UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
CURSO DE CIÊNCIA DA COMPUTAÇÃO

RUI FELIPE DE OLIVEIRA CARDOZO

Desenvolvimento de um Agente de Inteligência Artificial para o Jogo de Xadrez Baseado no Algoritmo Minimax

Monografia apresentada como requisito parcial para a obtenção do grau de Bacharel em Ciência da Computação

Orientador: Prof. Dr. Renato Perez Ribas

Porto Alegre
2025

À meu querido professor, orientador e amigo Dr. Renato Perez Ribas, cuja orientação, paciência e dedicação foram essenciais para a concretização deste projeto. Seu exemplo de excelência acadêmica e seu compromisso com o aprendizado foram inspiração constante ao longo desta jornada.

À minha amada esposa Elisângela, cuja presença e apoio incondicional tornaram possível a superação de cada desafio.

E, com especial saudade e amor, dedico aos meus amados pais, Elena e Rui, que continuam vivos em minhas lembranças e em meu coração.

RESUMO

Neste trabalho foi realizado um estudo sobre a análise e o desenvolvimento de um agente de xadrez utilizando o clássico algoritmo de busca competitiva Minimax, em conjunto com técnicas de otimização como a poda alfa-beta e a tabela de transposição. O estudo abrange desde a história do xadrez até o desenvolvimento de um algoritmo capaz de analisar posições do tabuleiro e calcular as melhores jogadas, junto com a infraestrutura necessária para o funcionamento do algoritmo. O agente desenvolvido permite análises em diferentes níveis de profundidade. O trabalho foi implementado utilizando a linguagem de programação Java e aborda a otimização de desempenho por meio da poda alfa-beta para reduzir o número de nós a serem avaliados. Também foi descrita a pesquisa de quiescência, que trata de táticas explosivas e trocas de peças, e a tabela de transposição, baseada no algoritmo de Zobrist Hashing, que permite armazenar e reutilizar avaliações já realizadas. O projeto teve como objetivo oferecer uma plataforma eficiente para jogadores e aprendizes de diferentes níveis, proporcionando partidas desafiadoras. Além disso, este trabalho propôs uma plataforma para atender a um projeto pedagógico com o objetivo de melhorar e difundir o ensino de xadrez para crianças. Foram também analisados códigos e abordagens empregadas pelo agente Stockfish, destacando as melhorias obtidas em desempenho e qualidade das avaliações, fornecendo uma base teórica para acadêmicos interessados no desenvolvimento de agentes para jogos e tabuleiros.

Palavras-chave: Xadrez. minimax. decisão competitiva. jogos de tabuleiro. heurística posicional.

Development of an Artificial Intelligence Agent for the Game of Chess Based on the Minimax Algorithm

ABSTRACT

In this work, a study was conducted on the analysis and development of a chess agent using the classic Minimax competitive search algorithm, combined with optimization techniques such as alpha-beta pruning and transposition tables. The study covers everything from the history of chess to the development of an algorithm capable of analyzing board positions and calculating the best moves, along with the necessary infrastructure for the algorithm to function. The developed agent allows for analyses at different depth levels. The project was implemented using the Java programming language and addresses performance optimization through alpha-beta pruning to reduce the number of nodes to be evaluated. Quiescence search, which deals with tactical explosions and piece exchanges, was also described, as well as transposition tables based on the Zobrist Hashing algorithm, which enables the storage and reuse of previously evaluated positions. The project aimed to provide an efficient platform for players and learners of various skill levels, offering challenging matches. Furthermore, this work proposed a platform to support a pedagogical project aimed at improving and promoting chess education for children. Additionally, codes and approaches employed by the Stockfish agent were analyzed, highlighting performance and evaluation quality improvements, providing a theoretical foundation for academics interested in the development of agents for games and board games.

Keywords: Chess, minimax, competitive decision, board games, positional heuristic.

LISTA DE FIGURAS

Figura 2.1 Mechanical Turk 19

Figura 3.1 Bitboard 32
Figura 3.2 JBitboard 33
Figura 3.3 Opeação entre Bitboards 35
Figura 3.4 Exemplo de operações com bitboards na Linguagem Java 36

Figura 4.1 Exemplo de Árvore do Algoritmo Minimax 47

Figura 6.1 Evolução do Elo ao longo das fases do algoritmo com técnicas adicionais .. 77

Figura 7.1 Interface Gráfica Adaptada: tabuleiro reduzido. 80
Figura 7.2 Interface gráfica adaptada: tabuleiro 8x8. 81

Figura A.1 Chaturanga 85

Figura B.1 Judit Polgar 101

Figura C.1 Exemplo Minimax 109
Figura C.2 Algoritmo Minimax 110

Figura D.1 Interface JChess posição inicial 127
Figura D.2 Interface JChess durante uma partida 128
Figura D.3 Interface JChess em configuração do tabuleiro 130

LISTA DE ABREVIATURAS E SIGLAS

CCC	Computer Chess Championship
CCRL	Computer Chess Rating Lists
CM	Candidato a Mestre
CPU	Central Process Unit
FIDE	Federação Internacional de Xadrez
FM	Mestre FIDE
GM	Grande Mestre
IA	Inteligência Artificial
JVM	Java Virtual Machine
Lc0	Leela Zero
MCTS	Monte Carlo Tree Search
MI	Mestre Internacional
NNUE	Efficiently Updatable Neural Network
TCEC	Top Chess Engine Championship
TPU	Tensor Processing Units)
UCT	Upper Confidence Bound applied to Trees
WIM	Mestre Internacional Feminino
WFM	Mestre FIDE Feminino
WGM	Grande Mestre Feminino

SUMÁRIO

1 INTRODUÇÃO 13
1.1 Motivação 16
1.2 Proposta 16
1.3 Estrutura do Texto 17
2 XADREZ E A INTELIGÊNCIA ARTIFICIAL 18
2.1 Agentes 18
2.1.1 Mechanical Turk 19
2.1.2 Stockfish 20
2.1.3 AlphaZero 21
2.1.4 Leela Zero 22
2.1.5 Komodo 23
2.1.6 Houdini 24
3 INFRAESTRUTURA PARA CONSTRUÇÃO DE UM AGENTE DE XADREZ 26
3.1 Notações de Xadrez 26
3.1.1 Notação Algébrica 26
3.1.1.1 Descrição de Movimentos 27
3.1.1.2 Jogadas Especiais 27
3.1.2 Notação Origem-Destino 27
3.1.2.1 Características da Notação Origem-Destino 28
3.1.2.2 Exemplos 28
3.1.2.3 Comparação com Outras Notações 28
3.1.3 Notação FEN (Forsyth-Edwards Notation) 29
3.1.3.1 Formato da Notação FEN 29
3.1.4 Notação PGN (Portable Game Notation) 30
3.1.4.1 Estrutura da Notação PGN 30
3.2 Bitboards 31
3.2.1 Aritmética 34
3.2.1.1 Operações Lógicas 34
3.2.1.2 Operações de Deslocamento 35
3.2.1.3 Exemplo de implementação com bitboards 35
3.2.2 Bitboards Mágicos 36
3.3 Tabela de Transposição 38
3.3.1 Objetivos 38
3.3.2 Implementação 39
3.3.3 Reuso da Informação 39
3.3.4 Impacto na Performance 40
3.3.5 Limitações 40
3.4 Gerador de Movimento 40
3.4.1 O Processo de Geração de Movimentos 41
3.4.2 Classificação e Seleção de Movimentos 41
3.4.3 Ordenação dos Movimentos 41
3.4.4 Busca Quiescente e Geração de Movimentos Especiais 42
3.4.5 Aplicação de Heurísticas no Contexto de Evasão e Capturas 42
3.5 Pesquisa de Quiescencia 43
3.5.1 O Problema do Horizonte de Pesquisas 43
3.5.2 Benefícios da Pesquisa de Quiescência 44
3.5.3 Limitações 44

4 TÉCNICAS PARA DETERMINAR A MELHOR JOGADA 45
4.1 Algoritmo Minimax 45
4.1.1 Funcionamento 45
4.1.2 Poda Alfa Beta 46
4.1.3 Exemplo de Árvore Minimax 46
4.1.4 Funções de Avaliação 48
4.1.5 Expansão combinatória 48
4.2 Pesquisa Monte Carlo (MCTS) 48
4.2.1 Estrutura Básica do Algoritmo Monte Carlo 49
4.2.1.1 Seleção com o UCT 49
4.2.1.2 Expansão 50
4.2.1.3 Simulação 50
4.2.1.4 Retropropagação 50
4.2.2 Aplicações da MCTS nos Agentes de Xadrez 51
4.2.3 Benefícios 51
4.2.4 Limitações 51
4.3 Heurística de Avaliação Posicional 52
4.4 Componentes da Avaliação Posicional 52
4.4.1 Valor Material 52
4.4.2 Atividade e Mobilidade 53
4.4.3 Controle do Centro 53
4.4.4 Estrutura de Peões 53
4.4.5 Segurança do Rei 53
4.4.6 Par de Bispos 54
4.5 Combinações e Pesos na Heurística de Avaliação 54
4.6 Evolução e Aprimoramentos da Avaliação Posicional 55
4.6.1 Avaliação Automatizada com Redes Neurais 55
4.6.2 Avaliação Dinâmica 55
4.7 Limitações 55
5 DESENVOLVIMENTO DE UM AGENTE DE XADREZ 57
5.1 Ambiente de Desenvolvimento 57
5.1.1 Hardware 57
5.1.2 Software 58
5.2 Implementação Inicial 58
5.2.1 Minimax Puro 58
5.2.2 Tabela de Valores de Posição (PST) 59
5.2.3 Poda Alfa-Beta 59
5.2.4 Ordenação PV 60
5.2.5 Tabela de Transposição 60
5.2.6 Aprimoramento da Função de Avaliação 61
5.2.7 Reflexão Sobre o Desenvolvimento da Implementação Inicial 61
5.3 Incorporação de Técnicas Avançadas do Stockfish 61
5.3.1 Geração de Movimentos (classe *movegen*) 62
5.3.2 Lista de Movimentos (classe *movepicker*) 62
5.3.2.1 Etapa de Movimentos Quietos 63
5.3.2.2 Etapa de Movimentos de Refutação 63
5.3.2.3 Etapa de Capturas Ruins 63
5.3.2.4 Etapa de Evasão 63
5.3.2.5 Etapa de Probcut 64
5.3.3 Poda Agressiva no Minimax do Stockfish 64
5.3.3.1 Poda de Distância de Mate 65

5.3.3.2 Razoring 65
5.3.3.3 Poda de Futilidade 65
5.3.3.4 Busca de Movimento Nulo 65
5.3.3.5 ProbCut 65
5.3.3.6 Poda em Profundidade Rasa 66
5.3.3.7 Extensões 66
5.3.3.8 Busca de Profundidade Reduzida (LMR) 66
5.4 Avaliação de Tabuleiros no Xadrez por Motores como o Stockfish 66
5.4.1 Estrutura de Peões 66
5.4.2 Controle do Tabuleiro 67
5.4.3 Atividade e Mobilidade das Peças 67
5.4.4 Segurança do Rei 67
5.4.5 Avaliação Material 67
5.4.6 Avaliação Tática 68
5.4.7 Avaliação Posicional 68
5.4.8 Avaliação Dinâmica 68
5.4.9 Avaliação de Finais 68
5.5 Reflexão sobre o Desenvolvimento da Agentes de Xadrez 69
6 ANÁLISE E VALIDAÇÃO DE NOSSO AGENTE 70
6.1 Testes Funcionais 70
6.2 Metodologia de Testes de Desempenho 70
6.3 Cálculo do Elo 71
6.4 Validação por Incrementos 72
6.5 Tabela Comparativa das Medições da Primeira Etapa 72
6.6 Incorporação das Técnicas Implementadas no Stockfish 74
6.7 Etapas de Incorporação 74
6.7.1 Geração de Jogadas com Bitboards Mágicos 74
6.7.2 Função de Avaliação 74
6.7.3 Adaptação do Minimax para a Poda Agressiva 75
6.8 Validação com Trace 75
6.9 Resultados e Limitações 76
6.10 Reflexão 77
7 ADAPTAÇÃO DO AGENTE PARA JOGOS PRÉ-ENXADRISTICOS 78
7.1 Tabuleiro Reduzido e Regras Simplificadas 78
7.2 Peças Auxiliares e Estratégias Avançadas 79
7.3 Interação Entre Crianças e Contra o Agente 79
7.4 Interface Adaptada 80
7.5 Benefícios Pedagógicos 81
7.6 Reflexão 82
8 CONCLUSÃO 83
APÊNDICE A — TEORIA DO XADREZ 85
A.1 O Desenho do Jogo 85
A.2 As Regras 86
A.2.1 A Evolução das Regras e Peças 86
A.2.2 Regras Atuais 87
A.2.3 O Enfoque Prático 87
A.2.4 Teoria do Ataque 88
A.2.5 Teoria do Equilíbrio 89
A.2.6 Xadrez 960 90
A.2.7 Outras Notações de Xadrez 91
A.2.7.1 Notação Descritiva 91

A.2.7.2 Notação ICCF 91
APÊNDICE B — XADREZ COMO ESPORTE 92
B.1 Torneios 92
B.1.1 FIDE 92
B.1.2 O Sistema de Ranking Elo 93
B.1.2.1 Funcionamento 93
B.1.2.2 Fórmula 93
B.1.2.3 Pontuação Esperada 94
B.1.2.4 Exemplo 94
B.1.2.5 Vantagens do sistema Elo 96
B.1.2.6 Limitações 96
B.1.2.7 Aplicações do sistema Elo 96
B.1.3 O Fator Tempo nos Torneios 97
B.1.4 Hierarquia e Ranking 97
B.1.5 Jogadores de Relevância Internacional 99
B.1.5.1 Jogadores Históricos 99
B.1.5.2 Jogadores Contemporâneos 99
B.2 Xadrez na Sociedade 100
B.2.1 Xadrez como Ferramenta Pedagógica 100
B.2.2 Mulheres no Xadrez 101
B.2.3 Xadrez na Interação e Inclusão Social 102
APÊNDICE C — COMPLEMENTAÇÃO DA INFRAESTRUTURA DE UM AGENTE 103
C.1 Estágios da Geração de Movimentos 103
C.2 Outras operações de Bitboard 103
C.2.0.1 Operações Morfológicas 103
C.2.0.2 Operação de Rotação 104
C.3 Função Zobrist Hashing 105
C.3.1 Como a função Zobrist funciona? 105
C.3.2 Cálculo do Hash Zobrist 105
C.3.3 Exemplo de Cálculo 106
C.3.4 Vantagens do Zobrist Hashing 106
C.4 Fases da Pesquisa de Quiescência 107
C.5 Exemplo de Funcionamento do Algoritmo Minimax com Poda Alfa-Beta 107
C.5.1 Conceitos Fundamentais 108
C.5.2 Funcionamento do Algoritmo 108
C.6 Redes Neurais e Algoritmos Baseados em Aprendizado de Máquina 110
C.6.1 Processo de Aprendizado em Redes Neurais 111
C.6.2 Arquiteturas de Redes Neurais 111
C.6.3 Estrutura e Componentes das CNNs em Agentes de Xadrez 113
C.6.4 Algoritmos de Aprendizado de Máquina 113
C.6.5 Integração de Redes Neurais com Aprendizado por Reforço 114
C.6.6 Arquitetura NNUE do Stockfish 114
C.6.7 Aplicações Futuras e Potenciais de Redes Neurais 115
C.6.8 Limitações 115
C.7 Detalhamento da Função de Avaliação de Agentes Modernos 116
C.7.1 Estrutura de Peões 116
C.7.2 Controle do Tabuleiro 117
C.7.3 Atividade e Mobilidade das Peças 118
C.7.4 Segurança do Rei 119
C.7.5 Material 120

C.7.6 Avaliação Tática 121
C.7.7 Avaliação Posicional 121
C.7.8 Avaliação Dinâmica 122
C.7.9 Avaliação de Finais 124
C.7.10 Outros Fatores 125
APÊNDICE D — INTERFACE GRÁFICA PADRONIZADA PARA O AGENTE 126
D.1 Elementos Essenciais de uma Interface Gráfica de Xadrez 126
D.2 Descrição da Interface do Agente de Xadrez 127
D.2.1 Ambiente de Jogo 128
D.2.2 Botões de Controle de Jogo 129
D.2.3 Ambiente de Configuração do Tabuleiro 129
D.2.4 Botões de Configuração 129
D.3 Controle de Tempo 130
D.4 Personalização e Configuração 131
D.5 Comandos Comuns em Interfaces de Xadrez 131
D.5.1 Comando `position fen` 131
D.5.2 Comando `go depth 8` 132
D.5.3 Comando `uci` 132
D.5.4 Comando `isready` 132
D.5.5 Comando `setoption name ...` 132
D.5.6 Comando `stop` 133
D.5.7 Comando `quit` 133
D.6 Desafios e Limitações das Interfaces de Xadrez 133
APÊNDICE E — ANÁLISE E DESEMPENHO DOS ALGORITMOS 134
E.1 Introdução à Análise de Desempenho 134
E.2 Métricas de Avaliação de Desempenho 134
E.3 Técnicas de Análise de Desempenho 135
E.3.1 Análise Empírica 135
E.3.2 Análise Assintótica 135
E.3.3 Profiling e Otimização de Código 136
E.4 Desafios Específicos na Avaliação de Algoritmos de Xadrez 136
E.4.1 Complexidade da Heurística de Avaliação Posicional 136
E.4.2 Profundidade de Busca e Recursos Computacionais 136
E.4.3 Efeito das Tabelas de Transposição 137
E.4.4 Poda alfa beta 137
E.4.5 Funcionamento da Poda Alfa-Beta 137
E.4.6 Ordenação dos Nós para Eficiência da Poda 138
APÊNDICE — REFERÊNCIAS 139

1 INTRODUÇÃO

O xadrez, um dos jogos mais antigos e reverenciados da história, transcende gerações como uma combinação única de arte, ciência e esporte. Originado na Índia há mais de mil anos, evoluiu através de diversas culturas até consolidar suas regras modernas no século XIX. Muito mais do que um jogo, o xadrez é reconhecido por sua capacidade de estimular o raciocínio lógico, a concentração e o pensamento estratégico, tornando-se uma ferramenta valiosa em contextos educacionais e competitivos.

Com o advento da tecnologia, o xadrez encontrou novos horizontes. A união entre o jogo e o computador começou a despontar em meados do século XX, com o desenvolvimento de algoritmos que possibilitavam aos computadores jogar contra humanos. Desde então, o progresso dos agentes de xadrez revolucionou a forma como o jogo é analisado, estudado e praticado.

O xadrez é, portanto, um ambiente de teste particularmente apropriado para um grande número de técnicas de busca e aprendizado de máquina (WINIEWSKA; WOJCIK, 2022). Seu espaço de estados é amplo e bem definido, o que exige que os agentes de xadrez sejam eficientes e precisas em suas avaliações.

