SVEUČILIŠTE U ZAGREBU
FAKULTET ORGANIZACIJE I INFORMATIKE
VARAŽDIN

Franco Kušek

IMPLEMENTACIJA IGRE S PREDIKTIVNIM AI PROTIVNIKOM:

PROJEKT

VIŠEAGENTNI SUSTAVI

Varaždin, 2024.

SVEUČILIŠTE U ZAGREBU

FAKULTET ORGANIZACIJE I INFORMATIKE

V A R A Ž D I N

Franco Kušek

Matični broj: 35918/07–R

Studij: Informacijski i poslovni sustavi

IMPLEMENTACIJA IGRE S PREDIKTIVNIM AI PROTIVNIKOM:

PROJEKT

Mentor:

Markus Schatten, Dijana Oreški

Varaždin, siječanj 2024.

Franco Kušek

Izjava o izvornosti

Izjavljujem da je ovaj projekt izvorni rezultat mojeg rada te da se u izradi istoga nisam koristio drugim izvorima osim onima koji su u njemu navedeni. Za izradu rada su korištene etički prikladne i prihvatljive metode i tehnike rada.

Autor potvrdio prihvaćanjem odredbi u sustavu FOI Radovi

Sažetak

Ovaj projekt predstavlja razvoj prediktivnog umjetne inteligencije (AI) protivnika za igranje šaha koristeći Minimax algoritam. Cilj je bio stvoriti AI koji može pouzdano igrati šah protiv ljudskih igrača, predviđajući moguće poteze i odabirući optimalnu strategiju. Algoritam Minimax koristi se za ocjenjivanje poteza prema njihovom potencijalnom utjecaju na ishod igre, minimizirajući mogući gubitak za najgore moguće scenarije. Implementacija uključuje šahovsku ploču upravljajuću kroz python-chess biblioteku, što omogućava standardno upravljanje pravilima i potezima u šahu. AI protivnik je opremljen osnovnom evaluacijskom funkcijom koja dodjeljuje bodove figurama na ploči i određuje vrijednost pojedinih poteza. Projekt također omogućuje izmjenu poteza između AI-a i ljudskog igrača, pri čemu se koristi konzolni unos za ljudske poteze i automatski generirani potezi za AI. Program prepoznaje šah-mat i remi kao završne uvjete igre.

Ključne riječi: riječ; riječ; ...riječ; Obuhvaća 7 ± 2 ključna pojma koji su glavni predmet rasprave u radu.

Sadržaj

1. Uvod 1

2. Razrada teme 2

3. Kritički osvrt 4

4. Opis implementacije aplikacije 5

4.1. Prikaz rada aplikacije 7

4.2. Kritički osvrt 7

5. Zaključak 10

5.1. Literatura 10

Popis literature 11

Popis slika 12

Popis isječaka koda 13

1. Uvod

Uvod

U današnjem tehnološki naprednom svijetu, umjetna inteligencija (AI) nalazi svoju primjenu u različitim aspektima naših života, od automatizacije složenih zadataka do poboljšanja korisničkog iskustva kroz personalizaciju. Jedno od područja koje konstantno fascinira istraživače i entuzijaste jest razvoj AI-a koji može izazvati ljudske igrače u igrama koje zahtijevaju strateško razmišljanje. Šah, kao igra koja je stoljećima simbol intelektualne snage i strateške dubine, postaje idealno polje za testiranje granica računalnog razmišljanja i odlučivanja. Implementacija AI protivnika u šahu koristeći Minimax algoritam predstavlja ne samo tehnički izazov već i priliku za istraživanje složenosti ljudskog kognitivnog procesa.

Motivacija

Motivacija za odabir ove teme leži u želji za razumijevanjem i razvojem AI sistema koji mogu simulirati ljudsku sposobnost predviđanja i planiranja. Šah je odabran zbog svoje neprocjenjive vrijednosti u igrama koje se koriste kao mjerilo za sposobnosti AI-a. Kroz povijest, šahovski AI je služio kao benchmark za mjerenje napretka u računalnoj znanosti, od Deep Bluea do modernih AI sistema poput AlphaZero.

