (ШІС)

Біологічна імунна система являє собою складну розподілену адаптивну систему інтелектуальної обробки певного інформаційного простору і призначена захищати окремий організм зовнішніх вірусів і інфекцій. Ця система здатна до самонавчання і розв’язання, зокрема, задач класифікації і пошуку, аналізуючи і перетворюючи величезні обсяги інформації.

ШІС — це адаптивна обчислювальна система, що використовує моделі, принципи, механізми та функції, описані в теоретичній імунології, які застосовуються для розв'язання прикладних задач.

* 1. Основні поняття

Природна імунна система являє собою складну систему, що складається з декількох функціонально різних компонентів. ІС використовує багаторівневий захист проти зовнішніх антигенів, включаючи дію неспецифічних і специфічних захисних механізмів. Основна роль імунної системи полягає у розпізнаванні чужорідних клітин (або молекул) організму. ІС не розпізнає весь хвороботворний організм цілком. Розпізнавання відбувається на рівні окремих молекул мікроба, які називають антигенами. Основним елементом, що забезпечує функціонування імунітету, є В і Т-клітини (лімфоцити). Вони містять на своїй поверхні спеціальні молекули (рецептори) — антитіла. Антитіла є безпосередніми виконавцями процесу розпізнавання чужорідних антигенів. Також вводиться поняття афітності (схожості) антитіла з антигеном. Це дозволяє імунній системі контролювати кількість лімфоцитів в організмі.

* 1. Історія розвитку

ШІС почало своє існування в середині 1980-х років з досліджень Фармера, Пакарда та Перелсона у 1986му, а також роботою Варела (1990). Форрест і Кепхарт опублікували свої перші роботи про ШІС в 1994 році. У цьому ж році Дасгупта виконує дослідження Negative Selection Algorithm. Хант і Кук почали роботи по моделюванню імунної мережі в 1995 році. Пізніше Тімміс і Ніл продовжили цю роботу і зробили кілька поліпшень. Перша книга по ШІС була видана під редакцією Дасгупти в 1999 році. Де Кастро і фон Зубен, а також Нікосії і Cutello в 2002 році виконали дослідження на тему: clonal selection вирішення проблем біоінформатики.

У 2008 році Дасгупта та Ніно опублікували підручник з імунологічних обчислень, який являє собою збірник до сучасних робіт, пов'язаних з імунітетом на основі методів ШІС.

В даний час вивчаються нові ідеї і алгоритми, що походять від імунної системи. Проте існують деякі сумніви, що вони ще не будуть пропонувати нічого понад існуючих алгоритмів ШІС.

* 1. Принцип роботи

При попаданні антигену в організм лише мала частина клітин імунної системи здатна до його розпізнавання. Таке розпізнавання стимулює процеси розмноження і диференціювання лімфоцитів, що призводять до утворення клонів ідентичних клітин (або антитіл). Цей процес, званий розмноженням клону, формує численну популяцію специфічних до антигену антитіл і продукуючих клітин. Розмноження клону імунокомпетентних клітин призводить до руйнування або нейтралізації антигену. Частина новостворених клітин зберігається для імунної пам'яті. В результаті, подальший вплив схожого антигену призводить до більш швидкої імунної реакції (вторинної відповіді).

Процес циркуляції В-і Т-лімфоцитів в первинних і вторинних лімфоїдних органах ретельно контролюється, що забезпечує попадання відповідних клітинних популяцій - наївних («ненавчених») і ефекторних клітин, а також клітин пам'яті - в різні місця призначення. Виборча міграція лімфоцитів в різні органи і тканини має назву хоумінга (самонаведення). У цих органах є спеціалізоване середовище для підтримки процесів розмноження активованих антигеном лімфоцитів та їх диференціювання в ефекторні клітини і клітини пам'яті. Цікаво, що клітини пам'яті надають виборчу перевагу тому типу тканин, в якому вони вперше зустрілися з антигеном. Ймовірно, це забезпечує повернення кожної конкретної клітини пам'яті в ту ділянку тіла, де вона, швидше за все, знову зустрінеться з антигеном.

Існує два основних варіанти імунної відповіді: гуморальний, опосередкований В-клітинами і їх продуктами, і клітинний, в якому беруть участь Т-клітини. Обома варіантами імунних реакцій відповідають подібні послідовності етапів захисту організму: активація, поділ, диференціювання, секреція, імунна атака, супресія і пам'ять, проте вони здійснюються різними способами. У регуляції як гуморального, так і клітинного імунітету беруть участь популяції Т-клітин, звані хелперними або супресорними клітинами, які посилюють або, відповідно, пригнічують імунну відповідь.

Імунна система являє великий інтерес як система, здатна ефективно обробляти значні обсяги даних. Зокрема, вона виконує великий обсяг складних високо паралельних розподілених обчислень. Поведінка імунної системи в цілому визначається всією сукупністю локальних взаємодій. Імунна система функціонує як «другий мозок», оскільки здатна зберігати інформацію про інтенсивність попередніх контактів складових її клітин і відповідати на нові, раніше не зустрічалися, структури (антигени). Опис закономірностей розвитку імунної відповіді являє собою цікаву задачу теорії динамічних систем. У відношенні можливих додатків для обробки інформації перспективні наступні властивості імунної системи. Розпізнавання. Імунна система здатна розпізнавати та класифікувати різні молекулярні структури і вибірково на них реагувати.

* 1. Методи

Розпізнавання відбувається в ході міжклітинних контактів, при цьому утворюються багато зав’язків, які визначаються формою молекул і величиною електростатичного заряду. Розпізнавання свого і чужого є однією з основних завдань, яке вирішує імунна система.

Виділення особливостей. Антиген-презентуючі клітини (АПК) інтерпретують антигенне оточення і виділяють особливості шляхом обробки антигенів та подання антигенних пептидів на своїй поверхні. Кожна АПК служить в якості «фільтра», що переважає молекулярний шум, і «збільшувального скла», що фокусує увагу лімфоцитів-рецепторів.

Різноманітність. Імунна система використовує комбінаторний механізм (генетично-обумовлений процес) для утворення безлічі різних рецепторів лімфоцитів з тим, щоб гарантувати, що хоча б один лімфоцит з усієї сукупності зможе провзаємодіяти з будь-яким наперед заданим (відомим або невідомим) антигеном.

Навчання. Імунна система оцінює структуру конкретного антигену, використовуючи його випадкові контакти зі складовими цю систему клітинами. Навчання полягає у зміні концентрації лімфоцитів, яке відбувається при первинній відповіді (у результаті першого контакту з антигеном). Отже, здатність імунної системи до навчання закладена, головним чином, у механізмі поповнення клонів, що приводить до утворення нових імунокомпетентних клітин з урахуванням поточного стану системи (цей процес має назву розмноження клону).

Пам'ять. Невелика частина лімфоцитів, що знаходяться в активованому стані, стає клітинами пам'яті (асоціативна пам'ять). Вважається, що час життя клітин пам'яті є динамічною величиною і визначається частотою стимуляції антигенами. Використовуючи короткочасні і довгострокові механізми імунної пам'яті, імунна система підтримує ідеальний баланс між економією ресурсів і виконанням функції за рахунок збереження мінімально необхідної, але достатньої пам'яті про попередніх контактах з антигеном.