Entre as contribuições mais notáveis, destacam-se os agentes como o Stockfish, que incorporam algoritmos sofisticados de busca e heurísticas de avaliação posicional. Tais avanços trouxeram uma nova era de acessibilidade e aprendizado, permitindo que jogadores de diferentes níveis utilizem essas ferramentas para melhorar suas habilidades. Paralelamente, projetos como o AlphaZero, que une aprendizado por reforço e redes neurais, ampliaram ainda mais o potencial da inteligência artificial aplicada ao xadrez, superando até mesmo os limites de agentes tradicionais.

A popularização das plataformas online de xadrez, como Chess.com e Lichess, democratizou o acesso ao jogo, permitindo que milhões de pessoas ao redor do mundo joguem, aprendam e se conectem com outros entusiastas. Essas plataformas oferecem uma vasta gama de recursos, desde tutoriais interativos até análises detalhadas de partidas, contribuindo para o crescimento contínuo da comunidade enxadrística.

O xadrez é um dos jogos mais antigos e populares do mundo, com origens que remontam a mais de 1.500 anos. Surgiu, provavelmente, na Índia, por volta do século VI, onde era conhecido como Chaturanga (CAZAUX; KNOWLTON, 2017). Essa versão inicial já possuía algumas peças e movimentos que se assemelham ao xadrez moderno (MURRAY, 1913). O jogo se espalhou para a Pérsia, onde recebeu o nome de Shatranj, e,

após a expansão árabe, chegou à Europa na Idade Média. As peças moviam-se em padrões distintos, com estilos não encontrado noutros jogos, como as damas. A vitória ou derrota baseava-se na movimentação das peças para proteger ou capturar uma peça chave do jogo, com a integridade da unidade referindo-se ao "rei" no xadrez atual (LINDER, 1994). No século XV, na Espanha e na Itália, o xadrez passou por significativas mudanças, como o movimento ampliado da dama e do bispo, tornando-se mais dinâmico. Essas adaptações deram forma ao xadrez moderno, que foi amplamente padronizado no século XIX, com o surgimento de torneios oficiais e a criação de regras unificadas.

A introdução do xadrez na Europa ocorreu através do Chaturanga ou Shatranj, que se disseminou pela Pérsia, Império Bizantino e, pelo crescente império árabe (YALOM, 2009). O registro mais antigo do jogo foi encontrado em um manuscrito do século X e documenta uma partida entre um historiador de Bagdá e seu aluno (MURRAY, 1913). Os muçulmanos levaram o xadrez para o norte da África, Sicília e Espanha no século X, enquanto os eslavos orientais o introduziram em Kiev na Rússia na mesma época(LINDER, 1994). Os vikings levaram o jogo à Islândia e Inglaterra, sendo responsáveis pela famosa coleção de 78 peças de marfim de morsa descobertas na Ilha de Lewis em 1831 e datadas do século XI ou XII (FISKE, 1905). No início, houve proibições periódicas impostas por reis e líderes religiosos, como a do rei Luís IX na França em 1254. Entretanto, o xadrez demonstrava um prestígio social associado à riqueza, conhecimento e poder, o que aumentou consideravelmente sua popularidade (MURRAY, 1913). O jogo era apreciado por monarcas como Henrique I, Henrique II, João e Ricardo I da Inglaterra, Filipe II e Alfonso X da Espanha, e Ivan IV da Rússia (YALOM, 2009).

Com a evolução do xadrez como esporte, surgiram grandes mestres e teóricos que contribuíram para o desenvolvimento estratégico do jogo. Jogadores como Wilhelm Steinitz, o primeiro campeão mundial oficial em 1886, e José Raúl Capablanca, famoso por seu estilo posicional e clareza, marcaram época. O século XX consolidou o xadrez como um esporte de alto rendimento, com a popularização de competições globais e o surgimento de campeões lendários, como Bobby Fischer, Anatoly Karpov e Garry Kasparov. As partidas passaram a ser estudadas em detalhes, impulsionadas por análises sistemáticas e pelo uso crescente de tecnologias. A busca por precisão e novas ideias foi o embrião para o surgimento dos primeiros agentes de xadrez.

Os agentes de xadrez, programas capazes de jogar e analisar partidas, começaram a se desenvolver no início da era dos computadores, por volta dos anos 1950. Um dos primeiros esforços foi realizado por Alan Turing, que criou um algoritmo simples,

embora não programado em um computador por limitações tecnológicas da época. Na década de 1960, surgiram os primeiros programas competentes, como o Mac Hack, desenvolvido no MIT. Com o aumento do poder de processamento nos anos 1970 e 1980, os programas começaram a competir contra jogadores humanos com relativo sucesso. Em 1997, a história do xadrez mudou drasticamente quando o supercomputador Deep Blue, desenvolvido pela IBM, derrotou Garry Kasparov, o então campeão mundial. Esse evento marcou o início da era em que as máquinas começaram a superar os humanos no xadrez.

Após a vitória do Deep Blue, o desenvolvimento de agentes se intensificou, com avanços em algoritmos de busca, heurísticas de avaliação e poder computacional. Surgiram agentes mais sofisticadas, como Fritz, Rybka e, posteriormente, o Stockfish, que é uma das mais fortes atualmente. Stockfish, de código aberto, usa a busca alfa-beta aprimorada e tabelas de transposição para avaliar milhões de posições por segundo. Além disso, outros agentes, como o Komodo, introduziram novas abordagens com análise posicional mais refinada. A disseminação de agentes revolucionou o estudo do xadrez, permitindo análises mais precisas e aprimorando o treinamento dos jogadores.

Nos últimos anos, o desenvolvimento da inteligência artificial levou os agentes a um novo patamar. Em 2017, a DeepMind, empresa da Alphabet, lançou o AlphaZero, que utiliza redes neurais e aprendizado por reforço. Diferentemente dos agentes tradicionais baseadas em força bruta e heurísticas humanas, o AlphaZero aprendeu xadrez do zero, apenas jogando contra si mesmo milhões de vezes. Em apenas quatro horas de treinamento, AlphaZero superou o Stockfish, impressionando pela criatividade e estilo de jogo "humano". Essa abordagem inovadora inspirou a criação de outros agentes baseadas em redes neurais, como o Leela Chess Zero (Lc0), que aplica um modelo de aprendizado semelhante ao AlphaZero, mas com código aberto e treinamento colaborativo.

Hoje, os agentes de xadrez desempenham um papel fundamental no desenvolvimento do jogo, seja no treinamento de jogadores, na análise de partidas ou no auxílio à arbitragem em torneios. Elas são ferramentas indispensáveis para enxadristas de todos os níveis, fornecendo insights precisos e explorando a profundidade estratégica do xadrez. Competições entre agentes, como o TCEC (Top Chess Engine Championship), demonstram a evolução constante desses programas. A combinação entre habilidades humanas e análise das máquinas impulsionou a compreensão do xadrez, tornando-o mais acessível, dinâmico e fascinante.

O incentivo a jogos de Xadrêz em escolas tem proporcionado um aumento de desempenho estudantil, especialmente em currículos educacionais de natureza lógica, in-

cluindo matemática e ciências (GOBET; CAMPITELLI, 2006). Jogos de tabuleiro possuem tradição de inclusão social e são promovidos em diversas comunidades e escolas para desenvolver educação, disciplina e trabalho em equipe.

1.1 Motivação

A criação de um agente de xadrez representa um desafio técnico e intelectual que desperta a curiosidade humana. Este trabalho busca compreender e implementar técnicas avançadas de inteligência artificial que são a base de agentes renomados, como minimax, heurísticas de avaliação posicional, tabelas de transposição e bitboards. Assim, a motivação que está por trás desse campo de pesquisa inclui a exploração de técnicas de inteligência artificial e algoritmos de otimização em curso. Isso não só fornece uma compreensão melhor e mais detalhada da teoria do jogo, mas também possibilita inovações práticas em um número crescente de áreas, incluindo logística, planejamento e decisão racional em sistemas complexos (SILVER et al., 2018a).

Estudar as abordagens utilizadas pelo Stockfish e outros agentes possibilita entender como otimizar algoritmos de busca e criar sistemas eficientes. Este aprendizado não se restringe ao campo acadêmico, mas também pode ser aplicado a ambientes escolares, onde o xadrez é uma poderosa ferramenta de desenvolvimento cognitivo e social.

1.2 Proposta

A proposta deste agente está alinhada com a curiosidade natural de explorar como as máquinas podem emular o raciocínio humano. Ao criar um software com total domínio sobre o núcleo de suas operações, é possível adaptá-lo para diversos contextos, como plataformas educacionais ou interfaces amigáveis para iniciantes no xadrez.

Este trabalho propõe o desenvolvimento de um agente de xadrez utilizando técnicas modernas de inteligência artificial. A escolha por implementar uma solução própria está ancorada na flexibilidade que um núcleo desenvolvido do zero proporciona.

Ao invés de depender exclusivamente de agentes existentes, a criação de um motor próprio permite personalizações profundas, como alterações no comportamento do algoritmo de busca ou a introdução de funcionalidades específicas para ambientes pedagógicos. Com isso, espera-se alcançar um equilíbrio entre desempenho técnico e adapta-

bilidade para fins diversos.

Além de buscar a excelência técnica, este projeto tem um viés educacional, utilizando o xadrez como meio para introduzir conceitos complexos de inteligência artificial e programação de maneira acessível a diferentes públicos.

1.3 Estrutura do Texto

O presente trabalho está estruturado de maneira a guiar o leitor pelos conceitos e técnicas fundamentais do desenvolvimento de um agente de xadrez.

No Capítulo 2, será abordado o histórico dos agentes e sua evolução, destacando projetos importantes como Stockfish, AlphaZero e Leela Zero. O objetivo é fornecer um panorama sobre como a inteligência artificial transformou o xadrez ao longo das décadas.

O Capítulo 3 explora as bases técnicas para a criação de um agente, como notações de xadrez, bitboards, tabelas de transposição e hash Zobrist. Estes elementos são essenciais para a implementação eficiente de algoritmos de busca e heurísticas de avaliação.

No Capítulo 4, será detalhada a aplicação de técnicas específicas para determinar a melhor jogada, com ênfase no algoritmo minimax, poda alfa-beta e heurísticas posicionais.

O Capítulo 5 apresenta o desenvolvimento prático do agente proposto, destacando as etapas de implementação inicial, aprimoramentos inspirados em técnicas do Stockfish e validação do desempenho.

No Capítulo 6, os resultados obtidos são analisados, seguidos de considerações sobre adaptações pedagógicas no Capítulo 7, onde o agente será discutido como ferramenta educacional para ambientes escolares.

Finalmente, o Capítulo 8 apresenta uma conclusão do projeto com perspectivas futuras. Os anexos complementam vários tópicos aboradados ao longo deste trabalho.

2 XADREZ E A INTELIGÊNCIA ARTIFICIAL

A inteligência artificial (IA) tem desempenhado um papel transformador no mundo do xadrez, tanto no desenvolvimento de programas que desafiam a capacidade humana quanto na evolução das estratégias de jogo. Desde a criação de programas como o Deep Blue, que derrotou o campeão mundial Garry Kasparov em 1997, a IA se tornou uma ferramenta essencial para jogadores de todos os níveis. Através de algoritmos sofisticados como o minimax e o uso de técnicas de poda alfa-beta, as máquinas conseguem analisar milhões de posições possíveis em uma fração de segundo, permitindo que façam decisões baseadas em cálculos profundos e uma vasta base de dados de partidas. Isso não apenas revolucionou o treinamento, como também estabeleceu novos padrões de jogabilidade, com agentes como Stockfish e AlphaZero servindo como guias para as melhores práticas de jogo (SILVER et al., 2018b).

Além de desafiar a supremacia humana, a IA também ampliou o alcance do xadrez, tornando o aprendizado e a análise mais acessíveis. Hoje, ferramentas baseadas em IA oferecem avaliações instantâneas de posições, sugerem movimentos e até explicam os raciocínios por trás das recomendações, tornando o xadrez mais acessível para iniciantes e profissionais. Esse avanço foi possível graças à integração de técnicas de aprendizado profundo, que permitem que as máquinas aprendam com experiências passadas, como no caso do AlphaZero, que aprendeu a jogar xadrez sem conhecimento prévio, apenas jogando contra si mesmo. A IA, portanto, não só elevou o nível de jogo, como também democratizou o acesso ao conhecimento estratégico.

2.1 Agentes

Um agente de xadrez é um software especializado em calcular e avaliar posições no jogo de xadrez, utilizando algoritmos de busca e heurísticas para determinar o melhor movimento possível em um dado momento. Ela opera por meio de uma análise extensiva do tabuleiro, explorando uma grande árvore de possibilidades de jogadas futuras. Agentes modernos de xadrez fazem uso de algoritmos de busca como o minimax combinado com poda alfa-beta para reduzir o número de posições analisadas, focando apenas nas mais promissoras. Esses agentes integram funções de avaliação, que atribuem valores numéricos às posições no tabuleiro com base em fatores como material, controle de espaço, segurança do rei, e estrutura de peões, com o objetivo de aproximar a qualidade da

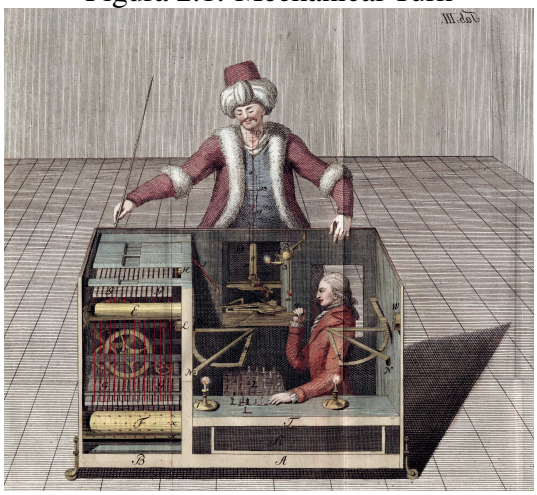
posição em termos de vitória, empate ou derrota.

Agentes mais avançados também incorporam aprendizado de máquina que utiliza redes neurais e aprendizado por reforço para aprimorar suas habilidades. Elas podem aprender jogando milhares de partidas contra si mesmas, ajustando continuamente suas avaliações com base nos resultados. Essas ferramentas são amplamente utilizadas tanto por jogadores amadores quanto por profissionais para análise pós-partida, preparação para torneios e estudo de aberturas e finais de jogo. Além disso, muitos desses agentes podem ser integradas com interfaces gráficas e plataformas de análise online, permitindo acesso fácil para usuários de diferentes níveis.

2.1.1 Mechanical Turk

O Mechanical Turk, construído em 1770 por Wolfgang von Kempelen, era uma máquina fraudulenta que simulava jogar xadrez contra humanos. Embora exibida por 84 anos como um autômato capaz de vencer oponentes, incluindo figuras como Napoleão Bonaparte e Benjamin Franklin, o Turk era, na verdade, controlado por um mestre de xadrez escondido em seu interior. Mesmo suspeitada por alguns, a fraude só foi revelada publicamente após sua destruição em um incêndio em 1854. Durante suas exibições, vários mestres de xadrez secretamente operaram a máquina, mas os operadores originais permanecem desconhecidos(Wikipedia contributors, 2024).

Figura 2.1: Mechanical Turk



Seção transversal do projeto de Racknitz da Mechanical Turk Fonte: (Wikipedia contributors, 2024)

Durante a existência do Mechanical Turk, muitos livros e artigos tentaram ex-

plicar seu funcionamento, mas a maioria estava incorreta. A revelação completa de seu segredo só ocorreu com Silas Mitchell, filho do último dono, que expôs que um mestre de xadrez operava a máquina por dentro. Outros artigos ao longo dos anos forneceram novos detalhes, como a utilização de uma vela dentro do gabinete para iluminar o operador. O interesse pelo Turk foi renovado com a criação do Deep Blue, o supercomputador da IBM, e resultou na publicação de mais livros e documentários comparando as duas máquinas(Wikipedia contributors, 2024).

2.1.2 Stockfish

Stockfish é um poderoso agente de xadrez gratuito e de código aberto, amplamente considerada a mais forte do mundo em 2024, com um Elo 3642, com controle de tempo 40/15 (40 movimentos em 15 minutos), de acordo com o CCRL. Criada inicialmente por Tord Romstad e outros colaboradores, é descendente do Agente Glaurung (2004). Stockfish se destaca por seu algoritmo de busca alfa-beta otimizado e suporte a grandes tabelas de transposição, chegando a utilizar até 1024 threads em CPUs multicore e até 32 terabytes de memória para análise (CONTRIBUTORS, 2024a). A CCRL é uma das organizações mais conhecidas para classificação de motores de xadrez, sendo frequentemente usada para comparar a força relativa entre diferentes agentes, como Stockfish, Komodo, Leela Chess Zero, entre outras.

Em 2013, foi introduzido o Fishtest, uma plataforma distribuída de testes que permite que voluntários contribuam com tempo de CPU para testar novas funcionalidades. Isso ajudou Stockfish a melhorar rapidamente, elevando seu Elo em 120 pontos em apenas um ano. O agente utiliza bitboards e estratégias avançadas de poda e redução de movimentos tardios, o que a torna extremamente eficiente em buscas profundas (JONES, 2022).

Em 2020, foi lançada a versão Stockfish NNUE, que incorpora redes neurais para melhorar ainda mais o desempenho. Desde então, Stockfish continua a dominar campeonatos de agentes, como o TCEC, CCC e competições do *chess.com*. Embora desafiada por motores de inteligência artificial como Leela Chess Zero, Stockfish mantém sua posição de liderança em muitas competições (CONTRIBUTORS, 2024g).

Stockfish é o maior vencedor da história do TCEC, onde figurou entre os dois primeiros colocados em praticamente todas as temporadas desde 2013. Ele também é dominante em outras competições, como o *Chess.com Computer Chess Championship*,

onde tem sido o agente mais bem-sucedida desde a criação do torneio.

Embora tenha enfrentado concorrência de motores de inteligência artificial como Leela Chess Zero, Stockfish continua a liderar muitas competições. O motor tem diversas variantes, como Fairy Stockfish, que foca em variantes regionais de xadrez, e ShashChess, que aplica métodos de Alexander Shashin para encontrar movimentos fortes (JONES, 2022).

Em uma partida de exibição contra o grande mestre Hikaru Nakamura (classificação 2800) em agosto de 2014, o Stockfish na versão 5 mostrou sua superioridade em relação aos melhores jogadores humanos. Mesmo sem seu livro de abertura e sem sua tabela de finais, o Stockfish venceu Nakamura por 1,5 a 0,5. O agente Stockfish rodou em um Mac Pro de 8 cores 3GHz com OS X como Sistema Operacional (CONTRIBUTORS, 2024a).