Razvijanjem šahovskog AI protivnika koji koristi Minimax algoritam, projekt ima za cilj stvoriti temelje za razumijevanje složenijih AI sistema koji se mogu prilagoditi i izvoditi u raznim scenarijima. Pored toga, implementacija alpha-beta rezanja pruža praktičnu primjenu teoretskih koncepta u optimizaciji algoritama, čineći AI protivnika bržim i efikasnijim.

Krajnji cilj ovog projekta je demonstrirati kako se osnovni principi AI-a mogu primijeniti u stvaranju efikasnih i inteligentnih sistema koji ne samo da izazivaju ljudske igrače, već i pružaju uvid u napredak mašinskog učenja i umjetne inteligencije.

2. Razrada teme

Glavni Dio Teme

Centralni aspekt ovog projekta je implementacija prediktivnog AI protivnika u šahu koristeći Minimax algoritam. Da bismo razumjeli kako AI radi, potrebno je prvo definirati ključne koncepte i teorijsku osnovu koja stoji iza algoritama odlučivanja koji se koriste u igrama.

Definicije

Šah: Šah je dvopartitivna igra s potpunim informacijama, što znači da su sve informacije o trenutnom stanju igre uvijek dostupne oba igrača. Cilj igre je dovesti protivnika u poziciju 'šahmata', gdje je njegov kralj napadnut i ne može se pomaknuti na sigurno polje.

AI Protivnik: AI protivnik u kontekstu šaha je računalni program koji koristi algoritme odlučivanja za izvođenje poteza umjesto ljudskog igrača.

Minimax Algoritam: Algoritam koji se koristi u igrama poput šaha za pronalaženje optimalnog poteza minimizirajući najveći mogući gubitak (maksimalnu štetu) u svim budućim scenarijima igre. Algoritam pretražuje stablo igre sve do definirane dubine i koristi evaluacijsku funkciju za određivanje vrijednosti svakog čvora (odnosno stanja igre).

Alpha-Beta Rezanje: Optimizacijska tehnika primjenjiva na Minimax algoritam koja "rezanje" grana stabla pretrage koje neće rezultirati poboljšanjem trenutno najboljeg poteza pronađenog za bilo kojeg igrača.

Teorija na Kojoj se Temelji Formalizam

Minimax algoritam temelji se na pretpostavci da oba ig

rača igraju optimalno. Algoritam analizira sve moguće poteze do određene dubine i vraća potez koji vodi do najboljeg mogućeg rezultata unutar najgoreg scenarija. Teorijski, algoritam analizira poteze sve do kraja igre, ali zbog računalnih ograničenja, u praksi se pretraživanje ograničava na određenu dubinu.

Teorija Igre: Minimax algoritam izvire iz teorije igre, grane primijenjene matematike koja proučava odlučivanje u situacijama s više sudionika (igrača) gdje rezultat za svakog igrača ovisi o izborima drugih igrača. U šahu, gdje su dva igrača, svaki potez ima odgovarajući odgovor, i Minimax se koristi za pronalaženje poteza koji će maksimizirati položaj igrača uzimajući u obzir najbolje moguće odgovore protivnika.

Algoritamska Dubina: U kontekstu Minimax algoritma, dubina se odnosi na broj poteza unaprijed koji algoritam razmatra prije nego što izvrši evaluacijsku funkciju. Dublje pretraživanje rezultira boljom prediktivnom sposobnošću, ali eksponencijalno povećava potrebne računalne resurse.

Evaluacijska Funkcija: Kritični dio Minimax algoritma je evaluacijska funkcija koja ocjenjuje koliko je položaj na ploči povoljan za određenog igrača. U šahu, ova funkcija može uzeti u obzir različite faktore poput materijalne vrijednosti figura, sigurnosti kralja, kontrolu centra ploče, mobilnost figura, strukturu pijuna, i druge strateške koncepte.

Alpha-Beta Rezanje: Teorija ovog pristupa temelji se na principu da nije potrebno evaluirati potez koji sigurno neće biti odabran. Kada algoritam pronađe potez koji je bolji od trenutno najboljeg (za maximizirajućeg igrača) ili lošiji od trenutno najgoreg (za minimizirajućeg igrača), daljnje grane se ne moraju razmatrati.

