|  | **Софийски университет „Св. Кл. Охридски”**  Факултет по математика и информатика |  |
| --- | --- | --- |

**Курсов Проект**

на тема: „Pac-Man-AI”

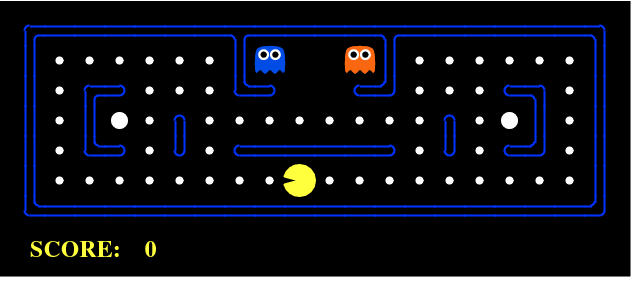
Студент: **Владимир Войков Стоянов** Ф.Н. **82284**

Курс: „КН курс 4 гр. 6“, Учебна година: 2023/24  
Преподаватели: **проф. Иван Койчев**:

=================================

Декларация за липса плагиатство:

* Плагиатство е да използваш, идеи, мнение или работа на друг, като претендираш, че са твои. Това е форма на преписване.
* Тази курсова работа е моя, като всички изречения, илюстрации и програми от други хора са изрично цитирани.
* Тази курсова работа или нейна версия не са представени в друг университет или друга учебна институция.
* Разбирам, че ако се установи плагиатство в работата ми ще получа оценка “Слаб”.

4.02.2024 г. Подпис на студента:  


Фигура 1. Играта [Pac-Man](#_5fl0nfoy754w)

**Съдържание**

[**1 Увод 2**](#_5aj94sxnb5r2)

[**2 Общ преглед на областта: Стратегии и алгоритми за оптимална игра на Pac-Man, базирана на ИИ 3**](#_pirm1tr85b93)

[Подходи и методи за решаване 3](#_njt768mtr65d)

[Възможни подходи към разработването на Pac-Man AI 4](#_mlqc6gw0v5fc)

[**3 Проектиране 4**](#)

[Анализ на изискванията 4](#_t09xzpzcm8i8)

[Обща архитектура 5](#_3a7xfml8kvh7)

[Модел на данни 5](#_l15h9oktd53j)

[Схема на представяне на знанията 5](#_jdly0bweu3wj)

[**Потребителски интерфейс 5**](#_hnhqo6b3rrtt)

[4 Реализация, тестване/експерименти 5](#_2et92p0)

[Реализация (Внедряване) 5](#_ypv9t6ea9l)

[Методологии за тестване 6](#_q6j2xzkv9t76)

[Експерименти и резултати 6](#_ov82cf74xz5l)

[Използвани технологии, платформи и библиотеки 7](#_xbpcdyjbshex)

[**5 Заключение 7**](#)

[**Бъдещо развитие 7**](#_1t3h5sf)

[**6 Използвана литература 7**](#)

# Увод

Проектът Pac-Man е разработени за [CS 188 в университета в Бъркли](#_5fl0nfoy754w). Целта е да се имплементират и приложат набор от AI техники за игра на Pac-Man. Този проект има за цел да вкара в употреба и сравни основни концепции за ИИ, като информирано търсене в пространството от състояния, машинно самообучение и др.

Идеалната цел на проекта е с помощта на ИИ агент да контролира героят Pac-man по оптимален начин, за да спечели играта с възможно най-висок резултат.

# Общ преглед на областта: Стратегии и алгоритми за оптимална игра на Pac-Man, базирана на ИИ

В сферата на изкуствения интелект и машинното самообучение (ML), разработването и внедряването на алгоритми за решаване на игри представлява завладяваща демонстрация за това как ИИ може да взаимодейства, анализира и навигира в сложни среди. Типичен пример за такова приложение е проектът за изкуствен интелект, базиран на класическата игра Pac-Man. Този проект показва разнообразие от алгоритми и стратегии, предназначени за навигация в лабиринти, оптимизиране на вземането на решения и подобряване на играта чрез интелигентно поведение на агента. Обхватът на проекта е в няколко ключови области, включително алгоритми за търсене, многоагентна координация и техники за машинно самообучение, всяка от които е пригодена за преодоляване на специфични предизвикателства в играта.