Розподілений пошук. За своєю суттю імунна система - це розподілена система. Клітини імунної системи, головним чином лімфоцити, безперервно циркулюють через кров, лімфу, лімфоїдні органи та інші тканини. У разі зустрічі з антигеном вони здійснюють специфічну імунну відповідь.

Саморегуляція. Імунний захист має властивість саморегуляції. Центрального органу, контролюючого функції імунної системи, не існує. Залежно від способу проникнення в організм і інших властивостей антигену, регуляція імунної відповіді може бути як локальної, так і системною.

Пороговий механізм. Імунна відповідь і розмноження імунокомпетентних клітин відбуваються лише після подолання деякого порогу, що залежить від сили хімічних зв'язків.

Спільна стимуляція. Активація В-лімфоцитів жорстко регулюється за допомогою додаткового стимулюючого сигналу. Другий сигнал (від хелперних Т-лімфоцитів) допомагає забезпечувати толерантність і проводити розходження між серйозною загрозою і «хибним дзвінком» (тобто небезпечними і безпечними антигенами).

Динамічний захист. Клональне розмноження і соматичне гіпермутування дозволяють імунній системі продукувати високоаффіні імунокомпетентні клітини (цей процес називається збільшенням афінності), що створює динамічний баланс між вивчаючою і захисною функцією адаптивного імунітету. Наявність динамічного захисту поступово призводить до розширення зони спостереження, контрольованої імунною системою.

Імовірнісне виявлення. Перехресні реакції в ході імунної відповіді - це процес стохастичний. До того ж виявлення антигену завжди неминуче відбувається приблизними чином; отже, лімфоцит може взаємодіяти з декількома структурно подібними антигенами.

* 1. Моделі

В імунній відповіді на антиген важливу роль відіграють і інші характеристики імунної системи, такі як адаптованість, специфічність, самотолерантність, диференціювання та інші. Всі ці властивості, що мають відношення до обробки інформації, створюють ряд цікавих можливостей з обчислювальної точки зору.

Для пояснення імунологічних механізмів існують різні теорії та математичні моделі. Також є зростаюче число комп'ютерних моделей для імітації динаміки різних компонентів імунної системи та її поведінки в цілому. Ці підходи включають моделі, сформульовані у вигляді систем диференціальних та стохастичних рівнянь, клітинно-автоматні моделі, моделі простору конфігурацій та інші. Разом з тим, природна імунна система служить джерелом нових ідей для розвитку інтелектуальних методів вирішення складних завдань, але робіт в цій області поки небагато. Необхідно проводити більше досліджень, зокрема, для вивчення механізмів обробки інформації в імунній системі, що може мати велике практичне значення.

На жаль, в даний час існує лише невелика кількість обчислювальних моделей, заснованих на принципах роботи імунної системи. Мабуть, це пов'язано з невизначеністю основних положень, запропонованих для її опису. Серед таких моделей часто використовуються наступні.

Модель імунної мережі. Ерне запропонував гіпотезу, згідно з якою імунна система являє собою регульовану мережу молекул і клітин, що розпізнають один одного навіть за відсутності антигену. Такі структури часто називають ідіотипічними мережами, вони служать математичною основою для вивчення поведінки імунної системи. Теорія Ерне інтерпретується у вигляді системи диференціальних рівнянь, що описує динаміку концентрації клонів лімфоцитів і відповідних молекул імуноглобулінів. Теорія ідіотипічної регуляції заснована на припущенні, що різні клони лімфоцитів один від одного не ізольовані, а підтримують зв'язок шляхом взаємодій між своїми рецепторами і антитілами. Отже, розпізнавання антигену здійснюється не поодиноким клоном клітин, а скоріше на системному рівні, за участю різних клонів, взаємодіючих за типом реакцій антиген-антитіло як єдина мережа.

Ерне вважав, що в ході імунної відповіді сам антиген викликає лише вироблення першого набору антитіл Аb1. Потім ці антитіла, діючи в якості антигену, викликають вироблення другого набору «антіідіотипічних» антитіл Ab2, які розпізнають ідіотипи на антитіла Аb1. Аналогічно здійснюється вироблення третій набір антитіл, Ab3, які розпізнають антитіла Ab2, і так далі.

Ключовий постулат теорії Ерне полягає в тому, що одинична клітина продукує лише один тип антитіл.

Формулюючи основи своєї теорії, Ерне ввів поняття формальних і функціональних мереж. Формальні мережі служать для вивчення питань репертуару, дуалізму та супресії. При розгляді функціональних мереж представлена кількісна картина теорії.

Імовірнісний підхід до вивчення ідіотипічних мереж на основі роботи Ерне запропонував Перельсон. Даний підхід гранично математизованих і, в основному, пов'язаний з описом фазових переходів. Перельсон розділив площину фазових змінних розглянутої системи рівнянь на докритичну область, область переходу і посткритичну область.

Останні 20 років запропонованій Ерне теорії імунної мережі приділялась значна увага, що привело до докладного вивчення багатьох обчислювальних аспектів відповідних математичних моделей або алгоримтів.

Алгоритм негативного відбору. Форрест запропонував алгоритм негативного відбору для виявлення змін, побудований на основі принципів розпізнавання свого і чужого в системі імунітету. В імунній системі таке розпізнавання забезпечується Т-лімфоцитами й іншими клітинами, що мають на своїй поверхні рецептори, здатні виявляти чужорідні білки (антигени). Рецептори створюються на основі псевдовипадкового генетично обумовленого процесу перегрупування в ході утворення Т-клітин. Потрапляючи в тимус, Т-клітини піддаються цензуруванню, званому негативним відбором, при цьому клітини, які вступили в реакцію з власними білками, знищуються, а інші (не утворюють з ними зв'язків) отримують можливість покинути тимус. Потім ці Т-клітини циркулюють по всьому організму і виконують функцію захисту від чужорідних антигенів. Аналогічно діє алгоритм негативного відбору, випадковим чином створюючи детектори і видаляючи ті з них, які розпізнають своє, так що залишаються детектори можуть виявляти будь-яке не своє.

Імунні мережеві алгоритми — алгоритми, що користаються теорією ідеотипічних мереж, запропонован Нільсом Кай Джерном, яка описує регуляції імунної системи за допомогою ідеотипічних антитіл (антитіл, які вибирають для інших антитіл). Цей клас алгоритмів сфокусовано на мережевому графі структур, де антитіла (або антитіла, які продукують клітини) є вузли та алгоритм навчання передбачає зростання або скорочення відстаней між вузлами на основі близькості (подібності в просторі подання проблеми). Імунні мережеві алгоритми були використані в кластеризації, візуалізації даних, контролі та оптимізації областей, а деякі — для розробки штучних нейронних мереж .