U praksi, kombinacija Minimax algoritma i alpha-beta rezanja omogućava stvaranje AI-a koji ne samo da efikasno igra šah, već pruža izazovno iskustvo za ljudske igrače, pružajući uvide u razvoj AI tehnologija i njihovu primjenu u različitim područjima.

3. Kritički osvrt

Projekt implementacije prediktivnog AI protivnika u šahu koristeći Minimax algoritam s alpha-beta rezanjem ima brojne aspekte koji zaslužuju kritički osvrt, posebno s obzirom na praktičnu izvedivost i primjenu.

Praktična Izvedivost

Složenost Algoritma: Iako je Minimax algoritam s alpha-beta rezanjem efikasan u smanjenju broja čvorova koje je potrebno istražiti, složenost igre poput šaha i dalje predstavlja izazov. Kako se dubina pretrage povećava, eksponencijalno raste i broj potencijalnih poteza, što može rezultirati značajnim vremenskim i procesorskim zahtjevima.

Ograničenja Evaluacijske Funkcije: Evaluacijska funkcija, koja je ključna za ocjenu ploče, često se oslanja na heurističke metode koje možda ne uzimaju u obzir sve aspekte igre. U slučaju šaha, ova funkcija može ocijeniti materijalnu vrijednost figura, ali možda neće učinkovito ocijeniti strategijske pozicije, kontrolu centra ili potencijalne prijetnje.

Vremenska Ograničenja: U stvarnim šahovskim partijama, posebno u brzopoteznim formatima, vremensko ograničenje može biti ključni faktor. AI koji zahtijeva duže vremensko razdoblje za izračunavanje poteza možda neće biti praktičan u takvim okolnostima.

Primjena

Edukativni Alat: AI temeljen na Minimax algoritmu s alpha-beta rezanjem može poslužiti kao odličan edukativni alat za učenje šaha, pružajući korisnicima mogućnost da igraju protiv izazovnog protivnika i uče iz njegovih poteza.

Istraživanje u AI: Projekt doprinosi razumijevanju osnova AI-a i nudi platformu za daljnja istraživanja, posebno u području poboljšanja evaluacijskih funkcija i učinkovitosti algoritama odlučivanja.

Ograničena Prilagodljivost: Dok je AI efikasan u igri šaha, njegova primjena je ograničena na ovu specifičnu igru. Prilagodba za druge igre ili scenarije zahtijeva značajne modifikacije

i razumijevanje specifičnosti tih igara, što može biti vremenski zahtjevno i tehnički složeno.

Budući Razvoj i Poboljšanja: Postoji potencijal za nadogradnju i integraciju naprednijih tehnika umjetne inteligencije, kao što su strojno učenje i neuronske mreže. Takav pristup može omogućiti AI-u da se prilagodi različitim stilovima igre i razvija nove strategije, čime se povećava njegova primjenjivost i izazovnost.

Interaktivnost i Korisničko Iskustvo: Iako je AI kompetentan protivnik, korisničko iskustvo može biti ograničeno zbog nedostatka interaktivnosti i vizualnih elemenata koji su česti u modernim igrama. Unapređenje korisničkog sučelja i vizualizacija igre moglo bi dodatno povećati privlačnost i pristupačnost projekta.

4. Opis implementacije aplikacije

Projekt implementacije šahovskog AI-a koristeći Minimax algoritam s alpha-beta rezanjem obuhvaća nekoliko ključnih komponenti i procesa. Evo detaljnog opisa implementacije:

1. Izbor Programskog Jezika i Okruženja Python: Odabran zbog svoje jednostavnosti, čitljivosti i bogate ekosustava biblioteka. Biblioteka python-chess: Koristi se za upravljanje šahovskim pravilima, potezima i stanjem ploče. 2. Implementacija Šahovske Ploče

```
import chess
```

Isječak koda 1: Import chess

Inicijalizacija Ploče: Pomoću python-chess, inicijalizira se standardna šahovska ploča s početnim rasporedom figura. 3. Razvoj Algoritma

```
board = chess.Board()
```