## Подходи и методи за решаване

* Алгоритми за търсене: алгоритми за неинформирано търсене като търсене в дълбочина (DFS), търсене в широчина (BFS), Uniform Cost Search (UCS) и за информирано - A\* са внедрени за насочване Pac-Man през лабиринти. Всеки алгоритъм предлага уникален подход към навигацията в лабиринта, показвайки своите силни страни и ограничения в различни конфигурации на лабиринти.
* Евристичен анализ: В контекста на A\* и други базирани на евристика търсения, разработването на ефективни евристики, като разстоянието Манхатън за обща навигация в лабиринта и по-специализирани евристики за събиране на храна и навигация в ъглите, са критични за оптимизиране на ефективността и ефективността на търсенето.
* Мултиагентни стратегии: Проектът се простира отвъд предизвикателствата на задачите с един агент, за да изследва динамиката на множество агенти. Разглеждайки стратегии като рефлексни агенти, които вземат [предвид близостта на храната, позицииите на призраците и други фактори,](#_vhh1ecgc4ow7) демонстрират основно реактивно поведение. По-сложни подходи, като например алгоритми Minimax, се използват за стратегическо вземане на решения в присъствието на противникови агенти и налична пълна информация за тях.
* Приложения за машинно обучение: Техники за машинно обучение, по-специално Q-learning и Approximate Q-Learning, се прилагат за разработване на стратегии за игри, които се адаптират и подобряват чрез опит. Тези методи се фокусират върху оптимизирането на производителността на Pac-Man в различни мрежови среди, демонстрирайки потенциала на ML в развиващите се стратегии за игри.

## Възможни подходи към разработването на Pac-Man AI

**1. Системи, базирани на правила**

* Евристични алгоритми за търсене
* Дървета на решенията

**2. Обучение с подсилване (RL)**

* Q-learning: RL алгоритъм без модел, при който AI научава политика (policy), за да направи най-доброто действие въз основа на текущото състояние на играта, като увеличи максимално очакваната стойност на общата награда за всички следващи стъпки.
* Дълбоки Q-мрежи (DQN): Комбинира Q-learning с дълбоки невронни мрежи за обработка на сложни пространства

**3. Еволюционни алгоритми**

* Генетични алгоритми: Симулиране на еволюцията чрез създаване на популация от Pac-Man стратегии, които се развиват с течение на времето, избирайки стратегии, които имат по-висок резултат в играта.
* Невроеволюция (Neuroevolution): Използване на еволюционни алгоритми, за развиване на теглата на невронните мрежи, които контролират агента Pac-Man, което потенциално води до ефективни стратегии.

**4. Търсене на дървета в Монте Карло (MCTS)**

* Алгоритъм за вземане на решения, идеален за играта на Pac-Man, където се симулират много прегравания на игра, за да вземат решения. MCTS балансира изследването на нови ходове с използването на ходове, за които знаем че са добри.

**5. Учене с учител**

* Имитационно обучение: Обучава се невронна мрежа върху набор от добри човешки игра, с цел да имитира човешки стратегии.

# **Проектиране**

Дизайнът на програмата, визуализацията и обектно ориентираната част са предварително разработени и използвани наготово от [сайта за проектът](https://inst.eecs.berkeley.edu/~cs188/sp22/projects/).

## Анализ на изискванията

Основното изискване за проектът е да се реализират и сравнят различни подходи към задачата за оптимална игра на Pac-Man. Необходимо изискване е агент сам да взема решение какъв ход да предприеме с цел да изиграе оптимално играта.

## Обща архитектура

Архитектурата на системата е модулна и се състои от няколко ключови компонента:

* AI Агенти: Този модул включва различни AI агенти (напр. SearchAgents, ReflexAgent), които прилагат различни алгоритми и стратегии.
* Алгоритми за търсене: Колекция от алгоритми за търсене (DFS, BFS, UCS, A\*), предназначени за навигация в лабиринтите на играта.
* Алгоритми за машинно самообучение: Включва алгоритми за машинно обучение, като Q-learning и Approximate Q-Learning, за адаптивни стратегии за игра.
* Дефиниции на проблеми: Дефинира специфични задачи (напр. FoodSearchProblem, CornersProblem), които агентите трябва да навигират и разрешат.
* Помощни програми: Осигурява поддръжка за механиката на играта, визуализацията и взаимодействието, включително състоянието на играта, графиките и компонентите на потребителския интерфейс.

## Модел на данни

Моделът на данните за проектите на Pac-Man е съсредоточен около представянето на състоянието на играта, включително оформлението на лабиринта, позицията на Pac-Man, позициите на призраците, местоположенията на храната и резултатите. Този модел улеснява взаимодействието между AI агентите и средата на играта, позволявайки вземане на информирани решения и движение въз основа на текущото състояние.

## Схема на представяне на знанията

Знанията в системата се представя чрез комбинация от представяния на състояния, алгоритми и евристични функции.

## Потребителски интерфейс

Потребителският интерфейс е графичен, предоставяйки визуално представяне на дъската за игра Pac-Man. Той позволява наблюдение в реално време на поведението и взаимодействията на AI агентите в игровата среда.

# Реализация, тестване/експерименти

## Реализация (Внедряване)

Фазата на реализация включва внедряването на различни AI алгоритми, както е посочено в демонстрационния скрипт. Това включва алгоритми за търсене (DFS, BFS, UCS, A\*), евристични дизайни за A\* (напр. евристика за разстоянието в Манхатън), стратегии за справяне със специфични проблеми, като намиране на храна или навигиране до ъглите, и техники за засилване на обучението (Q-learning и Приблизително Q-обучение). Всяко от тези изпълнения е предназначено да адресира определен аспект на играта, от основна навигация в лабиринта до сложно вземане на решения в присъствието на противници.