Клональний алгоритм вибору — клас алгоритмів, що використовують методи клонової селекції і теорію придбаного імунітету, який пояснює, як В і Т- лімфоцити покращують реакцію на антигени. Ці алгоритми основані на деяких атрибутах теорії Дарвіну, в якій вибір продиктований взаємодією антигенів з антитілами, та репродукції за принципом ділення клітин або за методом соматичної гіпермутації.

1. ОПИС МЕТОДІВ ТА ЗАСОБІВ ВИРІШЕННЯ ЗАДАЧІ

### Опис засобів програмної реалізації

* + 1. Опис мови

Для реалізації програми було обрано мову програмування С++. Це універсальна мова програмування високого рівня з підтримкою декількох парадигм програмування. Зокрема: об’єктно-орієнтованої та процедурної. Розроблена Б’ярном Страуструпом (англ. Bjarne Stroustrup) в AT&T Bell Laboratories (Мюррей-Хілл, Нью-Джерсі) у 1979 році та названа «С з класами». Страуструп перейменував мову у C++ у 1983 р. Базується на мові Сі. Визначена стандартом ISO/IEC 14882:2003.

У 1990-х роках С++ стала однією з найуживаніших мов програмування загального призначення.

При створенні С++ прагнули зберегти сумісність з мовою С. Більшість програм на С справно працюватимуть і з компілятором С++. С++ має синтаксис, заснований на синтаксисі С.

Нововведеннями С++ порівняно з С є:

* підтримка об'єктно-орієнтованого програмування через класи;
* підтримка узагальненого програмування через шаблони;
* доповнення до стандартної бібліотеки;
* додаткові типи даних;
* обробка винятків;
* простори імен;
* вбудовані функції;
* перевантаження операторів;
* перевантаження імен функцій;
* посилання і оператори управління вільно розподіленою пам'яттю.

В 1998 році мова Сі++ була стандартизована Міжнародною організацією стандартизації під номером 14882:1998 — Мова Програмування Сі++. В даний час робоча група МОС працює над новою версією стандарту під кодовою назвою C++09 (раніше відомий як C++0X), який має вийти в 2009 році.

Стандарт Сі++ на 1998 рік складається з двох основних частин: ядра мови і стандартної бібліотеки. Стандартна бібліотека Сі++ увібрала в себе бібліотеку шаблонів STL, що розроблялася одночасно із стандартом. Зараз назва STL офіційно не вживається, проте в кругах програмістів на Сі++ ця назва використовується для позначення частини стандартної бібліотеки, що містить визначення шаблонів контейнерів, ітераторів, алгоритмів і функторів.

Стандарт Сі++ містить нормативне посилання на стандарт Сі від 1990 року і не визначає самостійно ті функції стандартної бібліотеки, які запозичуються із стандартної бібліотеки Сі.

Поза тим, існує величезна кількість бібліотек Сі++, котрі не входять в стандарт. У програмах на Сі++ можна використовувати багато бібліотек Сі.

Стандартизація визначила мову програмування Сі++, проте за цією назвою можуть ховатися також неповні, обмежені достандартні варіанти мови. Спочатку мова розвивалася поза формальними рамками, спонтанно, у міру завдань, що ставилися перед ним. Розвиток мови супроводив розвиток кросс-компілятора Cfront. Нововведення в мові відбивалися в зміні номера версії кросс-компілятора. Ці номери версій кросс-компілятора розповсюджувалися і на саму мову, але стосовно теперішнього часу мову про версії мови Сі++ не ведуть.

* + 1. Опис стандартної бібліотеки

Стандартна бібліотека Сі++ включає стандартну бібліотеку Сі з невеликими змінами, які роблять її більш відповідною для мови Сі++. Інша велика частина бібліотеки Сі++ заснована на Стандартній Бібліотеці Шаблонів (STL). Вона надає такі важливі інструменти, як контейнери (наприклад, вектори і списки) і ітератори (узагальнені вказівники), що надають доступ до цих контейнерів як до масивів. Крім того, STL дозволяє схожим чином працювати і з іншими типами контейнерів, наприклад, асоціативними списками, стеками, чергами.

Використовуючи шаблони, можна писати узагальнені алгоритми, здатні працювати з будь-якими контейнерами або послідовностями, доступ до членів яких забезпечують ітератори.

Так само, як і в Сі, можливості бібліотек активізуються використанням директиви #include для включення стандартних файлів. Всього в стандарті Сі++ визначено 50 таких файлів.

STL до включення в стандарт Сі++ була сторонньою розробкою, на початку — фірми HP, а потім SGI. Стандарт мови не називає її «STL», оскільки ця бібліотека стала невід'ємною частиною мови, проте багато людей до цих пір використовують цю назву, щоб відрізняти її від решти частини стандартної бібліотеки (потоки введення/виводу (Iostream), підрозділ Сі тощо). Проект під назвою STLport, заснований на SGI STL, здійснює постійне оновлення STL, IОstream і рядкових класів. Деякі інші проекти також займаються розробкою приватних застосувань стандартної бібліотеки для різних конструкторських завдань. Кожен виробник компіляторів Сі++ обов'язково поставляє якусь реалізацію цієї бібліотеки, оскільки вона є дуже важливою частиною стандарту і широко використовується.

* + 1. Переваги мови

Переваги мови C++:

* Продуктивність. Швидкість роботи програм на С++ практично не поступається програмам на С, хоча програмісти отримали в свої руки нові можливості і нові засоби.
* Масштабованість. На мові C++ розробляють програми для самих різних платформ і систем.
* Можливість роботи на низькому рівні з пам'яттю, адресами, портами. (Що, при необережному використанні, може легко перетворитися на недолік.)
* Можливість створення узагальнених алгоритмів для різних типів даних, їх спеціалізація, і обчислення на етапі компіляції, з використанням шаблонів.
  + 1. Недоліки мови

Недоліки мови С++:

* Наявність безлічі можливостей, що порушують принципи типобезпеки приводить до того, що в С++-програми може легко закрастися важковловима помилка. Замість контролю з боку компілятора розробники вимушені дотримуватися вельми нетривіальних правил кодування. По суті ці правила обмежують С++ рамками якогось безпечнішої підмови. Більшість проблем типобезпеки С++ успадкована від С, але важливу роль в цьому питанні грає і відмова автора мови від ідеї використовувати автоматичне управління пам'яттю (наприклад, збірку сміття). Так візитною карткою С++ стали вразливості типу "переповнювання буфера".
* Погана підтримка модульності. Підключення інтерфейсу зовнішнього модуля через препроцесорну вставку заголовного файлу (#include) серйозно уповільнює компіляцію, при підключенні великої кількості модулів. Для усунення цього недоліку, багато компіляторів реалізують механізм прекомпіляциі заголовних файлів (англ. Precompiled Headers).
* Недостача інформації про типи даних під час компіляції (CTTI).
* Мова C++ є складною для вивчення і для компіляції.
* Деякі перетворення типів неінтуїтивні. Зокрема, операція над беззнаковим і знаковим числами видає беззнаковий результат.
* Препроцесор С++ (успадкований від C) дуже примітивний. Це приводить з одного боку до того, що з його допомогою не можна (або важко) здійснювати деякі завдання метапрограмування, а з іншою, в наслідку своєї примітивності, він часто приводить до помилок і вимагає багато дій з обходу потенційних проблем. Деякі мови програмування (наприклад, Scheme і Nemerle) мають набагато могутніші і безпечніші системи метапрограмування (також звані макросами, але макроси С/С++ вони мало нагадують).
* З кінця 20-го століття в співтоваристві С++ набуло поширення так зване метапрограмування на базі шаблонів. По суті, воно використовує особливості шаблонів C++ в цілях реалізації на їх базі інтерпретатора примітивної функціональної мови програмування, що виконується під час компіляції. Сама по собі дана можливість вельми приваблива, але, в наслідку вище згаданого, такий код вельми важко сприймати і відладжувати. Мови Lisp/Scheme, Nemerle і деякі інші мають могутніші і водночас простіші для сприйняття підсистеми метапрограмування. Крім того, в мові D реалізована порівнянна по потужності, але значно простіша в застосуванні підсистема шаблонного метапрограмування.
* Хоча декларується, що С++ мультипарадигмена мова, реально в мові відсутня підтримка функціонального програмування. Частково, даний пропуск усувається різними бібліотеками (Loki, Boost) що використовують засоби метапрограмування для розширення мови функціональними конструкціями (наприклад, підтримкою лямбд/анонімних методів), але якість подібних рішень значно поступається якості вбудованих у функціональні мови рішень. Такі можливості функціональних мов, як зіставлення зі зразком взагалі украй складно емулювати засобами метапрограмування.
  + 1. Висновки щодо обраних засобів реалізаціїї

С++ - це мова програмування загального призначення, що включає економiю представлення, сучасне керування потоком i структурою даних i багатий набір операторів. C не є мовою "дуже високого рiвня" i не призначена для певної області застосування. Але відсутність в ній обмежень та її загальність роблять її зручнішою i ефективнішою для багатьох завдань, у порівнянні з мовами, що вважаються потужнішими.

### Опис методів реалізації

### 3.2.1 Загальний алгоритм функціонування штучної імунної системи

Основою роботи програмного засобу є штучний імунна система яка використовує алгоритм клональної селекції який, в свою чергу, проводить відбір нейромережевих детекторів на основі показника їх афітності. Також в її задачі входить:

* генерація імунних детекторі;
* навчання імунних детекторів;
* видалення найменш афітних імунних детекторів.

Розглянемо процес формування імунодетекторів на основі нейронних мереж. Спочатку формується початкова популяція імунних детекторів, кожен з яких є штучною нейронною мережею. Потім формується набір чистих файлів, який, як правило, складається з утиліт операційної системи, різних документів та різних програмних файлів. На наступному кроці вибирається кілька комп'ютерних вірусів або підписів. На плоскому файлі віруси утворюють навчальний набір для мережевих детекторів. У процесі навчання нейронна мережа вчиться розпізнавати шкідливі програми з "чистого" програмного забезпечення.

Набір тренованих нейронних мереж формує сукупність імунодетекторів, які циркулюють у комп'ютерній системі та виявляють комп'ютерні віруси. Наявність різноманітних навчальних файлів та файлів елементів для генерації вхідних векторів дає можливість отримати велику кількість різних імунних структур у їх структурі.

У процесі сканування невідомого файлу нейрон ідентифікує невідомий файл, в результаті якого імунодетектор вирішує, належить файл до класу шкідливих програм або до класу чистих файлів.

Загальний алгоритм функціонування штучної імунної системи, можна представити у такій послідовності:

а) Генерування початкової сукупності імунних детекторів, кожен з яких являє собою штучну нейронну мережу зі випадковими синаптичними зв’язками:

|  |  |
| --- | --- |
|  | (3.1) |

Де – i-й нейромережевий імунний детектор, r – загальна кількість детекторів.

б) Навчання детекторів сформованих імунних нейронних мереж. Навчальний зразок формується випадково з колекції чистих файлів (як правило, це різноманітні системні утиліти операційної системи) і з колекції шкідливих програм або сигнатур. Опорні вихідні значення нейронної мережі генеруються відповідно:

|  |  |
| --- | --- |
|  | (3.2) |

Де L – розмір навчальної вибірки.

Відповідно, множина еталонних зразків виглядає наступним чином:

(3.3)

Еталонні вхідні значення для і-го детектору формуються так:

(3.4)

в) Відбір (селекція) нейромережевих імунних детекторів на тестовому зразку. В ході цього етапу детектори, які виявилися нездатними до навчання, і детектори, в яких є різні недоліки (наприклад, помилкові спрацьовування) знищуються. Для цього кожен детектор перевіряється в тестовому відборі. В результаті визначається значення середньої квадратичної похибки для кожного детектора.

Сумарна квадратична помилка і-го детектора визначається наступним чином:

|  |  |
| --- | --- |
|  | (3.5) |

Де – значення j–го виходу i–детектора при подачі на вхід k–го образу.

Значення загальної квадратичної помилки характеризує здатність детектора виявляти шкідливі файли, що і визначає міру афітності. Чим менше значення помилки, тим більша його афітність(придатність детектора). Тому величину загальної квадратичної помилки можна використовувати для вибору найкращих детекторів.

У процесі сканування невідомого файла нейронна мережа ідентифікує невідомий образ, в результаті чого імунний детектор приймає рішення про приналежність файла до класу шкідливих програм або до класу чистих файлів.

Селекція детектора відбувається наступним чином:

(3.5)

г) Кожен детектор наділяється часом життя і випадковим чином вибирає файл для сканування із сукупності ще не перевірених ним файлів.

д) Сканування кожним детектором вибраного файла, в результаті чого визначаються вихідні значення детекторів

e) Якщо і-й детектор не виявив вірус в сканованому файлі, тобто = 0, тоді він вибирає наступний файл для сканування. Якщо термін експлуатації i-детектора закінчився, він знищується, а замість нього генерується новий детектор.

є) Власне кажучи, виявляється i-ідентифікатор, що файл сканується, так що Zi1 = 0 Zi2 = 1, вважається, що виявлений зловмисний файл і здійснена робота виявленого детектора. Операція мутацій включається в додаткове навчання детекторів клонів на виявленому шкідливому файлі. Це створює набір детекторів, налаштованих на виявлену шкідливу програму.

ж) Відбір клонованих детекторів, у яких показник афітності є найбільшим, тобто з найменшою середньою квадратичною помилкою. Якщо , то детектор пройшов відбір. Тут - сумарна квадратична помилка j-го клона i-го детектора, яка вираховується на шкідливому файлі.

з) Детектори-клони проводять сканування файлового простору комп’ютерної системи до тих пір, поки не відбудеться знищення всіх проявів шкідливого програмного забезпечення.

и) Формування детекторів імунної пам’яті. На цьому етапі роботи виявляються нейромережеві імунні детектори, які показали найкращі результати при виявленні присутнього в комп’ютерній системі вірусу. Детектори імунної пам’яті знаходяться в системі достатньо довго і забезпечують захист від повторного зараження.