2.1.3 AlphaZero

AlphaZero é um agente de xadrez desenvolvido pela DeepMind, uma empresa subsidiária do Google, que usa um método revolucionário de aprendizado por reforço profundo. Diferentemente de motores tradicionais de xadrez, como o Stockfish, que dependem de pesquisa alfa-beta e heurísticas pré-programadas, o AlphaZero utiliza uma abordagem de rede neural profunda que aprende a jogar xadrez por meio de autotreinamento. Em vez de ser alimentado com conhecimento especializado, AlphaZero começou com as regras básicas do jogo e jogou milhões de partidas contra si mesmo, ajustando sua rede neural conforme evoluía. Isso permitiu que ele desenvolvesse estratégias criativas e altamente eficientes, surpreendendo até mesmo jogadores de xadrez humanos de elite. Dentro de 24 horas de treinamento, ele superou os campeões mundiais de xadrez (Stockfish), shogi (Elmo) e go (AlphaGo Zero), todos programas líderes em seus respectivos jogos (SILVER DAVID; HUBERT, 2017).

Do ponto de vista técnico, AlphaZero combina uma rede neural convolucional com um método de busca Monte Carlo Tree Search (MCTS). A rede neural prevê os melhores movimentos e a probabilidade de vitória a partir de uma determinada posição no tabuleiro, enquanto o MCTS é usado para explorar várias ramificações possíveis durante o jogo. A rede neural é constantemente refinada com base nos resultados dessas buscas, e o modelo se ajusta de maneira autônoma. O AlphaZero também foi projetado para operar em um ambiente de hardware altamente paralelo, utilizando grandes conjuntos de

TPUs (Tensor Processing Units), que são chips especializados em acelerar o treinamento de redes neurais profundas (CONTRIBUTORS, 2024c).

O treinamento do AlphaZero foi realizado usando unidades de processamento TPUs, otimizadas para os algoritmos de aprendizado profundo do Google. Para o xadrez, ele foi treinado por apenas nove horas e, em uma competição de 100 partidas, derrotou o Stockfish 8 com 28 vitórias, 0 derrotas e 72 empates. O AlphaZero realizou esse feito utilizando uma única máquina equipada com quatro TPUs, sem depender de livros de abertura ou tabelas de finais, ao contrário dos motores de xadrez tradicionais (CONTRIBUTORS, 2024c).

Em termos de desempenho, AlphaZero é conhecido por seu estilo de jogo intuitivo e agressivo, muitas vezes priorizando a ocupação estratégica do tabuleiro e a iniciativa em vez de acumular pequenas vantagens materiais. Seus jogos contra o Stockfish, outro dos motores mais fortes do mundo, mostraram a capacidade do AlphaZero de vencer com uma abordagem menos materialista e mais focada em coordenação e pressão. AlphaZero impressionou a comunidade enxadrística ao derrotar Stockfish em 2017, após apenas quatro horas de autotreinamento, provando a eficácia de sua abordagem baseada em redes neurais e aprendizado por reforço profundo. Ele sacrificou peças importantes, como damas e bispos, para explorar vantagens posicionais, um estilo de jogo considerado "alienígena"por alguns jogadores, como Demis Hassabis, fundador da DeepMind. Apesar das críticas de que o Stockfish não foi executado com configurações ideais, a vitória do AlphaZero foi amplamente reconhecida como um marco no desenvolvimento de IA para jogos de tabuleiro (KLEIN, 2017).

O impacto do AlphaZero foi tão significativo que inspirou o desenvolvimento de outros agentes baseadas em redes neurais, como o Leela Chess Zero. Mesmo sem ter seu código liberado ao público, o AlphaZero influenciou profundamente a comunidade de inteligência artificial e xadrez, levando a avanços contínuos no campo das redes neurais e aprendizado por reforço profundo.

2.1.4 Leela Zero

Leela Chess Zero (Lc0) é um agente de xadrez de código aberto baseado em inteligência artificial, inspirado no algoritmo AlphaZero da DeepMind. Ao contrário dos motores de xadrez tradicionais, que utilizam métodos de busca baseados em força bruta, como o algoritmo alfa-beta, o Lc0 combina redes neurais profundas com a pesquisa de

Monte Carlo Tree Search (MCTS). A ideia por trás de seu desenvolvimento foi criar um agente que pudesse aprender a jogar xadrez sozinha, sem a necessidade de qualquer conhecimento humano pré-programado sobre táticas, nenhum conhecimento intrínsico sobre aberturas ou finais. Assim como o AlphaZero, a Leela Zero é treinada através de autotreinamento, onde joga milhões de partidas contra si mesma e ajusta sua rede neural para melhorar gradualmente.

O projeto Leela Chess Zero foi lançado pela comunidade de xadrez como uma resposta à ausência de código público do AlphaZero. Para treinar a rede neural da Leela Zero, os desenvolvedores utilizam uma infraestrutura distribuída, onde milhares de voluntários contribuem com o poder computacional de seus próprios computadores. Essas contribuições permitem que a Leela jogue um número imenso de partidas e melhore seu desempenho ao longo do tempo. O uso de GPUs (Unidades de Processamento Gráfico) é essencial para o treinamento eficaz da rede neural, permitindo que o agente se concentre nas posições mais promissoras em vez de calcular milhões de posições como os motores tradicionais (SADMINE; HUSNA; MüLLER, 2024).

Desde o seu lançamento, Leela Chess Zero tem competido de igual para igual com os melhores motores de xadrez, incluindo o Stockfish, em competições como o TCEC (Top Chess Engine Championship). Embora tenha começado mais fraca, a Leela Zero melhorou drasticamente à medida que mais dados de treinamento foram gerados, demonstrando força competitiva suficiente para enfrentar os motores mais poderosos. Além disso, o estilo de jogo da Leela é frequentemente comparado ao do AlphaZero, com movimentos criativos e sacrifícios estratégicos que desafiam as expectativas dos especialistas em xadrez. O projeto continua a evoluir, sendo um dos principais exemplos do uso de inteligência artificial e redes neurais no xadrez (SADMINE; HUSNA; MüLLER, 2024).

2.1.5 Komodo

Komodo é um agente de xadrez altamente respeitado, conhecido por sua precisão e estilo de jogo posicional. Criado por Don Dailey e Mark Lefler, o agente passou por diversos estágios de desenvolvimento, combinando técnicas tradicionais de xadrez com algoritmos avançados para análise de posições. Inicialmente, o foco de Komodo estava em uma abordagem mais estratégica, com uma avaliação detalhada das nuances posicionais do tabuleiro, o que a diferenciava de outros agentes que priorizavam táticas de cálculo de força bruta. Com o passar do tempo, o agente evoluiu para equilibrar táticas e estratégias,

tornando-se uma das principais concorrentes no cenário de xadrez computacional.

Um dos fatores de destaque da Komodo é sua capacidade de personalizar o estilo de jogo, permitindo que os usuários ajustem parâmetros para simular preferências mais agressivas ou defensivas. Esse nível de controle tornou o agente popular não apenas entre jogadores que buscam desafios, mas também para análises profundas de partidas e preparação para torneios. Além disso, Komodo tem se mantido relevante com constantes atualizações e melhorias, com sua performance altamente afinada nas versões mais recentes, o que lhe permitiu competir com motores de xadrez como Stockfish e Leela Chess Zero.

Komodo também se destacou em competições de agentes de xadrez. O agente conquistou vitórias em edições do Top Chess Engine Championship (TCEC) e em diversos outros torneios de xadrez computacional. O agente foi projetada para ser eficaz em várias plataformas, utilizando tanto a CPU quanto a GPU de forma eficiente para otimizar sua capacidade de análise, o que contribuiu para seu sucesso em competições de alto nível e no uso prático por jogadores.

A partir de 2019, Komodo introduziu uma versão chamada Komodo MCTS, que combinou seu já robusto sistema de avaliação com a pesquisa de Monte Carlo (Monte Carlo Tree Search). Essa combinação permitiu ao agente explorar melhor certas variações e aumentar sua eficácia em jogos com regras variantes, como o Chess960. O uso de MCTS destacou-se pela capacidade de avaliar menos posições, mas com maior profundidade, um avanço significativo em relação às abordagens mais tradicionais de agentes (CONTRIBUTORS, 2024f).

2.1.6 Houdini

Houdini é um dos agentes de xadrez mais conhecidos por seu estilo de jogo agressivo e sua capacidade tática de alta precisão. Desenvolvido por Robert Houdart, o agente ganhou popularidade rapidamente logo após sua primeira versão em 2010, quando se destacou por superar a maioria dos agentes contemporâneas, como Rybka e as primeiras versões de Stockfish. Houdini era particularmente eficiente em explorar combinações táticas complexas e calcular variações que muitos outros motores de xadrez não detectavam, tornando-se uma escolha popular para jogadores e analistas.

O sucesso de Houdini está relacionado à sua abordagem baseada na pesquisa Alpha-Beta, que permitiu que o agente realizasse avaliações precisas ao reduzir drastica-

mente o número de posições a serem examinadas. Isso permitiu que Houdini se tornasse uma referência no que diz respeito ao cálculo de lances forçados e na identificação de táticas ocultas. Durante o auge de sua popularidade, Houdini liderou a lista de classificações de agentes e era frequentemente utilizada em competições de xadrez computacional, incluindo edições do TCEC.

No entanto, ao longo dos anos, Houdini começou a perder sua posição dominante com o surgimento de agentes baseados em redes neurais, como Leela Chess Zero, e o avanço de Stockfish, que adotou métodos mais modernos. O agente continuou relevante para jogadores e analistas até o final da década de 2010, mas, sem atualizações regulares ou inovações significativas em comparação com seus competidores, Houdini acabou ficando para trás. Mesmo assim, sua contribuição para o mundo dos agentes de xadrez foi significativa, especialmente em relação ao desenvolvimento de agentes focados em táticas detalhadas e cálculo preciso (CONTRIBUTORS, 2024e).

3 INFRAESTRUTURA PARA CONSTRUÇÃO DE UM AGENTE DE XADREZ

Para desenvolver um Agente de xadrez, é recomendável uma infraestrutura otimizada para utilizar o poder de processamento disponível. Inicialmente, uma CPU com múltiplos núcleos é essencial para calcular movimentos em paralelo e realizar buscas profundas no espaço de possíveis jogadas. Muitos agentes modernos, como a Komodo e o Stockfish, também se beneficiam do uso de GPUs, que são altamente eficientes para cálculos massivamente paralelos, especialmente em abordagens como redes neurais ou Monte Carlo Tree Search (MCTS).

Uma linguagem de programação eficiente como C++ ou orientada a objetos como Java e Python, deve ser utilizada para a contrução do algoritmo central para a tomada de decisões. Esse algoritmo costuma ser o minimax, aprimorado pela poda alfa-beta para reduzir o número de posições analisadas. Por fim, uma interface de comunicação como o Protocolo Universal de Xadrez (UCI) permite a integração com interfaces gráficas e ferramentas de análise.

3.1 Notações de Xadrez

O tabuleiro de xadrez é composto por uma matriz de 8×8 casas, totalizando 64 quadrados que alternam entre cores claras e escuras. Cada casa do tabuleiro é identificada por uma notação que permite registrar e descrever movimentos e posições de peças de forma concisa. A notação padrão usa uma combinação de letras e números: as colunas são identificadas por letras de “a” a “h” (da esquerda para a direita do jogador com as peças brancas), e as fileiras são numeradas de “1” a “8” (de baixo para cima no ponto de vista do jogador com as peças brancas). Dessa forma, a casa no canto inferior esquerdo é denominada `a1`, enquanto a casa no canto superior direito é denominada `h8`.

3.1.1 Notação Algébrica

A notação algébrica é a forma mais comum de registrar movimentos no xadrez e é amplamente usada tanto por jogadores quanto por programas de computador e agentes de xadrez. Esse sistema é simples e eficiente, indicando apenas a casa de origem e a casa de destino de uma peça, além de detalhes adicionais para jogadas especiais. Cada peça é

indicada por uma letra maiúscula específica (exceto o peão), e o movimento é registrado pela notação de sua posição de saída e chegada.

3.1.1.1 Descrição de Movimentos

No sistema de notação algébrica, cada peça é representada por uma letra: rei (`K`), dama (`Q`), torre (`R`), bispo (`B`), cavalo (`N`), e o peão, que é representado implicitamente (sem uma letra). Para descrever um movimento, utiliza-se a letra da peça e a casa para onde ela se move. Por exemplo, se o cavalo se move para `f3`, o movimento é registrado como `Nf3`. Movimentos de captura são indicados por um `x` entre a letra da peça e a casa de destino, como em `Bxf3`, que indica que um bispo capturou uma peça na casa `f3`.

3.1.1.2 Jogadas Especiais

Algumas jogadas especiais são registradas com notações específicas:

- **Roque**: O roque é indicado como `O-O` para o roque curto e `O-O-O` para o roque longo.
- **Promoção de Peão**: Quando um peão atinge a última fileira e é promovido, o movimento é registrado com a notação da casa de promoção e a peça escolhida, como em `e8=Q`.
- **Xeque e Xeque-mate**: Um movimento que coloca o rei adversário em xeque é seguido pelo símbolo +, e o xeque-mate é indicado por #.
- **Captura En Passant**: Quando um peão faz um movimento de duas casas e passa ao lado de um peão adversário, este pode capturá-lo como se ele tivesse se movido apenas uma casa. A captura é indicada pela notação da casa onde o peão que capturou estava e a casa do peão capturado, seguida de `x`. Por exemplo, se o peão branco avança de `e2` para `e4` e o peão preto está em `d4`, a captura en passant é registrada como `exd3`.

3.1.2 Notação Origem-Destino

A notação de movimento no formato de **origem-destino**, como `a1b2`, é uma forma simples de descrever jogadas no xadrez. Nessa notação, o movimento é descrito pela posição inicial e a posição final de uma peça. Aqui, a letra representa a coluna do

tabuleiro (de 'a' a 'h'), e o número representa a linha (de '1' a '8'). O exemplo `a1b2` significa que uma peça se move da casa `a1` para a casa `b2`. Repare que essa notação não descreve o tabuleiro e sim os movimentos. Portanto, para inferir a posição do tabuleiro, assim como na notação algébrica, é necessário ler todas as jogadas a partir do tabuleiro inicial.

3.1.2.1 Características da Notação Origem-Destino

Esse formato é mais direto e simples, mas não leva em consideração algumas nuances importantes do jogo, como capturas ou outras jogadas especiais. Para indicá-las, são usados símbolos adicionais, como:

- **Captura**: Quando uma peça captura outra, é comum inserir um `x` entre a origem e o destino, como em `a1xa2`, indicando que a peça na casa `a1` capturou a peça na casa `a2`.
- **Roque**: Quando se realiza um roque, a notação usa uma convenção especial, como `O-O` para o roque curto e `O-O-O` para o roque longo.
- **Promoção de Peão**: Quando um peão atinge a última linha e é promovido, a notação inclui a promoção, como `a8=Q`, indicando que o peão na casa `a8` foi promovido a uma dama.

3.1.2.2 Exemplos

- `e2e4`: O peão se move de `e2` para `e4`.
- `Ng1f3`: O cavalo se move de `g1` para `f3`.
- `a7xa6`: O peão na casa `a7` captura a peça na casa `a6`.

Essa notação é bastante popular em alguns contextos de jogos informais ou entre jogadores iniciantes, pois é direta e fácil de entender, mas carece da riqueza de detalhes de outras notações como a **algébrica** ou **PGN**, que fornecem informações adicionais sobre a partida, como o tipo de captura ou xeque.

3.1.2.3 Comparação com Outras Notações

Em contraste com a notação algébrica, que exige o uso de coordenadas mais precisas (e.g., incluindo xeque, promoção e roque), a notação de origem-destino mantém-se mais simples e concisa. Porém, pode ser difícil de interpretar corretamente sem o con-

texto, principalmente em jogos complexos, onde a informação adicional é fundamental para o entendimento da partida.

3.1.3 Notação FEN (Forsyth-Edwards Notation)

A notação FEN (Forsyth-Edwards Notation) é usada para registrar o estado completo de um tabuleiro de xadrez em um determinado momento. Ela descreve a posição das peças, o lado que tem o direito de jogar, a possibilidade de roque, a regra da captura en passant, e o contador de movimentos, permitindo que qualquer posição de jogo seja reconstituída com precisão. A FEN é amplamente utilizada para armazenar posições em bases de dados e para trocar informações entre agentes.

3.1.3.1 Formato da Notação FEN

A notação FEN consiste em seis campos, separados por espaços:

1. **Posição das peças**: Cada fileira do tabuleiro é descrita da fileira 8 até a fileira 1, usando letras minúsculas para peças pretas (`p` para peão, `k` para rei, etc.) e letras maiúsculas para peças brancas. Casas vazias são representadas por números de 1 a 8.
2. **Lado a jogar**: Um único caractere, `w` para as brancas e `b` para as pretas, indicando quem deve fazer o próximo movimento.
3. **Possibilidade de Roque**: Representado por qualquer combinação dos caracteres `K`, `Q`, `k`, `q` (para roque curto e longo, das brancas e das pretas), ou um − se nenhum roque for permitido.
4. **Captura en passant**: Caso uma captura en passant esteja disponível, a coluna correspondente é indicada (`e3`, por exemplo); caso contrário, usa-se −.
5. **Meio-movimentos**: Um número que representa a contagem de meio-movimentos desde a última captura ou movimento de peão.
6. **Contador de Movimentos**: Um número que conta o número total de movimentos realizados, incrementado a cada vez que as pretas jogam.

Exemplo da notação FEN para a posição inicial do tabuleiro:

```
rnbqkbnr/pppppppp/8/8/8/8/PPPPPPPP/RNBQKBNR w KQkq - 0 1
```

Onde:

`rnbqkbnr/pppppppp/8/8/8/8/PPPPPPPP/RNBQKBNR`: Representa as peças no tabuleiro, com as letras minúsculas para peças negras e as maiúsculas para peças brancas. Os números indicam o número de casas vazias em uma linha.

`w`: Indica que é a vez das brancas jogarem.

`KQkq`: Indica que ambas as cores têm direito ao roque (K e Q para as brancas, k e q para as negras).

`-`: Não há en passant.

`0`: Número de plies desde o último avanço de peão ou captura (importante para a regra de 50 movimentos).

`1`: Número do movimento.

3.1.4 Notação PGN (Portable Game Notation)

A notação PGN é um formato textual que armazena partidas de xadrez completas, incluindo os movimentos, a data, o local e os jogadores envolvidos. Usando a notação algébrica para os movimentos, a PGN organiza essas informações em um formato de fácil leitura, o que torna essa notação muito usada para análises, registros e compartilhamento de partidas.

3.1.4.1 Estrutura da Notação PGN

Um arquivo PGN começa com cabeçalhos delimitados por colchetes, que contêm informações contextuais, como o nome dos jogadores, a data e o local do jogo. Após os cabeçalhos, os movimentos são listados sequencialmente, onde cada movimento é numerado e registrado em notação algébrica. Ao final, o resultado da partida é indicado por `1-0` (vitória das brancas), `0-1` (vitória das pretas) ou `1/2-1/2` (empate).

