UNIVERSITÀ DEGLI STUDI DI SALERNO

Dipartimento di Informatica

Corso di Laurea Triennale in Informatica

TESI DI LAUREA

TITOLO TESI

RELATORE

Prof. **Fabio Palomba**

Università degli studi di Salerno

CANDIDATO

Giuseppe Pagano

Matricola: 0512106337

Anno Accademico 2021-2022

INSERIRE QUI UNA DEDICA O UNA CITAZIONE

Sommario

INSERIRE ABSTRACT

Indice

Elenco delle figure

Elenco delle tabelle

CAPITOLO 1

Introduzione

1.1 Contesto applicativo

Gli scacchi sono un gioco di strategia deterministico a somma zero e ad informazione completa che si svolge su una tavola quadrata formata da 64 caselle ,di due colori alternati, detta scacchiera sulla quale ogni giocatore contraddistinto da uno di due colori nero o bianco, dispone di 16 pezzi:un re, regina, due alfieri, due cavalli, due torri e otto pedoni. obiettivo del gioco è dare scacco matto, ovvero minacciare la cattura del re avversario mentre esso non ha modo di rimuovere il re dalla sua posizione di pericolo alla sua prossima semimossa. Ulteriori informazioni verranno fornite nei successivi capitoli quando maggiori conoscenze di teoria si riveleranno necessarie per poter procedere allo sviluppo del motore.

1.2 Motivazioni e obiettivi

Gli scacchi,gioco nato in india attorno al 600 d.C,da gioco utilizzato nelle corti aristocratiche per rappresentare rapporti di potere a campo di battaglia tra uomo e macchina in uno dei primi e più famosi tentativi di far superare ad una macchina l'intelletto umano (Kasparov vs Deep Blue 1996-1997) ,gli scacchi , non hanno mai fallito nel saper cattivare l'attenzione del grande pubblico nonostante abbiano ormai più di 1000 anni sulle spalle.
Quello che agli occhi di un profano potrebbe sembrare un fenomeno stranissimo è in realtà di facile spiegazione se ci si concentra su una delle caratteristiche fondamentali del gioco degli

scacchi questa caratteristica è la **complessità**,in una partita di scacchi fin dalla prima semimossa sono possibili 20 scelte per la seconda semimossa il totale di possibili combinazioni sale a 400, dopo 5 semimosse avremo 119,060,324 possibili risposte, le possibili mosse di una partita si stimano attorno alle 2^{155}.
Con uno spazio di ricerca cosi grande non dovrebbe stupire sapere che è da quando esistono i computer che si cerca un modo di sfruttare la loro potenza di calcolo nel mondo degli scacchi.
La nascita degli scacchi computazionali si deve al lavoro di Claude Shannon, famoso per i suoi innumerevoli contributi al campo della teoria dell'informazione,egli, con il suo paper "Programming a Computer for Playing Chess" del 1950 ha gettato le basi per quello che oggi è il campo conosciuto come scacchi computazionali.
Questa tesi nasce dalla volontà di esplorare questo vasto e interessante campo dell'informatica,e dal voler creare un testo in grado di guidare chiunque lo legga nella creazione di un motore scacchistico spiegando tutte le fasi della progettazione ed illustrando le possibili scelte che condizionano l'efficienza di un motore ,dato che la letteratura su questo fronte è non particolarmente florida e soprattutto quasi esclusivamente in lingua inglese.

1.3 Risultati ottenuti

1.4 Struttura della tesi

CAPITOLO 2

Progettazione e implementazione

In questo capitolo viene mostrato passo passo un procedimento guida alla realizzazione di un motore scacchistico

2.1 Prefazione

Lo sviluppo di un motore scacchistico è fortemente influenzato dalle scelte progettuali, una di queste è il linguaggio di programmazione che si vuole utilizzare, le prestazioni di un motore possono essere fortemente influenzate dalla natura del linguaggio di programmazione, in particolare l'utilizzo di un linguaggio interpretato e non compilato può impattare notevolmente sulla velocità con la quale il nostro motore è in grado di elaborare le milioni di posizioni con le quali dovrà avere a che fare in una singola partita. Tutti gli esempi di codice all'interno di questa tesi saranno scritti nel linguaggio C, si consiglia quindi di avere almeno una minima familiarità con tale linguaggio.Si segnalano comunque diversi tool per il linguaggio python per chi volesse approcciarsi a questo mondo utilizzando un linguaggio più beginner friendly quali:

- **python-chess**: una libreria di python che contiene funzioni di libreria per la rappresentazione di scacchiera e pezzi e per la generazione e validazione delle mosse,utile se ci si vuole concentrare esclusivamente sulla parte di ricerca e di valutazione di un motore scacchistico.
- **Sunfish**: un motore scacchistico per principianti scritto nel linguaggio python che in sole 111 linee di codice illustra ,in maniera semplificata, l'implementazione della gran parte dei concetti chiave di un motore scacchistico.