## Методологии за тестване

Тестването на всеки алгоритъм включва стартиране на играта Pac-Man със специфични параметри, които предизвикват AI агента да демонстрира своите способности. Процесът на тестване за всеки алгоритъм може да бъде разбит по следния начин:

* Алгоритми за търсене: Тествани в средни и големи лабиринти, за да се оцени ефективността им при намиране на път.
* Евристични функции: Ефективността на A\* с евристични функции се тества в сценарии, изискващи оптимално намиране на пътя и решаване на проблеми, като се използват параметри за указване на евристика
* Машинно обучение: Q-learning и Approximate Q-Learning се тестват в множество епизоди на различни размери на мрежата, като се фокусира върху способността на обучаващия се агент да подобри своята политика въз основа на опита.

Разбирането и реализацията на Q-learning алгоритмите се оповава на [статията](#_j4ur5yy5aapf).

## Експерименти и резултати

Експериментите, описани в сценария, демонстрират цялостна оценка на AI стратегии при различни условия. Основните констатации от тези експерименти включват:

* Ефективност при намиране на път: Сравнението между алгоритмите за търсене подчертава ефективността и ефективността на всеки метод, като A\* като цяло превъзхожда другите стратегии за търсене, когато е използвана подходяща евристика.
* Евристична ефективност: Прилагането на специфични за проблема евристики значително подобрява производителността на алгоритмите за търсене, намалявайки пространството за търсене и времето, необходимо за намиране на оптимални решения.
* Адаптивно обучение: Експериментите с машинно обучение илюстрират способността на агентите за обучение с подсилване да адаптират и оптимизират своите стратегии с течение на времето. Прогресията от първоначалните изпитания към по-късните епизоди демонстрира значително подобрение във вземането на решения и представянето на играта.

След проведените експерименти, най-добре се представя агентът, който взима решения на базата на политика генерирана след трениране на адаптивен Q-learning. Q-learning подхода също показва добри резултати, но след много по-голямо количество епизоди и съответно много повече време. Адаптивното Q-обучение превъзхожда традиционното Q-обучение в контекста на играта Pac-Man поради способността си да коригира стратегията си за обучение въз основа на динамиката и сложността на средата. Традиционното Q-обучение следва фиксирана скорост на учене и стратегия за изследване, която може да не е оптимална във всички етапи на учене или във всяка ситуация, срещана в играта. За разлика от това, адаптивното Q-обучение променя своите параметри, като скоростта на обучение (learning rate) и скоростта на изследване (exploration rate). Основното предимство е адаптивността към промените в околната среда. В среди като играта Pac-Man, където пространството на състояния могат да варират значително от една стъпка до следваща, способността на Adaptive Q-learning да коригира своите параметри му позволява да реагира по-ефективно към нови или променящи се състояния в сравнение с традиционното Q-обучение, което използва статичен подход през целия учебен процес.

## Използвани технологии, платформи и библиотеки

В проектът са използвани само и единствено библиотеки вградени в Python версия 3.6. Не са използвани външни библиотеки с имплементирани алгоритми.

**ВАЖНО** Проектът трябва да се изпълнява в среда с **Python 3.6**

# **Заключение**

Ефективност на търсенето: Сравняването на алгоритмите за търсене разкрива тяхната пригодност за различни сценарии на игра. Например, докато DFS и BFS предлагат основни възможности за търсене, тяхната производителност може да бъде надмината от UCS и A\* по отношение на оптималността на пътя и ефективността на търсене, особено в сложни лабиринти. Въпреки това подходът за машнно самобучение използващ адаптивен Q-learning надминава всички изброени опити от към производителност и резултати.

## Бъдещо развитие

Естественото продължение на опитът е да се развие алгоритъм за трениране на Deep Q-learning.   
Другите нереалзирани подходи, споменати във [Възможни подходи към разработването на Pac-Man AI](#_mlqc6gw0v5fc) също могат да бъдат развити и сравнени.

Една интересна задача за разработка би била оптимизирането на духовете (противниците) и тяхното поведение. Как това би променило моделите? Адаптивни ли са различните подходи?

# **Използвана литература**

### [1] [Projects - CS 188: Introduction to Artificial Intelligence, Spring 2022](https://inst.eecs.berkeley.edu/~cs188/sp22/projects/)

### [2][Reinforcement Learning Explained Visually (Part 4): Q Learning, step-by-step | by Ketan Doshi](https://towardsdatascience.com/reinforcement-learning-explained-visually-part-4-q-learning-step-by-step-b65efb731d3e)

### [3][PacMan: what kinds of heuristics are mainly used? - Stack Overflow](https://stackoverflow.com/questions/9994913/pacman-what-kinds-of-heuristics-are-mainly-used)