Exemplo da notação PGN para a posição inicial do tabuleiro:

```
[Event "Friendly Game"]
[Site "Online"]
[Date "2024.11.07"]
[Round "?"]
[White "Player 1"]
[Black "Player 2"]
[Result "*"]

1.  e4 e5 2.  Nf3 Nc6 3.  Bb5 a6 4.  Ba4 Nf6 5.  O-O
Be7
```

Onde:

As linhas entre colchetes (`[ ]`) contêm metadados sobre a partida, como o evento, a data e os jogadores.

A sequência de jogadas (por exemplo, `1. e4 e5 2. Nf3 Nc6...`) segue o formato de notação algébrica para os movimentos.

A partida ainda não tem um vencedor, então o resultado é $\star$ (indica que a partida não foi decidida).

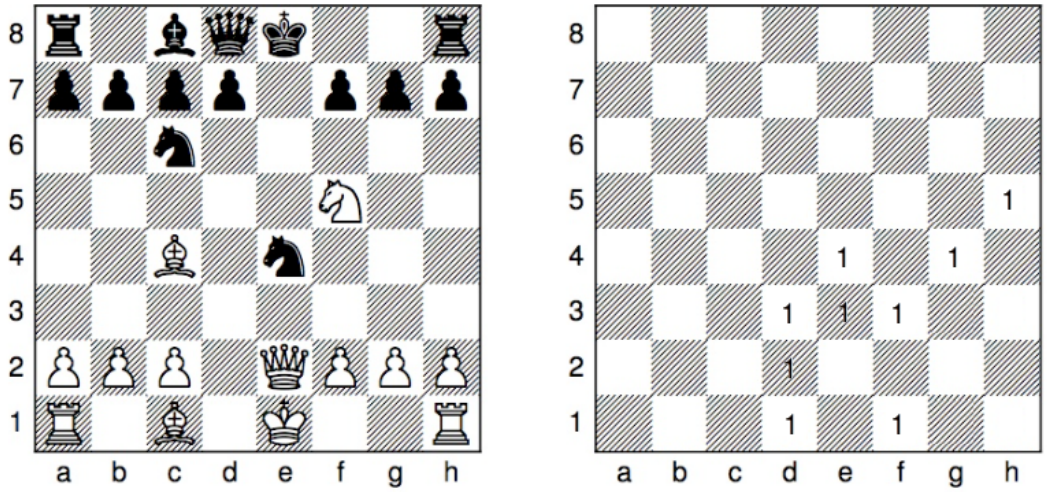
3.2 Bitboards

Bitboards são estruturas de dados que utilizam um número binário de 64 bits para representar um estado do tabuleiro de xadrez. Cada bit corresponde a uma casa no tabuleiro (um total de 64 casas em um tabuleiro padrão 8x8). Um bit "1"indica a presença de uma peça em uma casa específica, enquanto um bit "0"indica que a casa está vazia. Dessa forma, é possível representar a posição de cada peça de um jogo de xadrez com uma série de bitboards — um para cada tipo de peça (peões, torres, bispos, etc.) ou para cada cor (branco e preto). A principal vantagem dos bitboards está na sua eficiência computacional. Essas operações aritméticas são otimizadas pelo fato de que bitboards em linguagens como Java e C++ podem ser representados por tipos de dados como long e ulong, que têm 64 bits, exatamente o número de casas no tabuleiro de xadrez. Operações bitwise (operações lógicas como AND, OR, XOR) podem ser usadas para realizar cálculos extremamente rápidos, como verificar movimentos legais, ameaças, ataques e defesas de peças. Essas operações podem ser feitas pela JVM diretamente no nível do hardware do processador, tornando-as muito rápidas em comparação com abordagens que manipulam diretamente objetos ou estruturas de dados mais complexas. Por exemplo, com uma única operação de deslocamento de bits, pode-se calcular o movimento possível de um bispo ou de uma torre. Isso possibilita que um único registro de bitboard ocupe a menor quantidade de memória possível, enquanto as operações de manipulação de bits são realizadas de maneira extremamente rápida devido à correspondência direta com o hardware da CPU.

A figura 3.1 demonstra o uso de um bitboard para determinar os movimentos possíveis da peça rainha de cor branca.

A ideia dos bitboards foi inicialmente inspirada por representações computacionais otimizadas de problemas lógicos e de jogos em redes de computadores. Com o passar do tempo, a técnica foi adaptada para agentes de xadrez devido à sua eficiência em

Figura 3.1: Bitboard



Representação bitboard das casas disponíveis para peça Rainha da cor Branca Fonte: Adaptação de (TANNOUS, 2007)

permitir uma análise mais profunda das posições do tabuleiro em menos tempo.

A popularidade dos bitboards aumentou conforme os agentes começaram a competir em campeonatos mundiais de xadrez por computador, onde a eficiência de busca e a velocidade computacional são fatores críticos. Hoje, agentes como Stockfish, Komodo e Houdini usam bitboards como um de seus principais componentes para calcular milhões de posições por segundo.

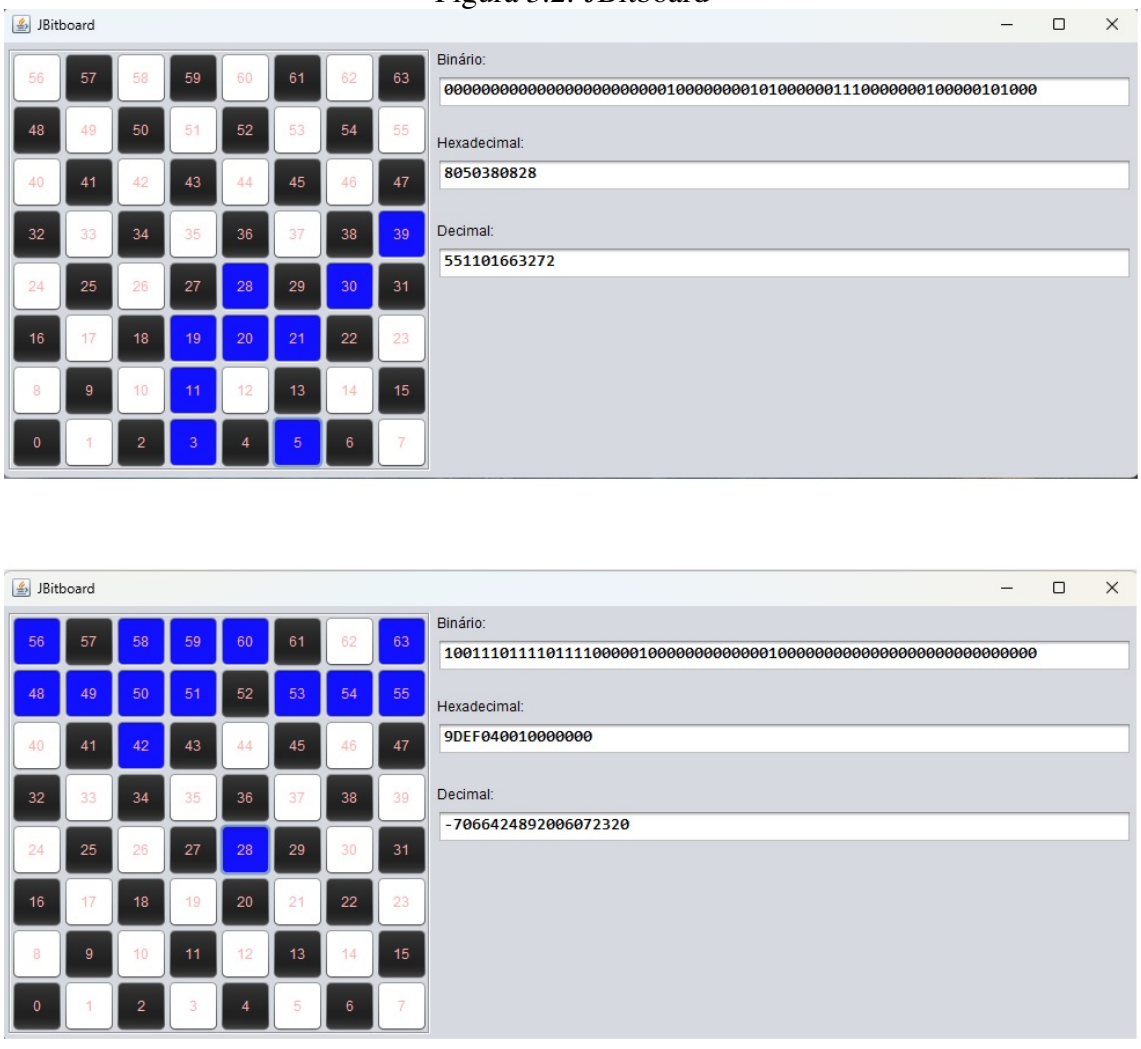
A utilização de bitboards facilita a implementação de heurísticas avançadas e de algoritmos de busca, como o algoritmo de poda alfa-beta, já que permite a verificação eficiente de possíveis movimentos e captura de peças. Muitos dos cálculos complexos envolvidos no processo de avaliação e geração de movimentos podem ser otimizados através de manipulações bitwise, o que é crucial para o desempenho de um agente de xadrez.

Na prática, os bitboards são utilizados de forma modular, com diferentes bitboards representando diferentes aspectos do jogo. Por exemplo, um bitboard pode representar a posição de todos os peões brancos no tabuleiro, enquanto outro bitboard representa as torres pretas. Em agentes modernos, como Stockfish, cada peça pode ter seu próprio bitboard, permitindo uma forma de verificar rapidamente as posições de qualquer peça específica. Para calcular os movimentos, os bitboards são manipulados usando operações de deslocamento (shifts), que deslocam os bits para a esquerda ou direita, simulando o movimento das peças no tabuleiro. Por exemplo, o movimento de um cavalo pode ser calculado com uma combinação de deslocamentos e operações lógicas para verificar quais

casas estão acessíveis e se estão ocupadas por peças inimigas ou aliadas.

O programa JBitboard demonstra um exemplo prático da utilização de bitboards do espaço de ataque da rainha branca da figura 3.1 em conjunto com o bitboard das peças pretas. Esses dados podem ser visualizados em duas instâncias do programa auxiliar JBitboard, figura 3.2. O programa oferece diferentes formas de visualização, incluindo representações gráficas, binárias, decimais e hexadecimais de cada bitboard, permitindo uma análise detalhada das informações.

Figura 3.2: JBitboard



Representação bitboard do espaço disponíveis para peça Rainha da cor Branca e das peças pretas
Fonte: Autoria própria

3.2.1 Aritmética

3.2.1.1 Operações Lógicas

Operações como AND, OR, XOR, deslocamento à esquerda e à direita são fundamentais para realizar cálculos de movimentos e ataques de maneira extremamente rápida. Essas operações são realizadas diretamente no nível do hardware, o que torna os bitboards uma ferramenta de alta performance.

Por exemplo, a operação de AND é utilizada para verificar interseções entre bitboards. Em um cenário de captura, o bitboard que representa o ataque de uma peça pode ser comparado com o bitboard que representa as peças adversárias. Se houver uma sobreposição de bits, isso indica que uma captura é possível naquela posição. Similarmente, a operação de OR permite combinar dois bitboards para gerar um conjunto que contém todos os bits ativos de ambos os bitboards. Essa operação é útil para representar áreas do tabuleiro que estão sob ataque ou ocupadas por peças de ambos os lados (BROWNE, 2014).

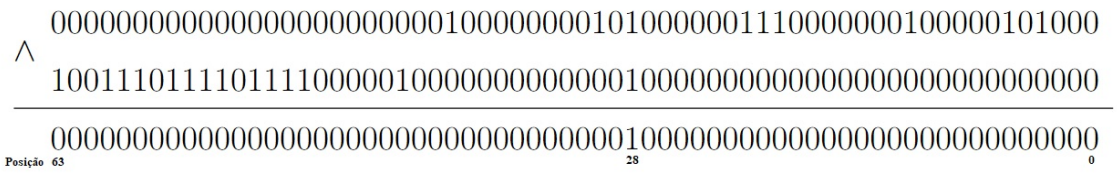
Outra operação frequentemente usada é o XOR, que permite alternar os estados dos bits. No contexto de uma troca de peças, pode-se usar o XOR para "alternar"a presença de uma peça de um jogador para outro, retirando-a do bitboard de um jogador e adicionando-a ao bitboard do oponente. Isso é eficiente porque o XOR consegue realizar ambas as tarefas em uma única operação, sem a necessidade de realizar modificações em múltiplas etapas (BROWNE, 2014).

A operação de complemento bit a bit, também conhecida como operação NOT, é amplamente utilizada na manipulação de bitboards para xadrez. Quando aplicada a um bitboard, essa operação inverte todos os bits: os que estão em 1 passam para 0, e os que estão em 0 passam para 1. Isso é particularmente útil ao se trabalhar com máscaras de exclusão. Por exemplo, ao querer definir todas as posições que não estão ocupadas por peças de um jogador, basta aplicar a operação ao bitboard de suas peças. Em termos práticos, o complemento de um bitboard pode ser usado para identificar as casas livres no tabuleiro ou para delimitar áreas que estão sob controle do adversário. Essa abordagem oferece uma maneira eficiente de gerar resultados sem a necessidade de percorrer cada bit individualmente.

Um exemplo de operação de bitboard é ilustrado na figura 3.3 efetuando uma opeação AND entre os bitboards da figura 3.2.

O resultado dessa operação pode se interpretado como as posições com peças

Figura 3.3: Opeação entre Bitboards



Representação binária da operação AND entre dois bitboards resultando em um terceiro Fonte: Autoria própria

pretas que estão sofrendo ataque da rainha branca. No exemplo, referente a figura 3.1, a posição 28 corresponde ao cavalo na posição *e4* do tabuleiro.

3.2.1.2 Operações de Deslocamento

As operações de deslocamento de bits (shift) são particularmente úteis no cálculo de movimentos em linha reta, como os de torres e bispos. Ao deslocar os bits de um bitboard para a esquerda ou direita, é possível calcular rapidamente o alcance de uma peça em uma direção específica. Por exemplo, ao deslocar um bitboard de peões para a esquerda ou direita, é possível determinar quais peões estão aptos para captura en passant ou ataques diagonais.

3.2.1.3 Exemplo de implementação com bitboards

A Figura 3.4 demonstra como as operações de bitboard podem ser implementadas na linguagem Java para determinar o avanço de um peão no tabuleiro.

- Linha 3: Definido atributo público *blackPawns* do tipo long (64 bits) como bitboard para representar os peões de cor preta.
- Linha 5: Assinatura do método *moveBlackPawnsForward* que executa o movimento do peão e o tipo de movimento especificado nos parâmetros.
- Linha 7: Determinação da máscara da posição com a constante *1* do tipo long deslocado para esquerda até a posição indicada pelo parâmetro *pawnPosition*.
- Linha 9: Operação OR de bitboard aplicado entre o atriuto *blackPawn* com a máscara deslocada para a direita 16 vezes no caso de movimento duplo na primeira jogada do peão ou 8 vezes no caso de mnovimento simples. O tipo de movimento é indicado pelo parâmetro booleano *doubleMove*. Essa operação OR (do tipo União) fará com que a posição destino fique com um peão mas não removerá o peão da

Figura 3.4: Exemplo de operações com bitboards na Linguagem Java

```java
public class Bitboard
{
    public long blackPawns = 0x00FF000000000000L;

    public void moveBlackPawnsForward(int pawnPosition, boolean doubleMove)
    {
      long mask = 1l << pawnPosition;

      blackPawns |= mask >>> (doubleMove ? 16 : 8);
      blackPawns &= ~mask;
    }

    //TESTE
    public static void main(String[] args)
    {
        Bitboard bitboard = new Bitboard();
        bitboard.moveBlackPawnsForward(52, true);
        System.out.println(String.format("%64s",
            Long.toBinaryString(bitboard.blackPawns)).replace(' ', '0'));

    }

}
```

Exemplo de um método Java que calcula o movimento de um peão no tabuleio. Fonte: Autoria própria

casa de origem, sendo necessário removê-lo.

- Linha 10: Por fim o peão da casa origem é removido com a execução da operação AND (do tipo Disjunção) entre o bitboard blackPawns com a máscara negada.
- Linha 16 a 20: Teste do método moveBlackPawnsForward. Um objeto do tipo bitboard é instanciado e o método é moveBlackPawnsForward é invocado com os parâmetros 52 e true. O resultado é a impressão da representação binária do bitboard *blackPawns*

3.2.2 Bitboards Mágicos

Os bitboards mágicos são uma técnica avançada usada em agentes de xadrez para otimizar a geração de movimentos, especialmente para peças como torres e bispos, que possuem padrões de ataque direcional em linhas ou diagonais. Essa técnica permite transformar um problema de busca de ataques (movimentos válidos) de uma peça em um índice para uma tabela precomputada, o que acelera significativamente o cálculo de possíveis movimentos em um tabuleiro de xadrez. Em essência, a magia dos bitboards reside no uso de um valor mágico específico, calculado de forma cuidadosa, para compactar a informação de ocupação de casas em um tabuleiro e gerar os movimentos correspondentes

com alta eficiência.

A relevância dos bitboards mágicos está em sua capacidade de lidar eficientemente com as complexas interações das peças de xadrez em um tabuleiro 8x8, reduzindo drasticamente o tempo de computação. Sem essa técnica, agentes de xadrez que calculam movimentos com base em busca direta e cálculo de ocupação demorariam muito mais para explorar todas as possibilidades de jogadas. A eficiência que os bitboards mágicos trazem é crucial para agentes modernos que precisam avaliar milhões de posições em questão de segundos, como Stockfish e Komodo. Essas otimizações permitem que os motores de xadrez aumentem sua profundidade de busca, resultando em jogadas mais precisas e avaliações mais fortes (KANNAN, 2007).

Os bitboards mágicos são usados principalmente para calcular os ataques de torres e bispos. Essas peças atacam em linhas e diagonais, onde a presença de outras peças pode bloquear seus ataques. O uso de bitboards mágicos permite que o agente calcule, de forma eficiente, todas as casas que uma torre ou bispo pode atacar, dado um arranjo específico de peças que bloqueiam seu caminho. Ao invés de realizar uma busca complexa a cada movimento, a técnica dos bitboards mágicos transforma o problema de encontrar casas de ataque em um simples problema de acesso a uma tabela, usando um índice gerado a partir de operações com o bitboard de ocupação e um valor mágico específico. Isso economiza tempo ao evitar cálculos repetitivos.

Para utilizar bitboards mágicos, é necessário primeiro calcular um valor mágico para cada peça e posição. Esse valor é escolhido com base em heurísticas que garantem que ele, quando combinado com o bitboard de ocupação, gere um índice único que pode ser usado para acessar uma tabela de movimentos precomputada. Em termos práticos, essa tabela contém todas as possíveis casas que uma torre ou bispo pode atacar a partir de uma posição específica, dado um arranjo de bloqueios. Um exemplo de como isso funciona:

1. Um bispo está na casa *c1* (linha 1, coluna 3).
2. Usamos o bitboard de ocupação para verificar quais casas estão bloqueadas em suas diagonais.
3. Esse bitboard de ocupação é combinado com um valor mágico, que gera um índice para a tabela de movimentos do bispo.
4. A tabela retorna todas as casas que o bispo pode atacar, levando em conta as obstruções.