Особливістю запропонованого алгоритму є те, що кожен мережевий детектор імунітету є повністю незалежним об'єктом (автономним агентом), тобто вибирає саму область сканування. Для цього він отримує список файлів, що зберігаються в просторі пам'яті, і випадковим чином вибирає список файлів для його перевірки. Після перевірки одного файлу детектор переходить до наступного файлу із наявного списку. Сканування файлів у мережі за допомогою імунного детектора продовжує працювати, поки детектор не виявить шкідливу програму або час, що минув для функціонування цього детектора. Широка кількість імунних детекторів нейронної мережі забезпечує своєчасне виявлення шкідливих програм. Таким чином, дотримується принцип децентралізації системи безпеки, побудованої на основі поєднання методів електронних мереж та штучної імунної системи, що значно підвищує відмовостійкість та безпеку системи в цілому.

### 3.2.2 Структура та алгоритм навчання нейромережевого імунного детектора

Оскільки основним завданням нейромережевого імунного детектора є розподіл простору вхідних зразків на два класи: клас чистих файлів та клас шкідливих файлів. Важливим етапом є вибір типу нейронної мережі детектору. В процесі циркуляції в системі нейромережевих імунних детекторів відбувається їх неперервне еволюціонування шляхом знищення старих і формування нових детекторів.

Після генерації нових детекторів відбувається процес їх навчання, складність якого пропорційна розмірності навчального зразка. Тому для підвищення швидкодії нейромережевої штучної системи потрібно обрати такий тип нейронної мережі, щоб його обов’язковою характеристикою було мінімальний розмір навчальної вибірки.

Для прикладу розглянемо багатошаровий перцептрон, котрий складається з n нейронів розподільного шару і 2 нейронів вихідного шару. Загальна кількість параметрів налаштування (вагових коефіцієнтів і порогових значень) в такій мережі визначаються наступним чином:

|  |  |
| --- | --- |
|  | (3.6) |

Для доброї класифікації розмір розмір навчальної вибірки має визначатись в відповідності з наступним виразом:

|  |  |
| --- | --- |
|  | (3.7) |

Де – допустима точність класифікації.

Нехай n = 128, m = 10 і = 0.1. Тоді 13120.

Аналогічний результат можна отримати для мультирекурентних нейронних мереж.

Тепер для порівняння розглянемо аналогічну мережу зустрічного поширення помилки з ідентичною кількістю нейронних елементів в шарах. В прихованому шарі будемо використовувати нейронні елементи Кохонена. В цьому випадку немає жорстких вимог до розміру до розмірності навчальної вибірки. Достатньо лише щоб кількість навчальних зразків була наступним:

|  |  |
| --- | --- |
|  | (3.8) |

Тому доцільніше використовувати для основи нейромережевого імунного детектора нейронну мережу зустрічного поширення помилки.

На рисунку 3.1 зображена архітектура нейромережевого імунного детектора, яких складається з трьох шарів нейронних елементів і арбітра.



Рисунок 3.1 – Схема нейромережевого імунного детектора

На вхід такого детектора в режимі функціонування подаються фрагменти файла що перевіряється, які формуються в відповідності до методу ковзного вікна. Перший шар нейронних елементів є розподільним. Він розподіляє вхідні сигнали на нейронні елементи другого шару. Кількість елементів розподільного шару дорівнює розмірності ковзного вікна. Других шар зіставлений з елементів Кохонена, котрі використовують конкурентний принцип навчання та функціонування згідно з правилом «переможець забирає все». Третій шар складається з двох елементів Гроссберга, які використовую лінійну функцію активації. Арбітр виконує процедуру кінцевого рішення про приналежність файла що сканується до чистого або вірусного класів.

Розглянемо вибір кількості нейронів в шарі Кохонена. Нейронний шар Кохоне кластеризує вхідний простір образів, в результаті чого утворюються кластери різних образів, кожен з яких відповідає своєму нейронному елементу. Кількість нейронів шару Кохона дорівнює m. При чому:

(3.9)

Де р – кількість нейронів шару Кохонена, котрі відповідають класу чистих файлів. r – кількість нейронів шару Кохонена, активність яких характеризує клас шкідливих файлів.

При навчанні нейромережевих імунодетекторів використовується навчальна вибірка, зіставлена з 50% чистого класу та 50% шкідливого класу. Таким чином, співвідношення файлів у виділеному виділення дорівнює один до одного. Цей взаємозв'язок отриманий експериментально, показавши найкращі результати.

Алгоритм формування навчальної вибірки складається з таких етапів:

а) формується набір чистих та вірусних файлів;

б) k чисті та h шкідливі файли вибираються із створеного зразка випадково;

в) з кожного файлу фрагменти довжини n вибираються випадковим чином, і в результаті формується навчальний зразок розміру L = (k + h) · A;

г) вибрані фрагменти нормалізуються за формулою:

|  |  |
| --- | --- |
|  | (3.10) |

Де – нормований аналог другого елемента в вибраному фрагменті.

Для навчання нейронів шару Кохонена використовується контрольоване конкурентне навчання. У цьому випадку зміна вагових коефіцієнтів переможців змінюється лише тоді, коли відбувається правильна класифікація вхідного образу, тобто вхідне зображення відповідає заданій множині нейронів у шарі Кохонена. Оскільки в шарі Кохона використовуються р нейрони для чистих вхідних образів, а r нейрони для шкідливих вхідних образів, то поточна класифікація є правильною, якщо введення чистого фрагмента є переможцем шару р нейрона Кохонена. Аналогічно правильна класифікація відбувається, якщо один з r нейронів в шарі Кохонена є переможцем. В інших випадках відбувається неправильна класифікація.

Нехай P і J характеризують відповідно чистий та шкідливий файл. Тоді правило коректної класифікації можна представити наступною імплікацією:

(3.11)

Де T – означення коректної класифікації.

При коретній класифікації вагові коефіцієнти нейрона-переможця підсилюються згідно з наступною формулою:

(3.12)

А при некоректній класифікації послаблюються:

(3.13)

Де – крок навчання.

Алгоритм навчання нейронів шару Кохонена можна представити у вигляді наступних кроків:

1. Випадково ініціалізуємо вагових коефіцієнтів нейронів шару Кохонена.
2. Подаємо вхідний, нормований образ з навчальної вибірки на розподільний шар нейронної мережі і проводимо наступні обрахування:

- вираховуємо зважену суму між вхідним образом і ваговими векторами нейронних елементів шару Кохонена за наступною формулою:

|  |  |
| --- | --- |
|  | (3.14) |

1. визначається нейронний елемент переможець з номер k:

(3.15)

1. проводиться модифікація вагових коефіцієнтів нейрона-переможця в відповідності до (3.12), якщо при подачі на вхід мережі чистого фрагменту переможцем є один з p нейронів, або при подачі на вхід мережі шкідливого фрагменту переможцем є один з r нейронів шару Кохонена. В іншому випадку проводиться модифікація вагових коефіцієнтів нейрона-переможця згідно з (3.13).