Isječak koda 2: Import chess

Minimax Algoritam: Srce AI-a, koristi rekurzivnu funkciju za evaluaciju svih mogućih poteza do određene dubine. Alpha-Beta Rezanje: Integrirano u Minimax za smanjenje broja nepotrebnih evaluacija, poboljšavajući performanse algoritma. 4. Evaluacijska Funkcija

```
def minimax(board, depth, alpha, beta, maximizing_player):
    if depth == 0 or board.is_game_over():
        return evaluate_board(board)

    if maximizing_player:
        max_eval = -float('inf')
        for move in board.legal_moves:
            board.push(move)
            eval = minimax(board, depth - 1, alpha, beta, False)
            board.pop()
            max_eval = max(max_eval, eval)
            alpha = max(alpha, eval)
            if beta <= alpha:
                break
        return max_eval
    else:
        min_eval = float('inf')
        for move in board.legal_moves:
            board.push(move)
            eval = minimax(board, depth - 1, alpha, beta, True)
            board.pop()
            min_eval = min(min_eval, eval)
            beta = min(beta, eval)
            if beta <= alpha:
                break
        return min_eval
```

Isječak koda 3: Import chess

Ocjena Stanja Ploče: Funkcija daje numeričku vrijednost trenutnom stanju ploče, uzi-

majući u obzir materijalnu vrijednost figura i druge strategijske faktore. Heuristike: Jednostavna evaluacijska logika temeljena na materijalnoj vrijednosti figura.

```
def evaluate_board(board):
    if board.is_checkmate():
        if board.turn:
            return -9999  # Black wins, negative value for white
        else:
            return 9999   # White wins, positive value for black
    if board.is_stalemate():
        return 0  # Draw

    # Simple material count for evaluation
    pawn = 1
    knight = 3
    bishop = 3
    rook = 5
    queen = 9
    score = 0
    for (piece, value) in [(chess.PAWN, pawn), (chess.KNIGHT, knight),
    ↪  (chess.BISHOP, bishop), (chess.ROOK, rook), (chess.QUEEN, queen)]:
        score += len(board.pieces(piece, chess.WHITE)) * value
        score -= len(board.pieces(piece, chess.BLACK)) * value
    return score
```

Isječak koda 4: Import chess

5. Izbor Poteza

Funkcija best_move: Odabire najbolji potez temeljen na rezultatima Minimax algoritma. Interakcija s Korisnikom: Korisnik unosi svoje poteze preko konzole, a AI odgovara svojim potezom.

```
def best_move(board, depth):
    best_move_found = None
    best_value = -float('inf')
    alpha = -float('inf')
    beta = float('inf')
    for move in board.legal_moves:
        board.push(move)
        board_value = minimax(board, depth - 1, alpha, beta, False)
        board.pop()
        if board_value > best_value:
            best_value = board_value
            best_move_found = move
            alpha = max(alpha, best_value)
    return best_move_found
```

Isječak koda 5: Import chess

6. Korisničko Sučelje i glavni dio programa

Konzolno Sučelje: Ploča se prikazuje u tekstualnom formatu s oznakama za redove i kolone. Unos Poteza: Korisnik unosi poteze u standardnoj šahovskoj notaciji.

7. Obrada grešaka Validacija Poteza: Provjera ispravnosti korisničkih poteza i obrada potencijalnih grešaka u unosu. Upravljanje Izuzecima: Osiguravanje da AI može adekvatno

```
board = chess.Board()
while not board.is_game_over():
    print(print_board_with_labels(board))  # Ispisuje ploču s oznakama
    if board.turn:  # Bijeli (AI)
        move = best_move(board, 3)  # Povećana dubina za bolji AI
        if move is None:
            print("Nema više legalnih poteza za AI.")
            break
        board.push(move)
    else:
        move = input("Vaš potez: ")
        try:
            board.push_san(move)
        except ValueError:
            print("Nevažeći potez, pokušajte ponovno.")
            continue
    print(print_board_with_labels(board))  # Ponovno ispisuje ploču nakon AI poteza

print("Kraj igre")
```

Isječak koda 6: Import chess

reagirati na neočekivane situacije, kao što su nevažeći potezi ili problemi s igrom.

```
try:
            board.push_san(move)
        except ValueError:
            print("Nevažeći potez, pokušajte ponovno.")
            continue
    print(print_board_with_labels(board))  # Ponovno ispisuje ploču nakon AI poteza
```

Isječak koda 7: Import chess

4.1. Prikaz rada aplikacije

Aplikaciju treba okrenuti preko komandne linije sah.py te ce se zatim otvorit aplikacija

Nakon sto se ekran pokrene prikaza ce nam se sahovska ploča sa imenima figura i sa strane pisu solova i brojevi ako bismo mogli lakse znati gjde odigrati. Imena figura su predstvaljene kao slova(p- pijuni, n- konj, b- lovac, r- kula, q- kraljicu i k- kralja). Igrač igra sa malim slovima dok AI igra sa velikim.