Essa abordagem elimina a necessidade de recalcular continuamente os ataques, permitindo que o movimento da peça seja obtido em tempo constante *O(1)*.

Outras operações de bitboards estão descritas na seção C.2.

3.3 Tabela de Transposição

A Tabela de Transposição (Transposition Table) é uma estrutura de dados fundamental em agentes de xadrez modernas, projetada para otimizar o desempenho de algoritmos de busca como o Minimax e suas variantes, como o Alpha-Beta Pruning. Ela serve para armazenar posições já analisadas durante a busca por jogadas ótimas, permitindo a reutilização de resultados previamente calculados para economizar tempo e evitar a repetição de cálculos.

A tabela de transposição é uma grande hash table que armazena informações sobre posições específicas do jogo de xadrez. Uma posição de xadrez é descrita pelo arranjo das peças no tabuleiro, o lado a jogar, possíveis roques, en passant, e outras informações contextuais. Como a busca de jogadas em agentes de xadrez frequentemente revisita posições já analisadas (especialmente em jogos com movimentos semelhantes em diferentes ordens, chamados de transposições), a tabela de transposição permite a reutilização dessas análises, evitando a reavaliação da mesma posição (MARSLAND, 1991).

Cada posição é identificada por uma chave de hash única, gerada usando um algoritmo de hashing Zobrist, que é rápido e eficiente. A partir dessa chave, a posição é armazenada na tabela com informações associadas, como o valor da avaliação daquela posição e a profundidade em que foi calculada (ZOBRIST, 1990).

3.3.1 Objetivos

1. Evitar a recalculação de posições: Durante a busca, muitas posições podem ser alcançadas por diferentes sequências de jogadas, ou seja, transposições. Sem uma tabela de transposição, cada vez que o agente revisita uma posição, ela recalcula seu valor. Com a tabela, o motor verifica se essa posição já foi calculada e simplesmente reutiliza o valor armazenado, economizando tempo de processamento (MARSLAND, 1991).
2. Alfa-Beta Pruning mais eficiente: A tabela de transposição também melhora a efici-

ência da poda alfa-beta. Quando uma posição já foi analisada, seu valor armazenado pode ser usado para ajustar os limites alfa e beta, potencialmente reduzindo a quantidade de posições que precisam ser avaliadas na árvore de busca (MARSLAND, 1991).

3.3.2 Implementação

A implementação Básica da Tabela de Transposição em um Agente de Xadrez compreende o calculo do hashing de posição e o armazenamento.

A posição é convertida em uma chave de hash única, geralmente utilizando Zobrist Hashing. Cada peça e cada quadrado do tabuleiro contribuem com um valor de hash gerado por uma função pseudoaleatória.

Quanto ao armazenamento de dados, para cada posição, são armazenados os seguintes dados:

- Valor da posição: O valor de avaliação obtido pela função de avaliação do agente.
- Tipo de Nó: Indica se o valor é exato (EXACT), ou se é um limite inferior (ALPHA) ou superior (BETA), dependendo de como foi obtido durante a poda alfa-beta.
- Profundidade: A profundidade da busca na qual o valor foi calculado, usada para determinar se a informação é suficientemente precisa para a busca atual.
- Melhor Movimento: O melhor movimento identificado naquela posição, que pode ser usado para guiar a busca de forma mais eficiente (heurística "move ordering").

3.3.3 Reuso da Informação

Quando uma posição é alcançada novamente durante a busca, o agente verifica a tabela de transposição. Se a posição já está armazenada, o agente pode utilizar o valor armazenado diretamente, economizando a reavaliação da posição. Caso a profundidade da busca atual seja maior, o valor armazenado pode ser refinado, recalculando com uma profundidade maior e substituindo o valor existente (CAMPBELL; HOANE; HSU, 2002).

3.3.4 Impacto na Performance

O impacto da tabela de transposição na performance dos agentes de xadrez é enorme. As buscas são aceleradas com o uso de uma tabela de transposição pois ocorre a redução drastica do número de nós avaliados durante a busca, pois evita a análise repetida de posições. Isso é especialmente útil em posições com muitas transposições. Também ocorre uma melhor ordenação de movimentos. Armazenar o melhor movimento para uma posição permite que o agente tente primeiro esse movimento em buscas subsequentes, melhorando a eficiência do algoritmo de poda alfa-beta. Quando os movimentos melhores são ordenados primeiro ocorrem mais oportunidades de poda. A tabela de transposição permite uma redução de nós visitados. Estudos mostram que a tabela de transposição pode reduzir o número de nós visitados em até 50% ou mais, dependendo da profundidade da busca e da complexidade da posição. Em um agente sem uma tabela de transposição, uma busca a uma profundidade de 8 pode exigir a avaliação de milhões de posições. Com uma tabela de transposição, esse número pode ser reduzido para centenas de milhares, ou até menos, dependendo da eficiência da tabela e da quantidade de transposições (CAMPBELL; HOANE; HSU, 2002).

3.3.5 Limitações

A tabela de transposição é limitada pelo tamanho da memória disponível. Em partidas muito longas ou quando a profundidade da busca aumenta, a tabela pode começar a descartar posições antigas para armazenar novas, o que pode diminuir sua eficiência. No entanto, algoritmos de substituição inteligentes, como "Least Recently Used"(LRU), são usados para mitigar esse problema (CAMPBELL; HOANE; HSU, 2002).

3.4 Gerador de Movimento

A geração de movimentos é uma das etapas mais críticas no desenvolvimento de agentes de xadrez. Esse processo envolve listar todos os movimentos legais possíveis para uma determinada posição no tabuleiro e classificá-los com base em sua potencial eficácia. Em agentes de xadrez modernos a geração de movimentos desempenha um papel fundamental na eficiência do algoritmo de busca, como o minimax com poda alfa-beta.

Neste capítulo, será explorado o funcionamento da geração de movimentos, abordando diferentes técnicas, estágios de geração e heurísticas utilizadas por agentes.

3.4.1 O Processo de Geração de Movimentos

O processo de geração de movimentos envolve a criação de uma lista de todos os movimentos pseudo-legais, ou seja, movimentos que não necessariamente levam em conta a condição de xeque, mas que são válidos em termos de regras de movimentação das peças. O agente, em seguida, filtra os movimentos ilegais (aqueles que deixam o rei em xeque) e atribui uma pontuação a cada um, dependendo de sua qualidade. Essa pontuação é usada para classificar os movimentos e, potencialmente, escolher os mais promissores para exploração durante a busca. Como discutido na Seção C, os movimentos são gerados em estágios específicos.

3.4.2 Classificação e Seleção de Movimentos

Além da geração de movimentos, o agente precisa classificá-los eficientemente para guiar a pesquisa de maneira produtiva. Stockfish faz isso utilizando várias heurísticas, como a "Most Valuable Victim/Least Valuable Attacker"(MVV/LVA), que dá prioridade a capturas de peças valiosas por atacantes de baixo valor, e o "history heuristic", que atribui uma pontuação elevada a movimentos que foram eficazes em posições semelhantes no passado. Essas classificações são essenciais para que o algoritmo minimax, com poda alfa-beta, explore primeiro os movimentos mais promissores, reduzindo o espaço de busca (Marsland, 1987).

3.4.3 Ordenação dos Movimentos

Durante a fase de geração de movimentos, o agente também realiza uma "partial insertion sort"(algoritmo de ordenação parcial), ordenando os movimentos com base em suas pontuações pré-calculadas. Movimentos com pontuações altas são investigados primeiro, permitindo que a poda alfa-beta elimine ramos desnecessários da árvore de busca mais rapidamente. Essa técnica reduz significativamente o número de nós avaliados pelo algoritmo, aumentando a eficiência do agente (Donninger et al., 1993).

3.4.4 Busca Quiescente e Geração de Movimentos Especiais

A busca quiescente é uma técnica usada para lidar com a "explosão de movimentos"em posições altamente voláteis, como aquelas com muitas capturas possíveis ou ameaças de xeque. O agente interrompe a pesquisa normal e foca apenas em capturas, cheques e movimentos que estabilizem a posição, buscando evitar erros de avaliação causados por essas situações dinâmicas. Nos estágios QCAPTURE e QCHECK, Stockfish gera movimentos com base nesses critérios, limitando a pesquisa a movimentos que poderiam alterar significativamente a avaliação de uma posição.

A geração de movimentos também pode ser combinada com a a tabela de transposição (TT), descrita anteriormente, que armazena avaliações de posições já analisadas. O agente reutiliza esses resultados sempre que encontra a mesma posição novamente, o que evita a repetição de cálculos desnecessários. Os estágios MAIN-TT, EVASION-TT, QSEARCH-TT e PROBCUT-TT são responsáveis por verificar a TT e retornar rapidamente a melhor jogada, se já estiver disponível.

3.4.5 Aplicação de Heurísticas no Contexto de Evasão e Capturas

Em situações críticas, como quando o rei está sob ataque, o agente gere movimentos de evasão, priorizando aqueles que removem a ameaça de xeque. Nos estágios EVASION-INIT e EVASION, movimentos de captura e bloqueio são gerados e classificados, utilizando a "Static Exchange Evaluation"(SEE), uma técnica que calcula o balanço material das capturas para determinar se uma troca será vantajosa ou desvantajosa. Esse cálculo permite que o agente descarte rapidamente capturas ruins, que poderiam resultar em desvantagem material (Buro, 1995).

As técnicas de geração de movimentos evoluíram ao longo do tempo, permitindo que os agentes modernos explorem milhões de nós por segundo, escolhendo os movimentos mais promissores em uma fração de segundo. A geração eficiente de movimentos, aliada a heurísticas de classificação e à poda alfa-beta, permite que os agentes joguem xadrez em um nível extremamente alto, rivalizando com os melhores jogadores humanos e superando-os na maioria das situações.

3.5 Pesquisa de Quiescencia

A técnica de pesquisa de quiescência é crucial no desenvolvimento de programas de xadrez avançados para lidar com as limitações da abordagem convencional do alfa-beta pruning no minimax quando o agente precisa avaliar situações envolvendo capturas e xeques críticos que podem resultar em avaliações imprecisas se a análise for interrompida prematuramente pela profundidade fixada na árvore de busca padrão.A pesquisa de quiescência busca aprimorar a precisão das análises ao estender a exploração além das jogadas táticas cruciais ou capturas imediatas.

3.5.1 O Problema do Horizonte de Pesquisas

Durante a análise de um jogo de forma superficial por um jogador convencional pode acontecer de se chegar a um momento em que o processo de avaliação está em andamento mas existem ameaças iminentes ou xeques que podem alterar significantemente a avaliação da situação atual do jogo. Por exemplo: suponha que o computador conclua que uma determinada posição é favorável para as peças brancas sem levar em consideração que as peças pretas têm a oportunidade de capturar uma peça valiosa no próximo lance. Neste caso específico a profundidade da análise pode ter atingido seu limite e portanto o movimento de captura não é levado em conta na decisão tomada pelo algoritmo. Isso leva a uma conclusão equivocada porque a máquina pode não se dar conta de que o preto tem uma reação imediata e impactante.

O estudo da quiescência foi desenvolvido para lidar com essa questão. A ideia é simples: em vez de interromper a investigação em uma profundidade específica para calcular a avaliação da posição atual, o mecanismo continua a analisar a posição se ainda houver atividade significativa (como capturas ou xeques importantes) disponíveis.

A pesquisa de quiescência expande a árvore de busca apenas para capturas, promoções e cheques. Esses movimentos são chamados de "movimentos de quiescência", pois são jogadas que podem mudar drasticamente a avaliação da posição. Assim, a pesquisa continua até que a posição atinja um estado onde não há mais capturas ou cheques imediatos que possam ser feitos. Este estado é considerado uma posição "quieta"e, finalmente, a avaliação estável é feita.

As fases da pesquisa de quiescência estão detalhadas na seção C.4

3.5.2 Benefícios da Pesquisa de Quiescência

A pesquisa de quiescência aumenta significativamente a precisão das avaliações, especialmente em posições táticas. Ela evita que o agente tome decisões com base em avaliações superficiais de posições instáveis, o que poderia resultar em jogadas erradas.

Sem a pesquisa de quiescência, o agente pode cortar uma linha de pesquisa com base em um valor beta que não leva em conta capturas ou cheques pendentes. Isso resulta em decisões subótimas. A quiescência garante que essas linhas sejam mais bem exploradas antes de tomar uma decisão de corte.

Embora a pesquisa de quiescência não aumente diretamente a profundidade de pesquisa em termos de níveis da árvore, ela melhora a profundidade efetiva em posições táticas importantes. Isso significa que o agente toma decisões com base em informações mais profundas e táticas, mesmo que a pesquisa principal tenha atingido o limite de profundidade.

3.5.3 Limitações

Um dos principais desafios da pesquisa de quiescência é o aumento do custo computacional. Como ela continua a busca em posições que já atingiram a profundidade máxima, pode prolongar o tempo necessário para cada busca. Por isso, é crucial implementar técnicas de poda eficientes para evitar a explosão combinatória. Outra limitação é que a pesquisa de quiescência não explora movimentos não táticos (como movimentar uma peça sem capturas ou cheques), o que pode ser um problema em posições onde um movimento estratégico não tático é a melhor opção. Isso limita sua capacidade de analisar completamente todas as possibilidades da posição.

4 TÉCNICAS PARA DETERMINAR A MELHOR JOGADA

4.1 Algoritmo Minimax

O algoritmo Minimax é um dos métodos mais fundamentais e amplamente utilizados em jogos de dois jogadores com informações completas, como o xadrez. O Minimax tem sido utilizado como base para a tomada de decisão em uma vasta uma infinidade de jogos, principalmente aqueles com natureza competitiva e de soma zero. No contexto do xadrez, o Minimax tem sido utilizado em agentes de xadrez desde os primeiros programas computacionais, sendo o principal componente para determinar o melhor movimento a partir de uma dada posição (SHANNON, 1950).

No xadrez, cada jogador alterna entre fazer movimentos, tentando maximizar suas próprias chances de vitória, enquanto minimiza as do oponente. O Minimax formaliza essa estratégia, simulando todos os possíveis movimentos futuros e atribuindo valores numéricos para cada posição no tabuleiro. A avaliação é feita de acordo com uma função heurística que considera vários fatores, como controle de material, segurança do rei e desenvolvimento de peças. Este capítulo explora a teoria por trás do algoritmo, sua aplicação no xadrez, as melhorias como a poda alfa-beta, e o papel que desempenha em agentes modernos como Stockfish.

4.1.1 Funcionamento

O Minimax opera sob a premissa de que ambos os jogadores tomarão decisões perfeitamente racionais, isto é, o jogador que está movendo buscará maximizar suas chances de vitória, enquanto o oponente buscará minimizá-las. A estrutura básica do algoritmo pode ser representada por uma árvore de decisão, onde cada nó corresponde a uma posição no tabuleiro de xadrez, e os ramos são os possíveis movimentos que podem ser feitos a partir dessa posição (RUSSELL; NORVIG, 2016). Mais detalhes e exemplos na apêndice C.5.

Em cada nível da árvore, o algoritmo alterna entre os dois jogadores. O jogador atual busca maximizar a pontuação (Maximizador), enquanto o oponente busca minimizá-la (Minimizador). A recursão continua até que um nó folha seja atingido, o que corresponde a uma posição terminal do jogo (vitória, derrota, ou empate), ou até que seja atingido um limite de profundidade especificado, no qual o estado da posição é avaliado por

uma função de avaliação heurística.

4.1.2 Poda Alfa Beta

A integração da poda alfa-beta ao Minimax é particularmente relevante no xadrez devido à imensa complexidade do jogo. Com 20 movimentos possíveis na abertura e milhões de combinações subsequentes, a árvore de busca cresce exponencialmente, tornando inviável explorar todas as possibilidades em profundidade. A poda alfa-beta resolve esse desafio ao estabelecer limites para os valores de alfa e beta, que representam, respectivamente, as melhores garantias de pontuação para os jogadores maximizador e minimizador. Se um ramo da árvore oferece um resultado pior do que a melhor alternativa já encontrada, ele é descartado sem avaliação adicional. Essa abordagem permite que a engine concentre seu poder computacional nos ramos mais promissores, resultando em análises mais profundas e decisões mais precisas dentro de um tempo limitado.

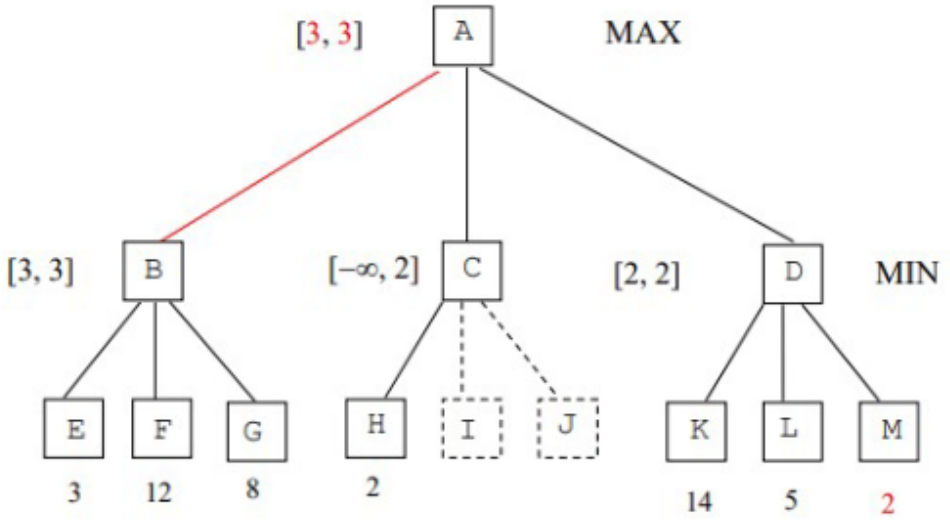
No contexto de agentes modernos de xadrez, o Minimax com poda alfa-beta serve como base para estratégias mais avançadas, como a pesquisa de quiescência e o uso de tabelas de transposição. A pesquisa de quiescência complementa o algoritmo ao lidar com posições instáveis, enquanto as tabelas de transposição armazenam avaliações de posições já analisadas para evitar cálculos redundantes. Em conjunto, essas técnicas permitem que motores de xadrez como Stockfish e Leela Chess Zero alcancem níveis de desempenho sobre-humanos. A aplicação do Minimax com poda alfa-beta, portanto, não só viabiliza a análise eficiente do jogo, mas também destaca o potencial da inteligência artificial para resolver problemas complexos e altamente dinâmicos.