Процес повторюється, починаючи з пункту 2 для всіх вхідних образів.

Навчання проводиться до бажаної степені погодження між вхідними та ваговими векторами, тобто до тих пір, поки значення сумарної квадратичної помилки не стане дорівнювати встановленому пороговому значенню.

Третій шар зіставлений з двох нейронних елементів Гроссберга, проводить відображення кластерів, сформованих шаром Кохонена, в два класа, які характеризують чисті та вірусні образи. В загальному випадку вихідне значення j – го нейрона третього шару визначається наступним чином:

|  |  |
| --- | --- |
|  | (3.16) |

Де – ваговий коефіцієнт між і-м нейроном в шарі Кохонена і j-им нейроном шару Гроссберга.

Якщо нейрон-переможець в шарі Кохонена має номер k, то вихідні значення j-го нейрона третього шару дорівнюють:

|  |  |
| --- | --- |
|  | (3.17) |

Арбітр приймає остаточне рішення чи є файл, що сканується шкідливим. Для цього він вираховує кількість чистих та шкідливих фрагментів сканованого файла в відповідності до наступних виразів:

|  |  |
| --- | --- |
|  | (3.18) |
|  | (3.19) |

Де L – множина образів файлу що сканується, – вихідні значення і-го нейрона шару Гроссберга при подачі на вхід мережі k-го образу.

Остаточне рішення про приналежність файла до класу шкідливих приймається в відповідності до наступного виразу:

|  |  |
| --- | --- |
|  | (3.20) |

Де – перший нейрон шару Гроссберга, – другий нейрон шару Гроссберга, відповідно.

Таким чином, простір вихідних значень арбітра можна представити як таблицю.

|  |  |  |
| --- | --- | --- |
| № | Z | Клас |
| 1 | 1 | Чистий |
| 2 | -1 | Шкідливий |
| 3 | 0 | Не визначено |

Таблиця 3.1 – Простір вихідних значень арбітра

Якщо вихідне значення арбітра має нульове значення, то файл що сканується відправляється на додаткову перевірку іншому нейромережевому імунному детектору.

### 3.2.3 Алгоритм функціонування нейромережевого імунного детектора

В процесі сканування файла на нейромережевий детектор послідовно подаються фрагменти файла за методом ковзного вікна.

Алгоритм функціонування нейромережевого імунного детектора в режимі сканування файла можна звести до наступної послідовності кроків:

1. Встановлюються наступні початкові значення:

|  |  |
| --- | --- |
|  | (3.21) |

1. За методом ковзного вікна послідовно подаються вхідні образи (k=1, L) з файлу що сканується на нейронну мережу і для кожного вхідного образу проводяться наступні дії:

* визначається зважена сума між вхідним образом та ваговими векторами нейронів шару Кохонена (3.14);
* визначається нейрон-переможець з номером k (3.15);
* визначаються вихідні значення нейронів шару Гроссберга (3.16);
* визначається кількість чистих та шкідливих фрагментів файлу (3.18), (3.19);

1. визначається принадлежність файлу до чистого або шкідливого класів (3.20);
2. якщо Z = 0, назначається інший детектор на повторну перевірку цього ж файлу.
   1. Архітектура програмного засобу

Model-View-Controller - схема розділення поданих даних, користувальницький інтерфейс та керована логіка на три окремих компонента: модель, представлення та контролер - таким чином, що модифікація кожного компонента може здійснювати незалежність.



Рисунок 3.2 – Шаблон проектування Model View Controller

* Модель - надає дані і реагує на команду контролера, змінивши своє стан.
* Представлення - відповідь за показаними даними моделями користувач, реалізуючи моделі змін.
* Контролер - інтерпретує діючого користувача, підтримуючи модель про необхідність зміни.

Основні цілі застосування цих концепцій містяться у відокремленні моделі від його представлення. Завдяки такому розділенню підвищується можливість повторного використання коду. Найбільше корисно застосовувати дану концепцію у тих випадках, коли користувач повинен побачити найновіші дані одночасно у різних контекстах та з різними точками зору. Наприклад виконуючи наступні завдання:

* До однієї моделі можна приєднати кілька видів, при цьому не затримати реалізовану модель. Наприклад, деякі дані можуть бути одночасно представлені у вигляді електронних таблиць, гістограм та кругової діаграми;
* Не затримати реалізацію виду, можна змінити реакції на діючого користувача (натиснення кнопки на миші, введення даних, тощо) - для цього достатньо використати інший контролер;
* Ряд розробників спеціалізується лише в одній із областей: або розробляє графічний інтерфейс, або розлучає бізнес-логіку. Ви можете, можливо, отримати те, що програмісти, займаючись розробкою бізнес-логіки (моделі), не будуть висвітлюватись, і таке представлення буде використане.

В таблиці 3.2 наведено модулі програми що відповідають за функціональність таких структурних елементів як Model.

|  |  |  |
| --- | --- | --- |
| № | Назва модуля | Опис |
| 1 | File | Відповідає за роботу зі сховищем данних |
| 2 | Math | Відповідає за обрахунки в програмі |

Таблиця 3.2 – Модулі програми які належать до типу Model

В свою чергу кожен з модулів включає в себе класи які відповідають за його роботу. На рисунку 3.3 зображено діаграму класів модуля File. Модуль включає в себе 1 клас для роботи зі сховищем даних. Клас реалізує в собі повний набір функцій та процедур для отримання, інтерпретації та збереження даних зі сховищ.