Igra se tako da napisete Veliko slovo figure koju zelite pomaknuti i onda mjesto gdje je zelite pomaknuti na primjer Ne6 sto ce staviti konja na e6 međutim ako zelite pomaknuti pijuna ne pisete nikakvo slovo nego sam mjesto gdje ga zelite pomaknuti.

4.2. Kritički osvrt

Kritički Osvrt na Temu Implementacije Šahovskog AI-a Koristeći Minimax Algoritam s Alpha-Beta Rezanjem

Projekt implementacije šahovskog AI-a koristeći Minimax algoritam s alpha-beta reza-

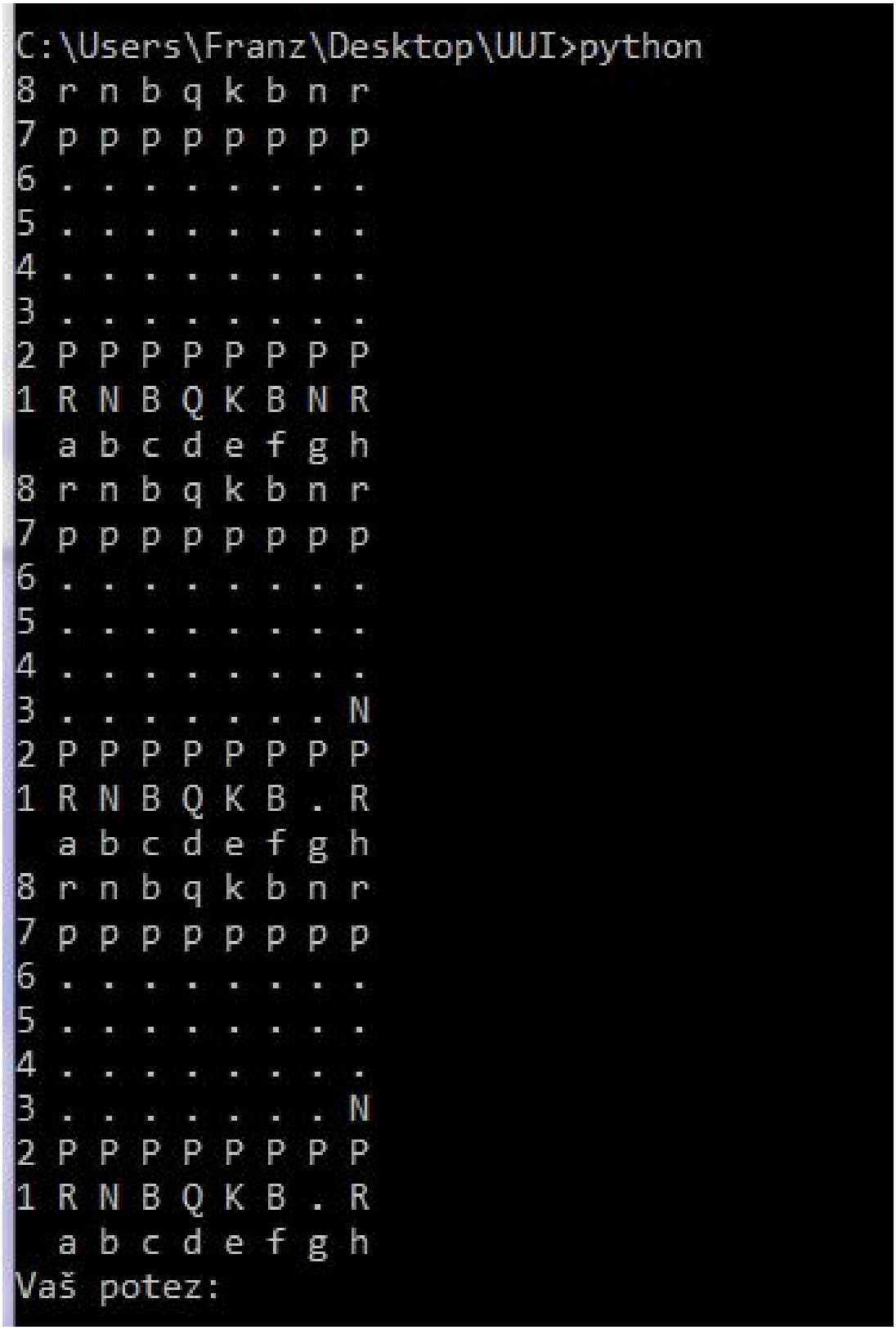
njem nudi zanimljiv uvid u složenost i izazove u razvoju umjetne inteligencije za igre. Iako ova implementacija služi kao koristan demonstrator AI tehnologije, postoje značajne točke koje zaslužuju kritičku analizu.