4.1.3 Exemplo de Árvore Minimax

O exemplo abaixo ilustra a aplicação do algoritmo Minimax com poda alfa-beta em uma árvore de decisão no contexto de um jogo, destacando como a poda economiza cálculos. Aqui, os nós rotulados como MAX e MIN indicam que o jogador MAX deseja maximizar sua pontuação, enquanto o jogador MIN deseja minimizá-la.

1. **Primeiro ramo (subárvore de B):** O nó MAX em **A** começa avaliando seu primeiro filho, o nó **B**. O jogador **MIN** em **B** considera os valores 3, 12 e 8 das folhas **E**, **F** e **G**, respectivamente. O menor valor entre eles é 3, o que significa que **B**

Figura 4.1: Exemplo de Árvore do Algoritmo Minimax



Exemplo Algoritmo Minimax com poda alfa-beta (RUSSELL; NORVIG, 2016)

retorna 3 para **A**. Durante essa análise, o intervalo de valores possíveis para **A** é atualizado para [3, 3] (alfa = 3, beta = 3).

2. **Segundo ramo (subárvore de C):** Em seguida, **A** passa para o próximo nó, **C**. No entanto, como o intervalo de valores possíveis já está limitado a [3, 3], a poda alfa-beta descarta imediatamente a análise das folhas **I** e **J**. Isso ocorre porque o valor mínimo possível em **C** não pode superar 3, tornando inútil considerar esse ramo.
3. **Terceiro ramo (subárvore de D):** Por fim, **A** analisa o nó **D**. O jogador **MIN** avalia os valores das folhas **K**, **L** e **M**, que são 14, 5 e 2, respectivamente. O menor valor é 2, que é retornado para **A**. Como 2 é menor que o valor já conhecido de 3, ele não afeta a decisão de **MAX**.

Após a análise, **A** escolhe o valor 3 como sua melhor jogada, ignorando ramos desnecessários graças à poda alfa-beta. Essa otimização reduz significativamente o número de cálculos necessários, economizando tempo e recursos computacionais. Observe que a ordem em que os nodos são analisados é de relevante importância. No terceiro ramo se o primeiro nó a ser analisado fosse o **M** a poda alfa-beta seria mais eficiente, evitando a análise dos nodos **K** e **L**.

4.1.4 Funções de Avaliação

No xadrez, uma função de avaliação é responsável por atribuir um valor numérico a uma posição intermediária. Essa avaliação é crítica para o Minimax, pois nem sempre é possível explorar todas as sequências de movimentos até o fim do jogo devido à vastidão do espaço de busca.

As primeiras funções de avaliação consideravam apenas o material no tabuleiro, atribuindo pesos específicos a cada peça: 1 ponto para peões, 3 para cavalo e bispo, 5 para torres e 9 para a dama. No entanto, agentes modernos como Stockfish utilizam funções mais sofisticadas que consideram fatores como controle de centro, mobilidade de peças, estrutura de peões, segurança do rei e espaço disponível para manobra. Todos esses fatores são ponderados para gerar uma pontuação final que orienta o Minimax na escolha do melhor movimento.

4.1.5 Expansão combinatória

O espaço de busca no xadrez é extraordinariamente grande. Após os primeiros 4 movimentos de cada jogador, o número de posições possíveis no tabuleiro ultrapassa 288 bilhões. O algoritmo Minimax, em sua forma mais básica, exige a exploração de todas essas posições, o que é computacionalmente inviável para uma profundidade significativa, como as 30 ou 40 jogadas que uma partida típica de xadrez pode exigir.

Esse fenômeno, conhecido como "explosão combinatória", foi um dos principais desafios enfrentados pelos primeiros programadores de IAs para xadrez. Para contornar esse problema, uma série de otimizações foram desenvolvidas. A mais conhecida é a poda alfa-beta.

4.2 Pesquisa Monte Carlo (MCTS)

O algoritmo Minimax com Pesquisa Monte Carlo (Monte Carlo Tree Search, ou MCTS) é uma abordagem que combina os princípios do Minimax com técnicas de simulação e análise probabilística, sendo amplamente utilizado em jogos com espaço de busca grande e profundo, como o xadrez e Go. A MCTS fornece uma maneira eficaz de navegar em árvores de decisões complexas, aplicando simulações estatísticas (ou "jogos

simulados") para aprimorar a precisão e profundidade das previsões. Diferente do Minimax puro, que depende principalmente de funções heurísticas, a Pesquisa Monte Carlo se apoia em análises estatísticas, permitindo que as decisões sejam baseadas em resultados obtidos através de simulações repetitivas (MADAKE et al., 2023).

4.2.1 Estrutura Básica do Algoritmo Monte Carlo

A Pesquisa Monte Carlo usa uma árvore de busca composta por nós que representam diferentes estados do jogo. Cada nó da árvore contém informações sobre o estado do tabuleiro e dados de vitória/derrota para esse estado, calculados a partir de simulações. O MCTS opera em quatro fases principais:

4.2.1.1 Seleção com o UCT

A árvore é percorrida a partir da raiz, escolhendo nós baseados em uma política de seleção, geralmente com o uso do *Upper Confidence Bound applied to Trees* (UCT). Cada nó representa um estado do jogo, e cada aresta representa uma ação tomada para se alcançar um estado subsequente. O *UCT* é uma estratégia que maximiza a exploração e exploração dos nós, balanceando a busca por caminhos menos explorados com a preferência por nós com melhores recompensas. A fórmula do UCT é dada por:

$$UCT = \frac{v_i}{n_i} + C \cdot \sqrt{\frac{\ln N}{n_i}} \tag{4.1}$$

onde:

v_i representa o valor médio do nó.

n_i é o número de visitas ao nó.

N é o número total de simulações realizadas.

C é uma constante de ajuste que controla o equilíbrio entre exploração e exploração.

O uso dessa fórmula permite ao MCTS focar em nós que mostram boas promessas, ao mesmo tempo em que continua a explorar novas possibilidades, evitando um overfitting precoce da árvore (*Kocsis & Szepesvári*, 2006).

4.2.1.2 Expansão

Quando um nó que representa um estado inexplorado é atingido, um novo nó filho é criado e adicionado à árvore. Isso aumenta o número de estados conhecidos pelo algoritmo. Após selecionar um nó promissor, o algoritmo verifica se ele já foi explorado por completo. Caso contrário, ele expande a árvore criando novos nós-filhos, cada um representando um possível movimento a partir do estado atual. Essa expansão aumenta o número de caminhos possíveis para simulação, o que melhora a precisão do MCTS ao longo das iterações.

4.2.1.3 Simulação

Uma simulação aleatória do jogo é executada a partir do estado representado pelo nó recém-adicionado, até que um estado terminal seja alcançado. Durante essa fase, são feitas jogadas aleatórias para prever o resultado potencial de um movimento, considerando estratégias gerais e movimentos viáveis. A etapa de simulação é onde o algoritmo aplica a pesquisa Monte Carlo propriamente dita, conduzindo várias simulações completas (ou quase completas) até um estado final do jogo. O objetivo é determinar qual movimento resulta nas maiores taxas de vitória para o jogador atual. Nessa fase, as simulações podem ser realizadas de forma aleatória ou com base em alguma heurística simples.

4.2.1.4 Retropropagação

O resultado da simulação é então retropropagado pela árvore, atualizando as estatísticas de cada nó percorrido. Dessa forma, o algoritmo ajusta a recompensa de cada nó, refletindo o valor das decisões ao longo das sequências exploradas. Ao concluir uma simulação, o resultado final (vitória, derrota ou empate) é então retropropagado pela árvore até o nó inicial. Cada nó registra o total de vitórias, derrotas e empates derivados dos caminhos testados e ajusta seu valor de UCT conforme novos dados são registrados. Isso significa que o algoritmo aprende a priorizar movimentos que produzem melhores resultados e a reduzir a probabilidade de explorar caminhos ineficazes.

Esse ciclo de quatro etapas continua até que um limite de tempo ou uma profundidade de busca seja atingido, e, ao final, o movimento mais promissor é escolhido com base na árvore gerada.

4.2.2 Aplicações da MCTS nos Agentes de Xadrez

A MCTS tornou-se uma técnica popular em agentes de xadrez de última geração devido à sua habilidade em avaliar posições de jogo complexas de forma probabilística. A MCTS é particularmente útil em situações onde não é possível calcular diretamente um valor exato para cada movimento, como ocorre em posições finais complicadas.

Na prática, a MCTS é capaz de competir com o Minimax em profundidade de análise, mas com uma diferença chave: enquanto o Minimax tenta explorar todas as ramificações possíveis dentro de uma profundidade específica, a MCTS se concentra em realizar simulações profundas nas áreas mais promissoras, oferecendo uma maneira mais focada de lidar com o espaço de busca. Agentes modernos de xadrez, como AlphaZero, da DeepMind, incorporam uma variante da MCTS, combinada com redes neurais profundas para obter resultados ainda mais precisos.

4.2.3 Benefícios

A MCTS pode se adaptar dinamicamente a diferentes tipos de jogos e posições, explorando movimentos com base na probabilidade de sucesso. Embora nem sempre seja capaz de explorar todas as opções, a MCTS compensa isso com simulações que aumentam a precisão ao longo das iterações. Como o algoritmo depende de simulações probabilísticas, ele é menos dependente de heurísticas fixas para avaliar posições intermediárias.

4.2.4 Limitações

A MCTS exige um número significativo de simulações para garantir precisão. Em jogos com limites de tempo apertados, isso pode ser uma desvantagem. Escolher um valor ideal para o parâmetro c (na fórmula UCT) é crítico para otimizar o desempenho da MCTS. Valores inadequados podem levar a decisões subótimas. Em árvores com baixa profundidade de ramificação, como em jogos com menos variações de movimento, o Minimax puro pode superar a MCTS em precisão.

4.3 Heurística de Avaliação Posicional

A heurística de avaliação posicional é um aspecto fundamental dos agentes modernos de xadrez, responsável por analisar e quantificar as características de uma posição de xadrez para orientar a escolha de movimentos. Essa avaliação não garante que o movimento escolhido seja o melhor possível, mas busca identificar um valor numérico que reflita as chances de vantagem para cada lado. A análise posicional é vital porque permite que o algoritmo de busca encontre movimentos promissores em profundidades mais rasas, reduzindo o esforço computacional e ajudando o agente a "pensar"estrategicamente.

Um algoritmo de avaliação posicional bem implementado utiliza uma função de avaliação combinada, onde cada fator é multiplicado por seu respectivo peso, e os valores resultantes são somados para gerar uma pontuação final. Essa função de avaliação, frequentemente chamada de Função de Avaliação Linear, é expressa assim:

$$f(\text{posição}) = a\times\text{Material} + b\times\text{Mobilidade} + c\times\text{Estrutura de Peões} + d\times\text{Segurança do Rei} + \ldots$$

Essa fórmula representa a função de avaliação combinada de uma posição no xadrez, onde cada fator é ponderado por uma constante ajustável, e o valor total orienta o agente sobre a qualidade da posição. Cada variável (a, b, c, d) representa o peso de cada componente de avaliação e é ajustada de acordo com experimentos e a análise de partidas anteriores. Agentes como o Stockfish e o Komodo ajustam esses valores automaticamente com base em dados históricos e feedback de jogos simulados, enquanto outros agentes de xadrez podem usar valores pré-definidos.

4.4 Componentes da Avaliação Posicional

As heurísticas de avaliação geralmente combinam uma variedade de fatores que ajudam o agente a avaliar posições.

4.4.1 Valor Material

O valor material é a base para qualquer avaliação posicional. Cada peça de xadrez é atribuída a um valor numérico: 1 para peões, 3 para cavalos e bispos, 5 para torres e

9 para a dama. O rei, como peça insubstituível, não recebe valor numérico, embora sua segurança seja crítica. A soma dos valores das peças restantes no tabuleiro ajuda o agente a medir qual lado tem mais força material disponível.

4.4.2 Atividade e Mobilidade

A mobilidade refere-se ao número de casas acessíveis a uma peça sem ser capturada. Peças com mais opções de movimento (mobilidade) são consideradas mais ativas. Essa atividade é especialmente valorizada em bispos e torres, que possuem maior alcance. A mobilidade é valorizada principalmente em posições de meio-jogo, onde a atividade das peças pode abrir caminho para ataques ou defesas cruciais (Campbell et al., 2002).

4.4.3 Controle do Centro

O controle das casas centrais (d4, d5, e4, e5) é essencial para a avaliação, pois quem controla o centro possui mais opções para coordenar ataques e defesas. Agentes de xadrez consideram peças que ocupam ou controlam o centro mais vantajosas, atribuindo valores adicionais a essas posições centrais.

4.4.4 Estrutura de Peões

A estrutura de peões afeta profundamente a avaliação posicional. Peões dobrados, peões isolados e peões passados são exemplos de estruturas que influenciam a posição. Peões passados (aqueles que não têm peões inimigos em sua coluna ou nas colunas adjacentes) são valiosos por seu potencial de promoção. Já os peões dobrados (dois peões da mesma cor na mesma coluna) e isolados (sem peões adjacentes) são fraquezas que podem ser exploradas pelo adversário.

4.4.5 Segurança do Rei

Avaliar a segurança do rei é uma prioridade nas heurísticas posicionais. O agente examina fatores como a presença de peões defensivos ao redor do rei e a possibilidade de

ataques adversários. No final do jogo, a segurança do rei pode ser menos prioritária do que o avanço de peões para promoção, mas no meio-jogo, um rei exposto pode rapidamente levar a uma derrota.

4.4.6 Par de Bispos

Em posições abertas, a posse de ambos os bispos é geralmente considerada uma vantagem, pois eles cobrem mais espaço do tabuleiro e podem coordenar ataques mais eficazes. Em posições fechadas, no entanto, a vantagem do par de bispos pode ser menor, dependendo do contexto posicional. Em casos em que há muitos peões alinhados em casas da mesma cor, é possível que o bispo dessa mesma cor de casa tenha seu valor rebaixado e seja, eventualmente, trocado por um cavalo, bispo ou até mesmo um peão (por exemplo o peão defensivo do rei) do adversário para pemitir melhor condição posicional.

4.5 Combinações e Pesos na Heurística de Avaliação

Para alcançar uma avaliação precisa, os agente ajustam o peso de cada fator conforme a fase do jogo (abertura, meio-jogo, ou final de jogo).

- Fase Abertura e Meio-Jogo: Nessas fases, a mobilidade e o controle central são altamente valorizados. O par de bispos também tende a receber uma avaliação favorável em posições abertas.
- Fase Final de Jogo: À medida que as peças vão sendo trocadas, a avaliação do valor material se torna mais importante, pois menos peças restam para proteger o rei. A segurança do rei também diminui em importância, enquanto o avanço de peões e a criação de peões passados aumentam em prioridade (Levy e Newborn, 1991).

Os pesos atribuídos a cada fator são calculados empiricamente, geralmente com base em milhões de jogos de xadrez analisados pelo agente. Esse ajuste permite que a avaliação posicional evolua conforme novas estratégias são descobertas e implementadas.

4.6 Evolução e Aprimoramentos da Avaliação Posicional

Os métodos de avaliação posicional evoluíram significativamente ao longo do tempo. Nos primeiros agentes, as heurísticas eram simples e baseadas em tabelas de valores fixos. Com o avanço da tecnologia, o uso de redes neurais e aprendizado de máquina tem permitido uma avaliação posicional mais precisa e adaptável.

4.6.1 Avaliação Automatizada com Redes Neurais

Redes neurais modernas, como as usadas em AlphaZero, são capazes de aprender padrões complexos de avaliação posicional de forma independente, ao analisar milhões de posições. Em vez de pré-definir valores para cada fator, a rede neural ajusta esses valores com base nos dados de treinamento, permitindo uma avaliação que pode evoluir e adaptar-se a novas posições (Silver et al., 2018).

4.6.2 Avaliação Dinâmica

A pesquisa e o desenvolvimento contínuos permitem que agentes de xadrez ajustem automaticamente as suas heurísticas, baseando-se em novas descobertas e adaptações de estratégias, como a inclusão de uma heurística específica para avaliar peões passados e o controle de casas centrais em certas aberturas.

4.7 Limitações

A heurística de avaliação posicional, embora amplamente utilizada nos agentes de xadrez, apresenta algumas limitações que impactam seu desempenho em diversas situações.

A heurística posicional é limitada ao uso de critérios definidos previamente, como valores materiais, mobilidade, segurança do rei e estrutura de peões, que funcionam bem para posições estáticas e de longo prazo. No entanto, em situações onde há táticas complexas ou sacrifícios temporários, essa heurística tende a falhar, pois não leva em conta combinações de longo alcance e exceções em que um sacrifício material temporário pode levar a uma vitória (Campbell et al., 2002).

Cada fator na heurística de avaliação é ponderado por um valor fixo, como o peso atribuído ao material ou à segurança do rei. Esses valores podem não se ajustar bem em diferentes fases do jogo (como transições entre meio-jogo e final de jogo), fazendo com que o agente avalie incorretamente uma posição que, em outra fase do jogo, teria um peso diferente (Silver et al., 2018).

Em posições fechadas, onde as peças têm pouca mobilidade, é desafiador para a heurística distinguir entre posições vantajosas e desvantajosas. A análise de mobilidade ou controle central, por exemplo, torna-se menos relevante, e o agente pode ter dificuldades em identificar planos de longo prazo, como manobras que reconfiguram as peças para abrir a posição (Levy e Newborn, 1991).

Em agentes baseados em aprendizado de máquina, os valores e pesos para avaliação podem ser ajustados com o treinamento, mas em heurísticas tradicionais, os valores são estáticos. Isso significa que o agente pode falhar em se adaptar a novas posições ou estilos de jogo. Por exemplo, contra adversários que jogam aberturas menos comuns, a heurística pode fazer uma avaliação errada pela falta de dados adaptáveis na função de avaliação posicional (Sadler e Regan, 2019).

A heurística de avaliação posicional falha ao capturar padrões estratégicos complexos, como o conceito de domínio espacial ou controle de peças que ainda não foram desenvolvidas. Esses padrões muitas vezes exigem uma análise mais abstrata e não são facilmente traduzíveis em pontuações específicas, dificultando a resposta precisa da heurística (Vinyals et al., 2019).

Quando a heurística de avaliação posicional se torna computacionalmente complexa, ela pode comprometer significativamente o desempenho do algoritmo minimax. Como o minimax precisa avaliar uma grande quantidade de posições para cada movimento possível (especialmente com poda alfa-beta), qualquer aumento no tempo necessário para calcular a função de avaliação em cada posição afeta diretamente o número de posições que podem ser analisadas em um período limitado. Isso reduz a profundidade da busca e limita a capacidade do agente de antecipar movimentos futuros, afetando a precisão das decisões estratégicas. Para manter a eficiência, as heurísticas devem ser otimizadas, focando apenas nos fatores mais impactantes para uma avaliação rápida e precisa (Russell e Norvig, 2010; Campbell et al., 2002)

5 DESENVOLVIMENTO DE UM AGENTE DE XADREZ

O desenvolvimento de um agente de xadrez é um processo desafiador que combina conhecimentos de programação, inteligência artificial e o domínio de regras e estratégias do jogo. Este capítulo detalha as etapas seguidas no desenvolvimento do agente de xadrez para este projeto de TCC, abordando as escolhas tecnológicas, o ambiente de desenvolvimento e as metodologias empregadas. A proposta foi inspirada por agentes renomados, como o Stockfish, mas adaptada para o contexto acadêmico e restrições técnicas do projeto.