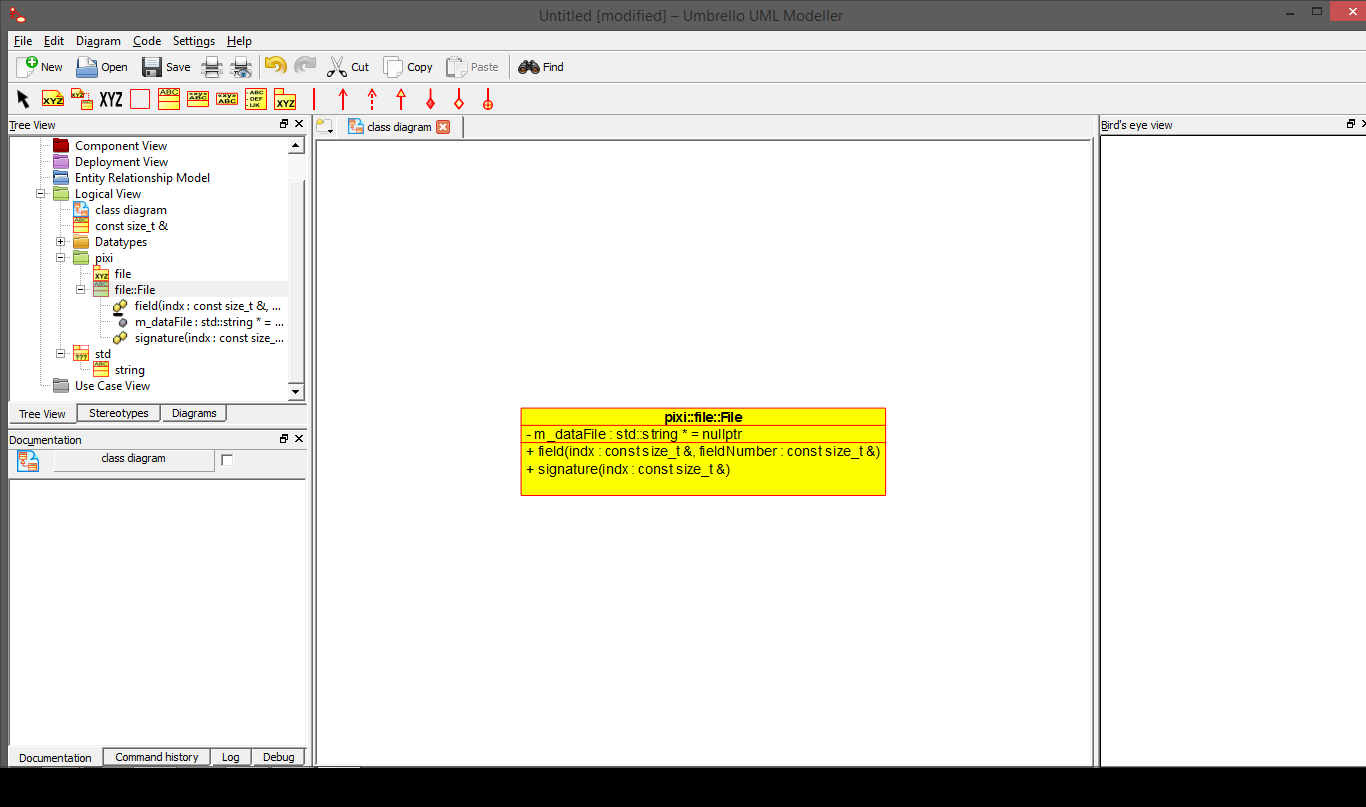


Рисунок 3.3 – Діаграма класів модуля File

На рисунку 3.4 зображено діаграму класів модуля Math. Модуль включає в себе 3 класи для роботи з математичними обчисленнями, діями над матрицями та повним набором операцій з нейронною мережею.

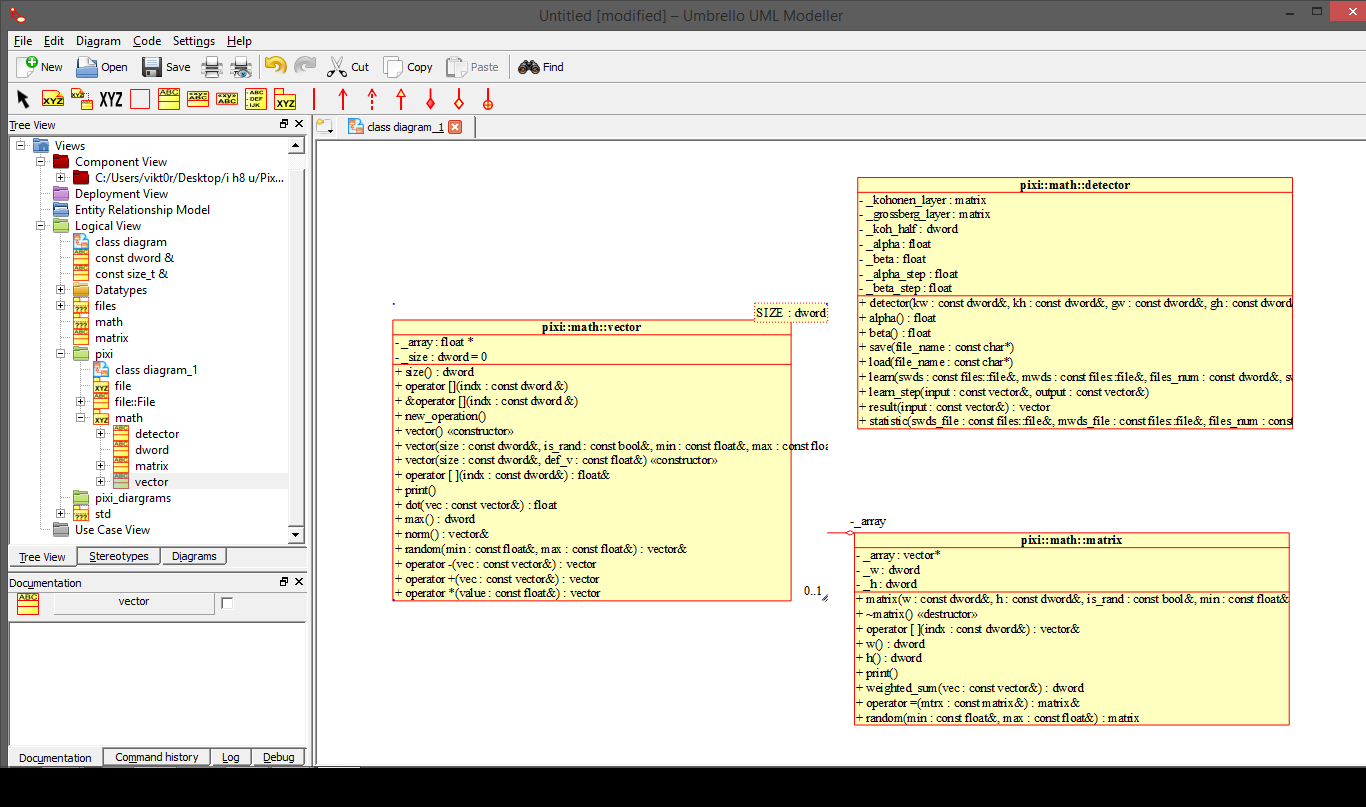


Рисунок 3.4 – Діаграма класів модуля Math

В таблиці 3.3 наведено модулі програми що відповідають за функціональність таких структурних елементів як View.

|  |  |  |
| --- | --- | --- |
| № | Назва модуля | Опис |
| 1 | UI | Відповідає за відображення вікна програми, елементів керування. |
| 2 | VP | Відповідає за графічне представлення поведінки прогами. |

Таблиця 3.3 – Модулі програми які належать до типу View

Графічне представлення в програмі реалізоване за рахунок використання терміналу операційної системи. В певній мірі це накладає деякі додаткові задачі для реалізації елементів керування та представлення. Головною причиною вибору термінального вікна операційної системи була максимальний баланс між швидкість відображення та виконням розрахунків.

На рисунку 3.5 зображена діаграма класів модуля UI. Модуль включає в себе 9 класів які забезпечують контроль користувачем роботи програми, відображення даних та дій, що їх виконує програма.