Složenost i Efikasnost Algoritma: Minimax algoritam s alpha-beta rezanjem je učinkovit u smanjenju broja pregledanih čvorova, što je ključno u šahu gdje broj potencijalnih poteza raste eksponencijalno. Međutim, čak i uz ovo poboljšanje, algoritam se može suočiti s izazovima pri visokim dubinama pretrage zbog ogromnog broja mogućih kombinacija poteza. To može dovesti do značajnih vremenskih kašnjenja pri izračunu poteza, što je posebno problematično u vremenski ograničenim partijama.

Ograničenje Evaluacijske Funkcije: Evaluacijska funkcija koja se koristi u ovom projektu temelji se uglavnom na materijalnoj vrijednosti figura, što je prilično osnovan pristup. U stvarnom šahu, pozicijska igra, kontrola prostora, struktura pijuna i sigurnost kralja su jednako važni. Nedostatak složenije evaluacijske logike može dovesti do suboptimalnih poteza AI-a.

Univerzalnost i Prilagodljivost: Dok je AI specifično dizajniran za šah, njegova arhitektura i algoritmi nisu lako prilagodljivi drugim igrama ili problemima. Ova specifičnost čini AI manje fleksibilnim i ograničava njegovu primjenu na širok spektar problema u području umjetne inteligencije.

Edukativna Vrijednost: Iako projekt nudi odličnu priliku za učenje o algoritmima i AI-u, moguće je da nedostaje dublje pedagoške strukture ili smjernica koje bi pomogle korisnicima u razumijevanju temeljnih principa AI-a i njihove primjene u stvarnom svijetu.



Slika 1: Prvi ispis programa

5. Zaključak

: Implementacija šahovskog AI-a koristeći Minimax algoritam s alpha-beta rezanjem je vrijedan poduhvat koji demonstrira ključne koncepte umjetne inteligencije. Međutim, postoje ograničenja u pogledu složenosti algoritma, dubine analize i univerzalnosti primjene. Budući rad bi se mogao fokusirati na integraciju naprednijih tehnika poput strojnog učenja, poboljšanja evaluacijske funkcije

za bolje razumijevanje šahovske strategije i pozicije, te proširenje projekta kako bi uključivao interaktivnije korisničko sučelje ili pedagoške elemente za edukaciju korisnika. Osim toga, istraživanje načina za poboljšanje prilagodljivosti i efikasnosti AI-a u različitim igračkim okruženjima moglo bi značajno proširiti njegovu primjenjivost i doprinijeti napretku u polju umjetne inteligencije.

5.1. Literatura

Bibliografija

[1] Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Pearson, 3rd edition, 2009.

[2] Gary Linscott. *Python Chess Programming*. Online: `https://python-chess.readthedocs.io/en/latest/`

[3] George T. Heineman, Gary Pollice, and Stanley Selkow. *Algorithms in a Nutshell*. O'Reilly Media, 2nd edition, 2015.

[4] Jeremy Silman. *The Complete Book of Chess Strategy: Grandmaster Techniques from A to Z*. Siles Press, 1998.

Popis slika

1. Prvi ispis programa 9

Popis isječaka koda

1. Import chess 5
2. Import chess 5
3. Import chess 5
4. Import chess 6
5. Import chess 6
6. Import chess 7
7. Import chess 7