5.1 Ambiente de Desenvolvimento

Para realizar o desenvolvimento do agente, foi utilizada uma máquina desktop com especificações modestas, mas adequadas para a tarefa. A escolha do ambiente de desenvolvimento e ferramentas foi baseada em compatibilidade com as necessidades do projeto e na familiaridade com os recursos oferecidos pelas ferramentas selecionadas.

A construção do agente de xadrez seguiu uma abordagem incremental e iterativa, onde cada passo introduziu novas funcionalidades e melhorias técnicas. Ao longo do desenvolvimento, foram analisados os efeitos de cada etapa em termos de comportamento do agente, limitações encontradas e os desafios superados.

5.1.1 Hardware

A máquina de referência para o desenvolvimento foi configurada com as seguintes especificações técnicas:

- Processador (CPU): Intel® Core™ i5-9400F, com 6 núcleos físicos e suporte a 6 threads.
- Memória RAM: 16GB, proporcionando capacidade suficiente para manipular estruturas de dados complexas e realizar testes intensivos.
- Placa-Mãe (MOBO): ASUSTeK TUF B360M-PLUS.
- Sistema Operacional: Linux Ubuntu 24.04.1 LTS, escolhido por sua estabilidade e amplo suporte a ferramentas de desenvolvimento.

5.1.2 Software

O projeto foi implementado utilizando as seguintes ferramentas de software:

- Linguagem de Programação: Java, com a versão OpenJDK 21.0.5, por oferecer uma plataforma robusta e um ecossistema extenso de bibliotecas úteis.
- IDE: Eclipse 2024.6, que proporcionou um ambiente integrado para escrita, depuração e testes do código.
- Estudo do Agente Stockfish: A análise do código-fonte do agente Stockfish foi realizada utilizando o compilador G++ versão 13.2.0 e a IDE Visual Studio Code (VSCode) versão 1.94.2, permitindo maior compreensão dos algoritmos de referência escritos em C++.
- Ferramenta de diff: A ferramenta Meld foi utilizada para comparar o log com o trace gerado pelo Stockfish com o log gerado pela nosso agente

A combinação dessas ferramentas e equipamentos forneceu uma base sólida para o desenvolvimento do agente, permitindo explorar conceitos avançados como poda alfa-beta, tabelas de transposição e heurísticas de avaliação posicional assim como o estudo do código fonte do Agente Stockfish.

5.2 Implementação Inicial

Na primeira etapa do desenvolvimento do agente, foram aplicados os conhecimentos adquiridos nas aulas da disciplina de Inteligência Artificial sobre jogos de tabuleiro. O algoritmo base escolhido foi o minimax com poda alfa-beta, uma técnica amplamente utilizada para otimizar a busca por jogadas em árvores de decisão. A implementação inicial priorizou a simplicidade, com um foco em garantir que o agente fosse funcional e capaz de tomar decisões básicas no tabuleiro.

5.2.1 Minimax Puro

O primeiro passo consistiu na implementação básica do algoritmo Minimax, explorando uma profundidade de 6 níveis. Essa abordagem permitiu que o agente tomasse decisões baseadas em um modelo recursivo de maximização e minimização de pontua-

ção. Embora o agente fosse capaz de se defender bem contra ataques adversários, ele apresentava um comportamento extremamente reativo, priorizando evitar perdas em vez de buscar oportunidades de ataque. Essa limitação tornava o estilo de jogo pouco proativo e previsível. A primeira versão da função de avaliação levou em consideração apenas a avaliação material sem considerar a situação posicional.

A implementação inicial demandou atenção para evitar redundâncias e garantir a correta alternância de turnos na árvore de busca. A ausência de heurísticas mais avançadas resultou em um alto custo computacional, limitando a capacidade de aumentar a profundidade de busca.

5.2.2 Tabela de Valores de Posição (PST)

O segundo passo introduziu uma tabela de valores de posição (Piece-Square Table, PST) para melhorar a função de avaliação. Cada peça passou a ser avaliada não apenas pelo seu valor intrínseco, mas também pela sua posição no tabuleiro. Essa modificação trouxe um ganho significativo em termos de proatividade do agente, que passou a buscar melhores posicionamentos e a explorar mais o centro do tabuleiro.

O agente tornou-se mais estratégico, preferindo posições que maximizassem a influência no tabuleiro. A tabela PST adicionou um componente posicional à avaliação, permitindo um jogo mais equilibrado e um agente com mais iniciativa. Agora a função de avaliação leva em consideração não apenas a avaliação material mas também o peso posicional da tabela PST.

Determinar os valores ideais para cada posição na tabela exigiu uma pesquisa na literatura para encontrar os valores PST ideais para cada posição do tabuleiro e que variavam de acordo com as fases do jogo: abertura, meio de jogo e fim de jogo.

5.2.3 Poda Alfa-Beta

Nesse passo, foi implementada a poda Alfa-Beta, que reduziu significativamente o número de nós avaliados na árvore de busca. Essa otimização permitiu aumentar a profundidade de busca para 7 níveis, ampliando a capacidade do agente de planejar movimentos em maior detalhe.

O desempenho computacional foi aprimorado, reduzindo o tempo necessário para

cada movimento ou o aumento de um nível de profundidade. O agente tornou-se mais robusto ao evitar análises redundantes de ramos irrelevantes.

A implementação inicial exigiu cuidado para garantir que os cortes ocorressem nos momentos corretos sem comprometer a precisão da busca.

5.2.4 Ordenação PV

No terceiro passo, foi introduzida a ordenação da Principal Variation (PV), priorizando os ramos mais promissores da árvore de busca com base em heurísticas iniciais. Com essa técnica, a profundidade de busca foi aumentada para 8 níveis, permitindo decisões ainda mais estratégicas. O algoritmo de ordenação recomendado na literatura devido as características da variação principal é o *Insertion Sort*.

A ordenação de movimentos acelerou o processo de busca, tornando a poda alfa beta do passo anterior mais eficiente. Os movimentos do agente começaram a refletir maior capacidade de antecipação de jogadas adversárias. A implementação da ordenação exigiu testes extensivos para evitar inversões inadequadas na sequência de movimentos avaliados.

5.2.5 Tabela de Transposição

No quinto passo, foi incorporada uma tabela de transposição para armazenar os resultados de subárvores já analisadas. Isso reduziu a redundância nas análises e permitiu elevar a profundidade de busca para 9 níveis.

O desempenho foi significativamente otimizado, especialmente em posições repetitivas ou similares. O agente tornou-se capaz de explorar melhor posições complexas, aumentando sua eficiência geral.

A gestão da tabela de transposição exigiu ajustes para evitar consumo excessivo de memória. Também foi necessário implementar uma política de substituição para lidar com colisões na tabela.

5.2.6 Aprimoramento da Função de Avaliação

No último passo, a função de avaliação foi aprimorada para incluir métricas de mobilidade e ataques. Essa modificação proporcionou uma visão mais completa do tabuleiro, permitindo o agente avaliar não apenas a posição estática, mas também o potencial dinâmico de cada movimento. A melhoria da função de avaliação aumentou significativamente a eficiência da poda alfa beta permitindo o aumento da profundidade de busca para 10 níveis.

O agente apresentou um comportamento mais proativo, com jogadas mais agressivas e estratégicas. A avaliação de mobilidade permitiu maior exploração de possibilidades, enquanto a análise de ataques reduziu vulnerabilidades.

Apesar das melhorias, o agente demonstrou dificuldades em finalizar jogos no fim de partida, especialmente para encontrar o xeque-mate. A implementação de métricas adicionais aumentou a complexidade da função de avaliação, exigindo maior tempo de ajuste e testes.

5.2.7 Reflexão Sobre o Desenvolvimento da Implementação Inicial

O processo de desenvolvimento do agente evidenciou uma evolução clara no comportamento do agente. Enquanto a versão inicial com Minimax puro era defensiva e reativa, o uso de heurísticas e técnicas avançadas, como a tabela de transposição e a melhoria na função de avaliação, transformou o agente em um jogador mais estratégico e proativo. Nossa implementação ficou interessante e divertida para jogadores iniciais e intermediários de xadrez. Entretanto, jogadores mais avançados conseguem vencê-la com certa facilidade. Ao comparar com agentes consolidados como o Stockfish, Komodo ou o GNU-Chess percebemos o abismo tanto em termos estratégico como de performance. Afinal, o que existe no Stockfish que o torna tão rápido e tão preciso? Vamos analisar o código, estudar e incorporar os recursos do Stockfish em nosso Agente.

5.3 Incorporação de Técnicas Avançadas do Stockfish

A segunda etapa do desenvolvimento do agente envolveu estudar e adaptar recursos avançados utilizados pelo motor Stockfish, reconhecida como uma das melhores

agentes de xadrez do mundo. Apesar de sua complexidade, foi decidido não incorporar recursos baseados em redes neurais neste projeto, deixando essa implementação como possibilidade para trabalhos futuros. O foco foi na adaptação do algoritmo minimax com poda alfa-beta, inspirado nas técnicas de poda agressiva e funções de avaliação sofisticadas presentes na Stockfish.

5.3.1 Geração de Movimentos (classe *movegen*)

No motor Stockfish, a classe *movegen* é responsável por gerar movimentos pseudo-legais de maneira extremamente rápida. Uma das principais inovações encontradas nessa classe é o uso de bitboards mágicos.

Os bitboards mágicos são utilizados para calcular movimentos potenciais de cada peça individualmente, levando em conta restrições como bloqueios de outras peças. Por exemplo, ao calcular os movimentos de uma torre, os bitboards mágicos permitem identificar rapidamente todas as casas que a torre pode atingir sem necessidade de percorrer cada casa do tabuleiro sequencialmente. Esta abordagem reduz significativamente o tempo de execução, especialmente em posições complexas.

O conjunto de bitboards mágicos é completamente pré-calculado durante a inicialização do agente e é acessado de forma estática nas etapas de geração da lista de movimentos. Assim os movimentos são gerados sem a necessidade de iteraração para cada possível movimento de uma determinada peça, resultando em um bitboard com todas as movimentações possíveis da peça em questão.

5.3.2 Lista de Movimentos (classe *movepicker*)

Outro aspecto essencial do Stockfish é a classe *movepicker*, que organiza os movimentos gerados pelo *movegen* em uma ordem que favorece a eficiência do algoritmo de poda alfa-beta. A ordenação inicial é realizada com base em um conjunto de etapas, cada uma das quais reflete a probabilidade de um movimento ser relevante em determinado contexto. A Tabela 5.1 apresenta a lista de etapas para montagem da lista de movimentos.

5.3.2.1 Etapa de Movimentos Quietos

Movimentos quietos são aqueles que não resultam em captura de peças adversárias, não colocam o rei adversário em xeque e não promovem peões. Eles são importantes porque podem preparar ataques futuros, melhorar a posição das peças ou fortalecer a defesa sem provocar uma resposta imediata do oponente. Em termos de estratégia, movimentos quietos são frequentemente usados para melhorar a coordenação das peças e criar ameaças sutis que podem ser exploradas em lances subsequentes.

5.3.2.2 Etapa de Movimentos de Refutação

Movimentos de refutação são lances que respondem a uma jogada do adversário de maneira a neutralizar ou refutar a ameaça ou plano que ele estava tentando implementar. Esses movimentos são críticos em situações onde o oponente tenta uma combinação tática ou uma armadilha, e a refutação correta pode transformar uma posição desfavorável em uma vantajosa. Por exemplo, se o adversário faz um ataque que parece forte, mas você encontra um movimento de refutação que desmonta o ataque e deixa o adversário em uma posição pior, esse movimento é considerado uma refutação.

5.3.2.3 Etapa de Capturas Ruins

Capturas ruins são movimentos de captura que, embora removam uma peça adversária do tabuleiro, resultam em uma posição desfavorável para o jogador que realiza a captura. Isso pode ocorrer, por exemplo, quando a peça capturadora fica exposta a um contra-ataque imediato ou quando a captura não compensa a perda de material ou posição subsequente. Em outras palavras, são capturas que, ao serem analisadas mais profundamente, acabam prejudicando mais do que ajudando.

5.3.2.4 Etapa de Evasão

Evasion refere-se a movimentos que um jogador faz para escapar de uma ameaça direta, como um xeque. Esses movimentos são críticos em situações onde o rei está sob ataque e precisa se mover para uma casa segura ou quando uma peça precisa ser movida para evitar uma captura iminente. A eficiência na geração e avaliação de movimentos de evasão é essencial para a defesa eficaz em posições complicadas.

5.3.2.5 Etapa de Probcut

Probcut é uma técnica de poda seletiva utilizada para melhorar a eficiência do algoritmo alfa-beta. Desenvolvida por Michael Buro em 1994, essa técnica permite excluir subárvores provavelmente irrelevantes de serem pesquisadas profundamente1. A ideia é que o resultado de uma busca superficial pode ser uma estimativa razoável do resultado de uma busca mais profunda. Se a busca superficial indica que uma subárvore é improvável de alterar o resultado final, essa subárvore pode ser podada, economizando tempo de processamento. Probcut e sua variante Multi-Probcut (MPC) têm sido eficazes em jogos como Othello e Shogi, e foram implementados com sucesso em agentes de xadrez como o Stockfish.

Tabela 5.1: Etapas do Movepicker e suas Descrições

Etapa	**Descrição**
MAIN_TT	Movimentos principais da tabela de transposição
CAPTURE_INIT	Inicialização de capturas
GOOD_CAPTURE	Capturas boas
REFUTATION	Movimentos de refutação
QUIET_INIT	Inicialização de movimentos quietos
QUIET	Movimentos quietos
BAD_CAPTURE	Capturas ruins
EVASION_TT	Evasões da tabela de transposição
EVASION_INIT	Inicialização de evasões
EVASION	Evasões
PROBCUT_TT	Cortes probabilísticos da tabela de transposição
PROBCUT_INIT	Inicialização de cortes probabilísticos
PROBCUT	Cortes probabilísticos
QSEARCH_TT	Busca quiescente da tabela de transposição
QCAPTURE_INIT	Inicialização de capturas na busca quiescente
QCAPTURE	Capturas na busca quiescente
QCHECK_INIT	Inicialização de cheques na busca quiescente
QCHECK	Cheques na busca quiescente

5.3.3 Poda Agressiva no Minimax do Stockfish

Nesta seção exploraremos as técnicas de poda agressiva implementadas no algoritmo Minimax do motor Stockfish. A análise dos comentários do tipo "Step"no código do Stockfish revela uma série de estratégias sofisticadas que aumentam significativamente a eficiência da busca, permitindo que o agente avalie posições de forma mais rápida e precisa.

As técnicas de poda agressiva implementadas no Stockfish permitem uma análise mais eficiente e precisa das posições no tabuleiro. Ao eliminar movimentos e subárvores que são improváveis de alterar o resultado final, o agente pode focar seus recursos computacionais em movimentos mais promissores, resultando em uma performance superior.

Essas estratégias de poda são essenciais para a competitividade do agente, permitindo que ele avalie posições complexas de forma rápida e precisa, mantendo-se à frente em um campo altamente competitivo.

5.3.3.1 Poda de Distância de Mate

Esta técnica evita buscas desnecessárias quando um mate mais curto já foi encontrado em um nível superior da árvore de decisão. Se a melhor pontuação possível (mate-in) não supera o valor de alfa atual, a busca é interrompida, economizando tempo de processamento.

5.3.3.2 Razoring

Esta técnica elimina movimentos que, com base em uma avaliação superficial, são improváveis de superar o valor de beta. Isso permite focar a busca em movimentos mais promissores.

5.3.3.3 Poda de Futilidade

Movimentos que não têm potencial de melhorar significativamente a posição são podados, especialmente em níveis mais profundos da árvore de decisão. Isso é baseado na premissa de que certos movimentos são fúteis e não merecem uma análise detalhada.

5.3.3.4 Busca de Movimento Nulo

Esta técnica envolve fazer uma "jogada nula"(passar a vez) para verificar se a posição do oponente é suficientemente forte para justificar uma poda. Se a posição resultante da jogada nula é muito boa para o oponente, a busca pode ser podada.

5.3.3.5 ProbCut

Uma técnica de poda probabilística que elimina subárvores com base em uma busca superficial que sugere que a subárvore é improvável de alterar o resultado final.

Isso é especialmente útil para capturas que parecem promissoras, mas que, após uma análise mais profunda, não justificam a busca completa.

5.3.3.6 Poda em Profundidade Rasa

Movimentos que não são promissores em profundidades rasas são podados para focar a busca em movimentos mais relevantes.

5.3.3.7 Extensões

Em certos casos, movimentos são estendidos para uma análise mais profunda, especialmente se um movimento singular é identificado como potencialmente decisivo.

5.3.3.8 Busca de Profundidade Reduzida (LMR)

Movimentos são inicialmente pesquisados em profundidade reduzida. Se um movimento falha em superar o valor de beta, ele é reavaliado em profundidade total.

5.4 Avaliação de Tabuleiros no Xadrez por Motores como o Stockfish

A avaliação de tabuleiros é uma das principais funções desempenhadas por agentes modernos de xadrez, como o Stockfish. Esta funcionalidade permite que o agente atribua uma pontuação numérica a uma posição no tabuleiro, orientando suas escolhas e ajudando na tomada de decisões durante a busca minimax com poda alfa-beta. O valor resultante reflete a vantagem ou desvantagem relativa de um jogador na posição analisada. Para isso, diversas componentes são avaliadas individualmente e combinadas em uma heurística sofisticada que engloba elementos estratégicos e táticos.

5.4.1 Estrutura de Peões

A estrutura de peões é um dos elementos mais importantes na avaliação posicional. Agentes analisam a integridade e a posição dos peões no tabuleiro, buscando identificar fraquezas como peões dobrados, isolados ou atrasados. Além disso, o avanço de peões em colunas abertas ou semi-abertas pode gerar uma posição favorável, especialmente quando os peões avançados estão protegidos e prontos para promover.

5.4.2 Controle do Tabuleiro

O controle das casas centrais e de colunas abertas é um fator essencial na avaliação do tabuleiro. Os agentes de xadrez consideram a ocupação das casas centrais (e4, d4, e5, d5) como um indicativo de superioridade posicional. Peças centralizadas tendem a ter maior alcance e influência em comparação àquelas posicionadas em extremidades do tabuleiro.