2.2 Rappresentazione della scacchiera e dei pezzi

Il primo passo dello sviluppo di un motore scacchistico è decidere come si vuole rappresentare la scacchiera , si tratta di una scelta fondamentale,non solo perché in seguito ci permetterà di testare le funzioni che andremo a implementare, ma anche perché è nella scacchiera che ,generalmente, viene conservato lo stato generale della partita.[1] Inoltre il tipo di codifica può influenzare la rapidità e la facilità col quale possiamo accedere alle informazioni sullo stato corrente dei pezzi e come vedremo in seguito è in grado di influenzare funzioni come la generazione delle mosse. Non è raro per motori scacchistici particolarmente complessi l'utilizzo di più tipi di board in base al tipo di informazione da conservare e all'utilizzo che se ne vuole fare. Per la rappresentazione di una scacchiera sono chiaramente possibili moltissime scelte, di seguito verranno illustrate alcune tra le più popolari ed utilizzate.

[1] per stato di una partita si intendono informazioni come informazioni su chi ha diritto di muovere, i permessi di arrocco,lo stato della regola delle 50 mosse etc

2.2.1 Rappresentazione Pezzocentrica

Si definisce rappresentazione pezzocentrica, un qualsiasi tipo di rappresentazione della scacchiera che mantiene liste array o set dei pezzi attualmente presenti sulla scacchiera con annesse le informazioni sulle caselle da essi occupate. Le rappresentazioni più comuni sono:

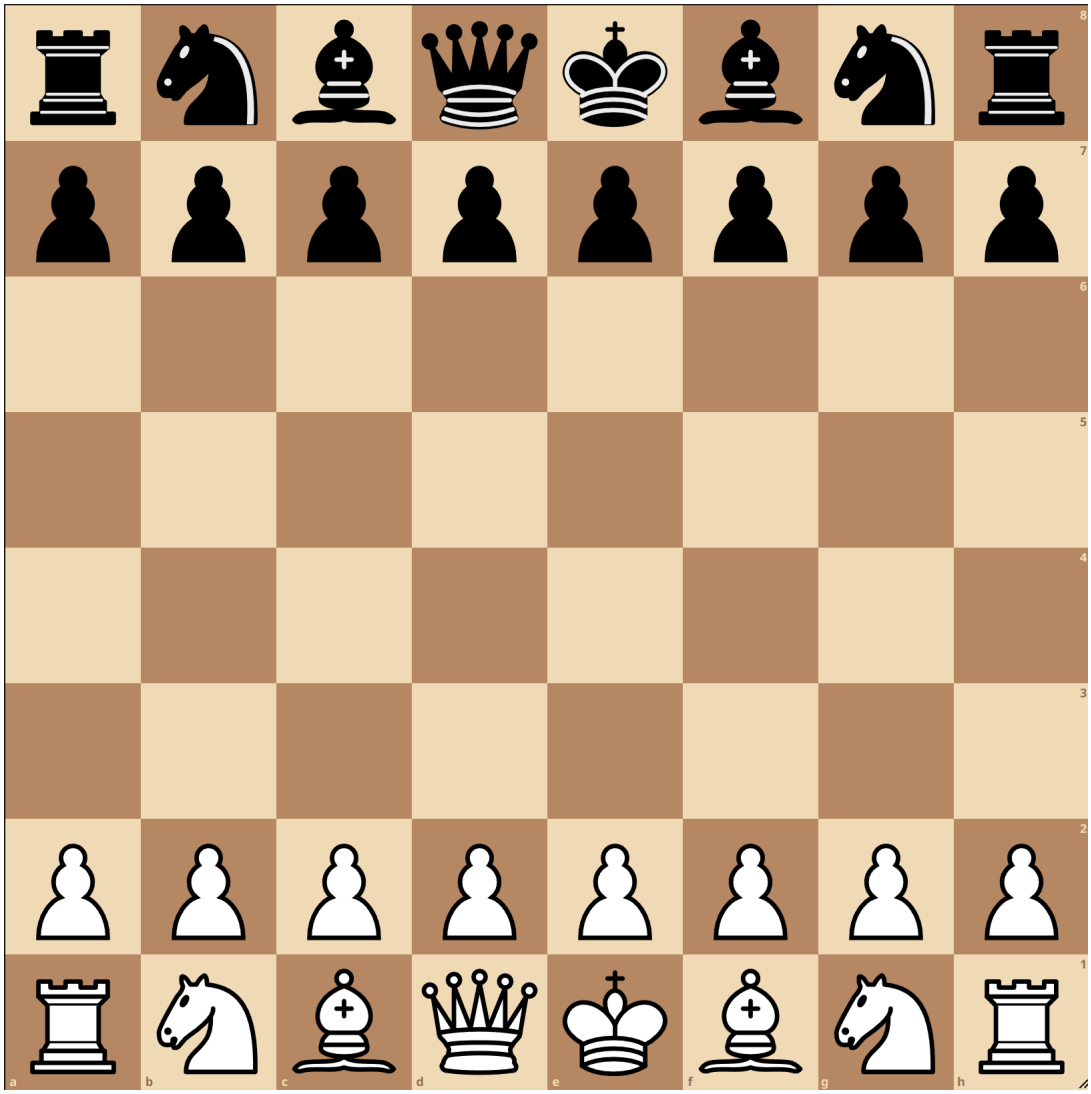
Piece-Lists

liste o array di ogni pezzo sulla scacchiera,ogni elemento della lista o dell'array associa un pezzo alla casella che esso occupa .le caratteristiche di ogni pezzo (colore,tipo etc) possono essere associate all'indice dell'array in cui si trovano o essere presenti in ulteriori array o liste esterne. *un'implementazione pratica di una Piece-List verrà mostrata in seguito all'interno di questo capitolo*

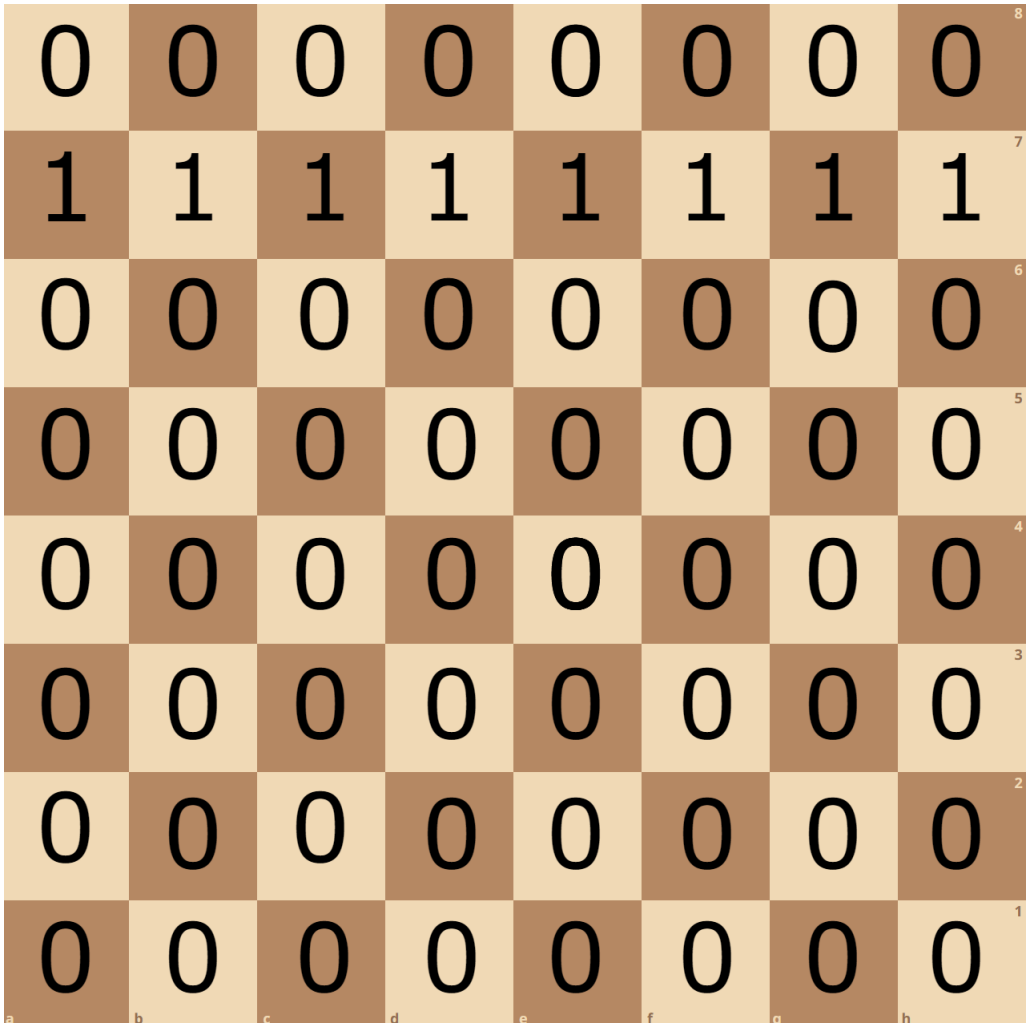
Bitboards

Una Bitboard è una struttura dati specifica per i giochi da tavolo, si tratta in sostanza di una struttura dati in grado di immagazzinare lo stato di ogni casella della scacchiera all'interno di una parola [2] di 64 bit. Vediamo un esempio pratico, immaginiamo di avere una scacchiera che si trova nello stato di default di inizio partita:

[2] Una parola è un gruppo di bit di una determinata dimensione che sono gestiti come unità dal set di istruzioni o dall'hardware di un processore



Una bitboard tipica è quella che ci permette di sapere in quali caselle è presente un pedone nero,per costruirla, operando casella per casella, ci poniamo una domanda "in questa casella è presente un pedone nero?" se si allora quella casella viene marcata con un 1 , altrimenti viene marcata con uno 0, il risultato di questa traduzione diventa in questo caso:



Piece-Sets

rappresentazione con set con un bit per ogni pezzo dentro una parola a 32 bit o 2 parole a 16 bit (una per lato) i Piece-sets hanno delle somiglianze con le bitboards, ma ogni bit del set non è direttamente correlato ad una casella, ma ad un indice dentro ad una piece-list. Spesso la bit-position di un piece-set implica , di che tipo e colore il pezzo è. - mentre le bitboards solitamente mantengono set distinti per pezzi diversi.

2.2.2 Rappresentazione Casellocentrica

Mailbox

8x8 Board

10x12 Board

Vector Attacks

0x88

2.3 Move Generation

2.4 Ricerca

2.5 Valutazione

2.6 libro delle aperture,Tablebases

CAPITOLO 3

Validazione preliminare

BREVE SPIEGAZIONE CONTENUTO CAPITOLO

Introduzione al concetto di ELO,rappresentazione ELO stimata del motore,metriche prestazionali varie

CAPITOLO 4

Stato dell'arte per Algoritmi di intelligenza artificiale e motori scacchistici per il gioco degli scacchi

Questo capitolo illustra lo stato dell'arte e i lavori presenti in letteratura sugli aspetti di ricerca trattati nel nostro studio. ECC ECC...

4.1 Pre Neural Network Stockfish

4.1.1 Board Representation

8x8 Board Bitboards with Little-Endian Rank-File Mapping (LERF) Magic Bitboards BMI2 - PEXT Bitboards (not recommend for AMD Ryzen [28] prior to Zen 3)

4.1.2 Search

Iterative Deepening Aspiration Windows Parallel Search using Threads YBWC prior to Stockfish 7 Lazy SMP since Stockfish 7, January 2016 Principal Variation Search Transposition Table Shared Hash Table 10 Bytes per Entry, 3 Entries per Cluster Depth-preferred Replacement Strategy No PV-Node probing Prefetch Move Ordering Countermove Heuristic Counter Moves History since Stockfish 7, January 2016 [32] History Heuristic Internal Iterative Deepening Killer Heuristic MVV/LVA SEE Selectivity Extensions Check Extensions if SEE >= 0 Restricted Singular Extensions Pruning Futility Pruning Move Count Based Pruning Null Move Pruning Dynamic Depth Reduction based on depth and value Static Null Move Pruning Verification search at high depths ProbCut SEE Pruning Reductions Late Move Reductions Razoring Quiescence Search

4.1.3 Evaluation

4.1.3.1 Material

Bishop Pair Imbalance Tables Material Hash Table

4.1.3.2 Mobility

Trapped Pieces Rooks on (Semi) Open Files

4.1.3.3 Pawn Structure

Pawn Hash Table

Backward Pawn

Doubled Pawn

Isolated Pawn

Phalanx

Connected Pawns

Passed Pawn

King Safety

Attacking King Zone Pawn Shelter Pawn Storm Square Control

Tapered Eval

Evaluation Patterns

Piece-Square Tables

Space

Outposts

4.2 Google AlphaZero

4.3 Post Neural Networ Stockfish-NNUE e LCZero

Nonostante le critiche ed i dubbi sulla performance di AlphaZero, gli appassionati di scacchi computazionali non rimasero impassibili davanti ai meriti di un approccio orientato alle reti neurali,il 6 agosto del 2020, un anno e mezzo dopo la pubblicazione definitiva dell'articolo su AlphaZero da parte di DeepMind e dopo un anno di lavoro, viene ufficialmente introdotta ,all'interno della repo di Stockfish,NNUE. NNUE, acronimo di "Effeciently Upgradable Neural Network" scritto da sinistra a destra, è una rete neurale per la valutazione di posizioni shogi, alle quali assegna un punteggio utilizzato poi in fase di potatura,adattata per operare sugli scacchi ed essere integrata in Stockfish. In questa versione Stockfish mantiene le caratteristiche principali che contraddistiguono la sua versione precedente, e la valutazione NNUE viene utilizzata solo in posizioni materialmente bilanciate

CAPITOLO 5

Conclusioni e Sviluppi Futuri

BREVE SPIEGAZIONE CONTENUTO CAPITOLO

Ringraziamenti

INSERIRE RINGRAZIAMENTI QUI