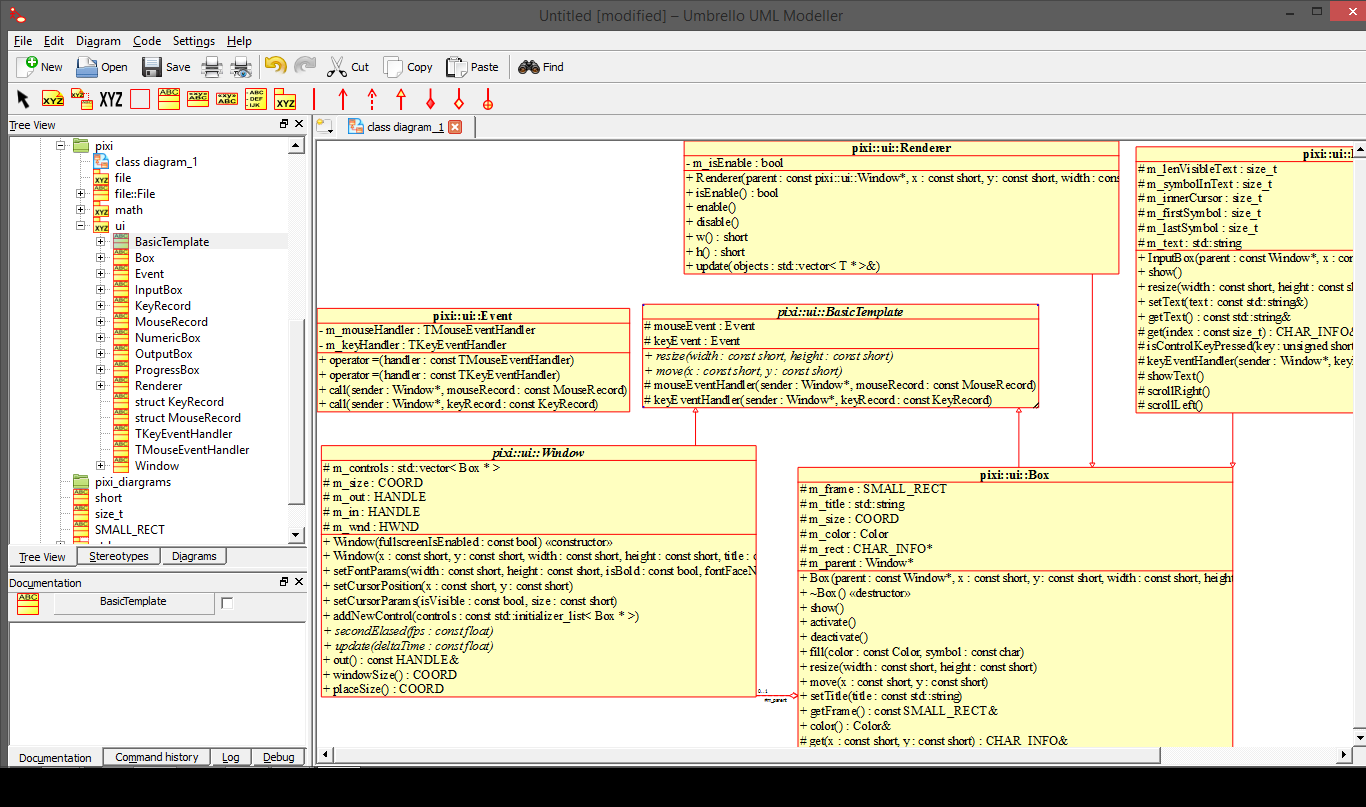


Рисунок 3.5 – Діаграма класів модуля UI

На рисунку 3.6 зображена діаграма класів модуля VP. Модуль включає в себе 4 класи які забезпечують зрозуміле предаставлення для користувача дій програми.

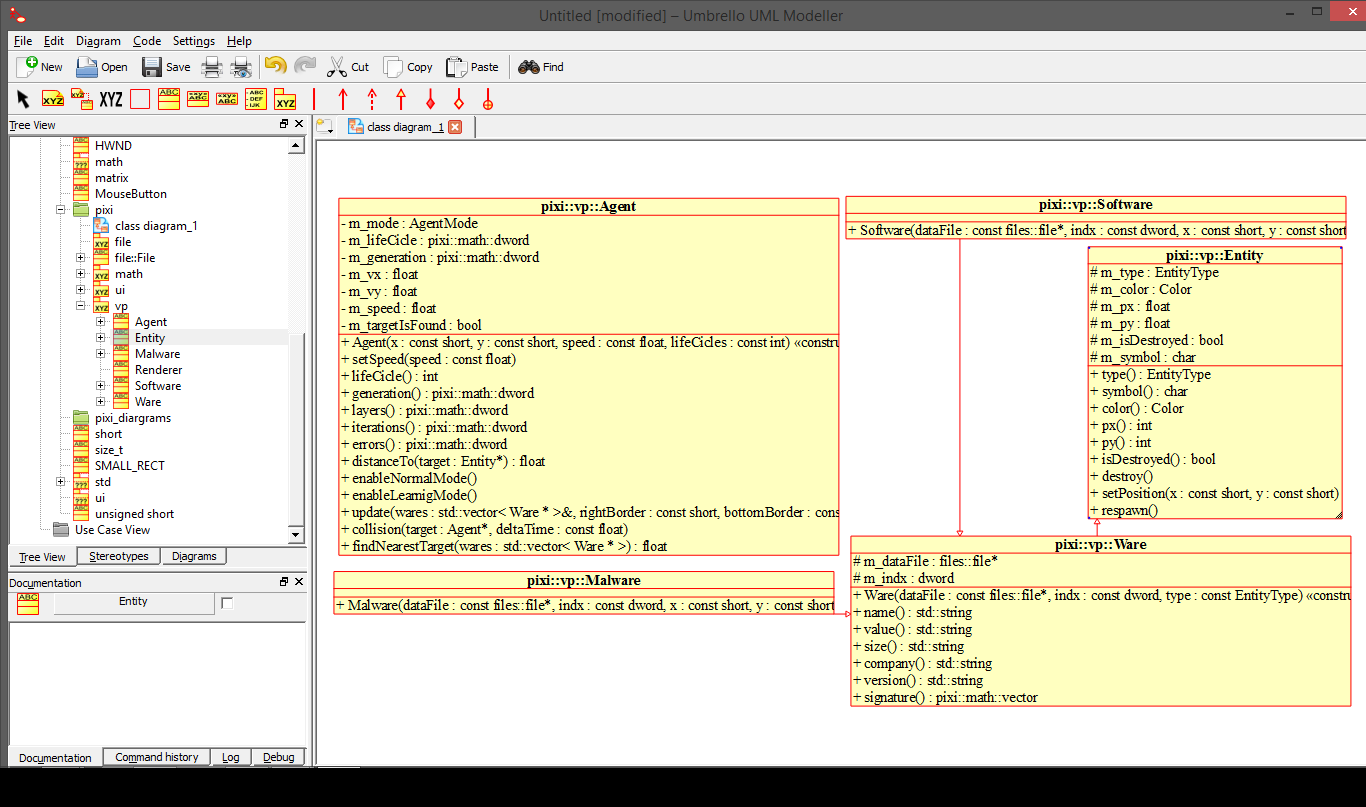


Рисунок 3.7 – Діаграма класів модуля VP