5.4.3 Atividade e Mobilidade das Peças

A atividade das peças está relacionada à quantidade e qualidade das casas que cada peça pode atingir. Peças ativas são aquelas com maior liberdade de movimento e influência no jogo. A mobilidade, por outro lado, mede a quantidade de movimentos legais disponíveis em uma posição. Agentes de xadrez penalizam peças passivas, como bispos bloqueados por peões ou torres em colunas fechadas.

5.4.4 Segurança do Rei

A segurança do rei é um critério crucial na avaliação de uma posição. O Stockfish verifica a posição do rei em relação a ameaças potenciais, especialmente em partidas abertas onde peças adversárias podem rapidamente atingir a posição do rei. Uma posição com o rei exposto ou vulnerável é penalizada, enquanto uma estrutura de peões saudável e bem protegida ao redor do rei é valorizada.

5.4.5 Avaliação Material

A avaliação material é a base da análise em agentes de xadrez. O agente atribui valores fixos às peças (1 ponto para peões, 3 para cavalos e bispos, 5 para torres e 9 para a dama), mas também considera fatores dinâmicos, como a atividade das peças. Em certas situações, o agente pode preferir uma torre ativa a uma dama passiva, mesmo que o valor material seja menor.

5.4.6 Avaliação Tática

A detecção de táticas é um aspecto crítico do cálculo em agentes de xadrez. Isso inclui a identificação de ameaças diretas, como garfos, cravadas, raios-x e capturas. O Stockfish combina a avaliação tática com a busca de movimentos, garantindo que soluções imediatas e forçadas não sejam negligenciadas.

5.4.7 Avaliação Posicional

A avaliação posicional envolve elementos de longo prazo, como o posicionamento das peças, peões e controle de espaço. Um cavalo em uma casa forte, um bispo em uma diagonal longa ou uma torre em uma coluna aberta são exemplos de aspectos posicionais positivos. O agente leva em consideração a harmonia entre as peças e a capacidade de restringir o jogo adversário.

5.4.8 Avaliação Dinâmica

A avaliação dinâmica analisa fatores temporários que podem influenciar o jogo em um curto prazo. Isso inclui iniciativas táticas, ameaças de ataque e a capacidade de pressionar o adversário. Agentes como o Stockfish reconhecem que um pequeno sacrifício material pode ser compensado por um ganho dinâmico significativo.

5.4.9 Avaliação de Finais

Nos finais de partida, a avaliação do tabuleiro assume características específicas. Agentes de xadrez priorizam a promoção de peões, a atividade do rei e a coordenação das peças restantes. Tabelas de finais (endgame tablebases) são frequentemente utilizadas para identificar sequências perfeitas de jogadas, garantindo vitórias ou empates em posições específicas.

5.5 Reflexão sobre o Desenvolvimento da Agentes de Xadrez

O desenvolvimento de um agente de xadrez em Java teve como principal objetivo o estudo aprofundado do algoritmo minimax com poda alfa-beta, avaliando seu desempenho em diferentes situações do jogo e compreendendo os fatores que influenciam sua eficiência. Como parte desse processo, analisamos o código do motor Stockfish, amplamente reconhecido como um das mais poderosos e eficientes disponíveis atualmente. Esse estudo foi crucial para identificar os avanços tecnológicos e as técnicas que a tornaram tão eficaz, permitindo-nos incorporar parte dessas inovações à nosso agente. Assim, nosso projeto não se limitou ao desenvolvimento de uma ferramenta, mas também visou servir como uma base para estudos futuros no campo da inteligência artificial aplicada ao xadrez.

É importante destacar que o propósito do projeto não foi criar um agente super rápido ou competitivo que pudesse rivalizar com agentes consolidados, como a Stockfish. A escolha da linguagem Java, reconhecidamente mais lenta que C++ em termos de desempenho, reforça esse foco no aprendizado e na aplicação de técnicas, priorizando a clareza e a flexibilidade para fins didáticos. Nosso objetivo foi o estudo, a implementação e a experimentação de conceitos fundamentais, como heurísticas de avaliação posicional, pesquisa de quiescência e tabelas de transposição com Zobrist Hashing, visando entender como esses elementos contribuem para o funcionamento de um agente.

Uma das grandes vantagens de desenvolver nosso próprio agente é a possibilidade de adaptá-la para projetos pedagógicos. Por exemplo, podemos configurá-la para suportar variações simplificadas do xadrez, como partidas em tabuleiros reduzidos ou modalidades específicas como a "batalha de peões", em que o jogo é realizado apenas com peões e sem a presença da peça Rei. Essas variações simplificadas são valiosas no ensino do xadrez, pois permitem que aprendizes foquem em aspectos específicos do jogo. Além disso, elas não podem ser reproduzidas em agentes tradicionais, que geralmente não estão adaptadas para lidar com a ausência de peças fundamentais, como o Rei.

Portanto, o desenvolvimento desse agente, ainda que modesto em termos de desempenho, proporciona um rico campo de aprendizado e experimentação. Mais do que isso, ele pode ser uma ferramenta educativa poderosa, ajudando a democratizar o xadrez e ampliar seu alcance, seja como um jogo, seja como uma ferramenta de desenvolvimento cognitivo e aprendizado estratégico.

6 ANÁLISE E VALIDAÇÃO DE NOSSO AGENTE

A validação do agente desenvolvido foi conduzida em diversas etapas, com o objetivo de garantir a precisão dos movimentos, a eficiência dos algoritmos implementados e a progressão de desempenho ao longo das fases do projeto. Este capítulo descreve as etapas de teste realizadas, os critérios adotados para avaliação e a metodologia utilizada para calcular o Elo do agente.

6.1 Testes Funcionais

A primeira etapa de validação focou em testes funcionais básicos para assegurar que o agente respeitava as regras do jogo de xadrez. Foram realizadas simulações para verificar a execução correta de movimentos básicos e especiais, como:

- **Roque**: Testou-se a movimentação simultânea do Rei e da Torre, obedecendo as condições para sua execução.
- **En Passant**: Validou-se a captura especial de peões, garantindo que fosse aplicada nas condições corretas.
- **Promoção de Peão**: Confirmou-se a transformação de peões ao atingirem a oitava fileira.

Esses testes garantiram que o agente estivesse em conformidade com as regras do xadrez antes de iniciar os testes de desempenho e cálculo do Elo.

6.2 Metodologia de Testes de Desempenho

Após os testes funcionais, iniciou-se uma bateria de partidas contra **bots** da plataforma *Chess.com*. Um **bot** em computação é um software ou programa que automatiza tarefas específicas, simulando o comportamento humano em determinados contextos. A palavra "bot"vem de "robot"(robô) e é usada para descrever programas que executam ações repetitivas, interagem com usuários, ou realizam tarefas de forma autônoma, como simular jogadas de xadrez.

Cada fase de implementação do agente foi testada em partidas de 100 rodadas, configuradas de forma semi-automática, para avaliar o desempenho do agente e estimar

seu Elo. Durante os testes, o tempo máximo de tolerância para cada jogada do agente foi configurado em **1 minuto**, garantindo consistência nas condições de jogo entre as fases.

Um exemplo dessa metodologia pode ser observado na etapa do **Minimax Puro**, onde:

- No nível de profundidade **5**, o agente apresentava um tempo médio de **15 segundos por jogada**.
- No nível de profundidade **6**, o tempo médio subia para **1 minuto e 20 segundos**, excedendo o limite estipulado.

Dessa forma, o nível **5** foi o escolhido para a validação dessa fase, e assim por diante para as demais implementações.

6.3 Cálculo do Elo

Para estimar o Elo do agente em cada fase, utilizamos o seguinte método:

1. Nosso agente jogou **100 partidas** contra bots de Elo fixo (por exemplo, 1500).
2. Ao final das partidas, registramos os seguintes resultados:
   - **63 vitórias**, correspondendo a **63% de aproveitamento**.
   - **9 empates**, correspondendo a **9% de aproveitamento**.
   - **28 derrotas**, correspondendo a **28% de aproveitamento**.

Com esses dados, aplicamos a fórmula de cálculo de Elo baseada na performance relativa:

$$\text{Elo estimado} = \text{Elo oponente} + 400 \times (\text{W} - 0.5)$$

Onde:

- **Elo oponente** é o Elo do bot adversário (1500 neste caso).
- **W** é a taxa de vitórias, calculada como:

$$\text{W} = \frac{\text{Vitórias} + 0.5 \times \text{Empates}}{\text{Total de partidas}} = \frac{63 + 0.5 \times 9}{100} = 0.675$$

Substituindo na fórmula:

$$\text{Elo estimado} = 1500 + 400 \times (0.675 - 0.5) = 1500 + 400 \times 0.175 = 1500 + 70 = 1570$$

Assim, o Elo estimado para essa etapa do agente foi **1570**, considerando os resultados obtidos contra um bot de Elo 1500. Esse cálculo foi repetido para cada fase, refletindo os incrementos proporcionados pelas melhorias implementadas.

6.4 Validação por Incrementos

Cada fase foi analisada individualmente para medir o impacto de cada melhoria no desempenho do agente. As condições foram mantidas consistentes durante os testes, e o nível de profundidade da busca foi definido com base no tempo médio necessário para uma jogada. Isso permitiu identificar com clareza os avanços alcançados em termos de Elo, conforme descrito no capítulo anterior.

6.5 Tabela Comparativa das Medições da Primeira Etapa

A tabela 6.1 detalha as diferentes fases do desenvolvimento de um algoritmo de xadrez, indicando o progresso do desempenho através do incremento no **Elo**, que é uma medida de habilidade no jogo. A primeira coluna identifica cada fase do algoritmo, começando com a implementação do *Minimax puro* e avançando até a inclusão de recursos como *Poda alfa-beta*, *Ordenação de movimentos* e métodos de avaliação mais sofisticados, como *mobilidade*, *segurança do rei*, *domínio do centro* e *estrutura de peões*. A segunda coluna, denominada "Nível", indica a profundidade ou complexidade associada a cada fase do algoritmo. A terceira coluna apresenta o **incremento no Elo**, mostrando o ganho direto em desempenho em comparação com a fase anterior. A quarta coluna mostra o **Elo acumulado** após a implementação de cada fase, evidenciando a evolução do algoritmo em termos de força de jogo. Por fim, a última coluna destaca o **tempo médio por jogada** em segundos, que aumenta gradativamente à medida que o algoritmo se torna mais sofisticado, mas também mais computacionalmente exigente. O desenvolvimento culmina com a fase *Multithread*, que mantém o Elo final em **2089** e otimiza o tempo médio de processamento por jogada.

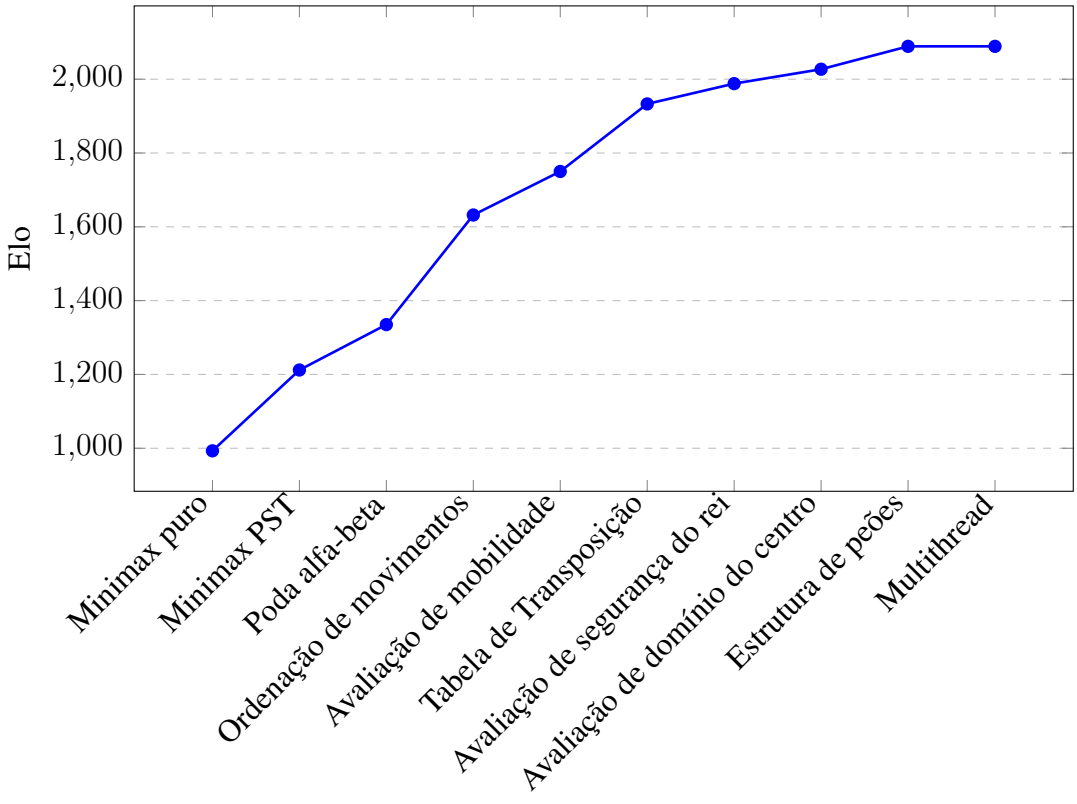
O gráfico 6.5 associado à tabela ilustra visualmente a **evolução do Elo** ao longo das fases do algoritmo de xadrez. No eixo **x (abcissa)**, estão representadas as diferentes fases do algoritmo, dispostas em sequência cronológica, desde o *Minimax puro* até a fase final de *Multithread*. O eixo **y (ordenada)** apresenta os valores do Elo, começando em

993 e alcançando **2089** conforme novos métodos são implementados. A curva resultante mostra um crescimento quase linear no Elo nas primeiras fases, como *Poda alfa-beta* e *Ordenação de movimentos*, seguidos de incrementos mais modestos em fases como *Avaliação de segurança do rei* e *Estrutura de peões*. O gráfico também destaca a estabilização do Elo na fase final (*Multithread*), onde o desempenho máximo é alcançado sem ganhos adicionais no Elo, embora o tempo médio por jogada tenha sido otimizado. Os pontos ao longo da curva são marcados com precisão, permitindo observar claramente o impacto de cada fase no aumento do desempenho do algoritmo.

Tabela 6.1: Incrementos de Elo e tempo médio por jogada para cada fase do algoritmo

Fase do algoritmo	**Nível**	**Incremento Elo**	**Elo**	**Tempo jogada(s)**
Minimax puro	5	0	993	15
Minimax PST	5	+219	1212	15
Poda alfa-beta	6	+123	1335	18
Ordenação de movimentos	8	+297	1632	23
Avaliação de mobilidade	9	+118	1750	30
Tabela de Transposição	10	+183	1933	24
Avaliação de segurança do rei	10	+55	1988	31
Avaliação de domínio do centro	10	+39	2027	38
Estrutura de peões	10	+62	2089	47
Multithread	10	0	2089	28

Evolução do Elo ao longo das fases do algoritmo



6.6 Incorporação das Técnicas Implementadas no Stockfish

A incorporação das técnicas do Stockfish no nosso agente foi realizada em três etapas principais, com o objetivo de adaptar metodologias amplamente testadas e eficazes para melhorar o desempenho da nossa implementação. Este capítulo descreve cada etapa, os desafios encontrados e o processo de validação utilizado para garantir a consistência com o motor Stockfish.

6.7 Etapas de Incorporação

A integração das técnicas do Stockfish seguiu um planejamento estruturado, conforme descrito abaixo:

6.7.1 Geração de Jogadas com Bitboards Mágicos

O primeiro passo foi implementar a técnica de **bitboards mágicos** para a geração de jogadas. Essa abordagem substituiu a representação anterior de tabuleiros e movimentos, resultando em uma geração de jogadas mais eficiente e compacta. O uso de bitboards mágicos trouxe um ganho significativo de desempenho na manipulação de dados relacionados ao estado do jogo, especialmente para movimentos de peças como torres, bispos e rainhas, que dependem de cálculos precisos de alcance em linhas e diagonais.

6.7.2 Função de Avaliação

Posteriormente, adaptamos a **função de avaliação** do Stockfish. Nossa implementação original da função de avaliação diferia substancialmente do modelo utilizado no agente de referência, o que limitava a eficácia de comparações diretas e validações. Optou-se, então, por importar a função de avaliação do Stockfish em sua totalidade, ajustando-a ao formato e às estruturas de dados da nosso motor de xadrez. Essa modificação trouxe maior precisão nas avaliações posicionais, incluindo fatores como estrutura de peões, mobilidade das peças e segurança do rei.

6.7.3 Adaptação do Minimax para a Poda Agressiva

Finalmente, adaptamos o **algoritmo minimax** para utilizar a **poda agressiva** implementada no Stockfish. Essa técnica permitiu reduzir drasticamente o número de posições avaliadas durante a busca, priorizando ramos promissores e descartando ramos menos relevantes de forma mais eficiente. A adaptação exigiu mudanças substanciais no gerenciamento de valores como alfa, beta e no uso da tabela de transposição.

6.8 Validação com Trace

Para validar a incorporação das técnicas, utilizamos o sistema de **tracing** do Stockfish. Esse sistema gera um log detalhado do processamento do agente, registrando informações como:

- Valores de alfa e beta.
- Avaliação do tabuleiro.
- Índices e valores da tabela de transposição.
- Estado de hashing.

Implementamos um sistema de tracing semelhante em nosso agente, garantindo que ambos seguissem o mesmo formato. Durante os testes, os dois agentes foram configuradas para analisar o mesmo tabuleiro em uma profundidade específica. O objetivo era garantir que os logs gerados fossem idênticos. Quando discrepâncias eram encontradas, iniciava-se um processo de **debug passo a passo** para identificar a origem do erro.

Esse ciclo de validação foi repetido centenas de vezes devido à complexidade do Stockfish. A linguagem **C++** introduziu desafios adicionais, como a necessidade de compreender e adaptar:

- **Sobrecarga de Operadores**: A habilidade do C++ de redefinir o comportamento de operadores para classes específicas tornou a interpretação de algumas partes do código menos intuitiva.
- **Referências e Ponteiros**: A gestão de referências e ponteiros no C++ adicionou complexidade ao rastreamento de valores e ao comportamento esperado durante o processamento.
- **Templates Avançados**: O uso extensivo de templates genéricos no Stockfish apre-

sentou desafios para adaptar o código de forma consistente com a nossa implementação.

6.9 Resultados e Limitações

Após a validação completa, obtivemos uma adaptação funcional de nosso agente, significativamente mais alinhada às técnicas do Stockfish. Entretanto, é importante destacar que nossa implementação não é uma tradução literal do agente de referência. Algumas funcionalidades do Stockfish foram omitidas, como:

- **Ponderação**: A capacidade de ponderar buscas enquanto o oponente ainda não realizou sua jogada.
- **Multithreading**: O suporte a múltiplos threads para acelerar as buscas.
- **Interrupção da Busca**: A habilidade de interromper o processamento do minimax e retornar o melhor resultado encontrado até o momento.

Tabela 6.2: Resultados das Técnicas Aplicadas

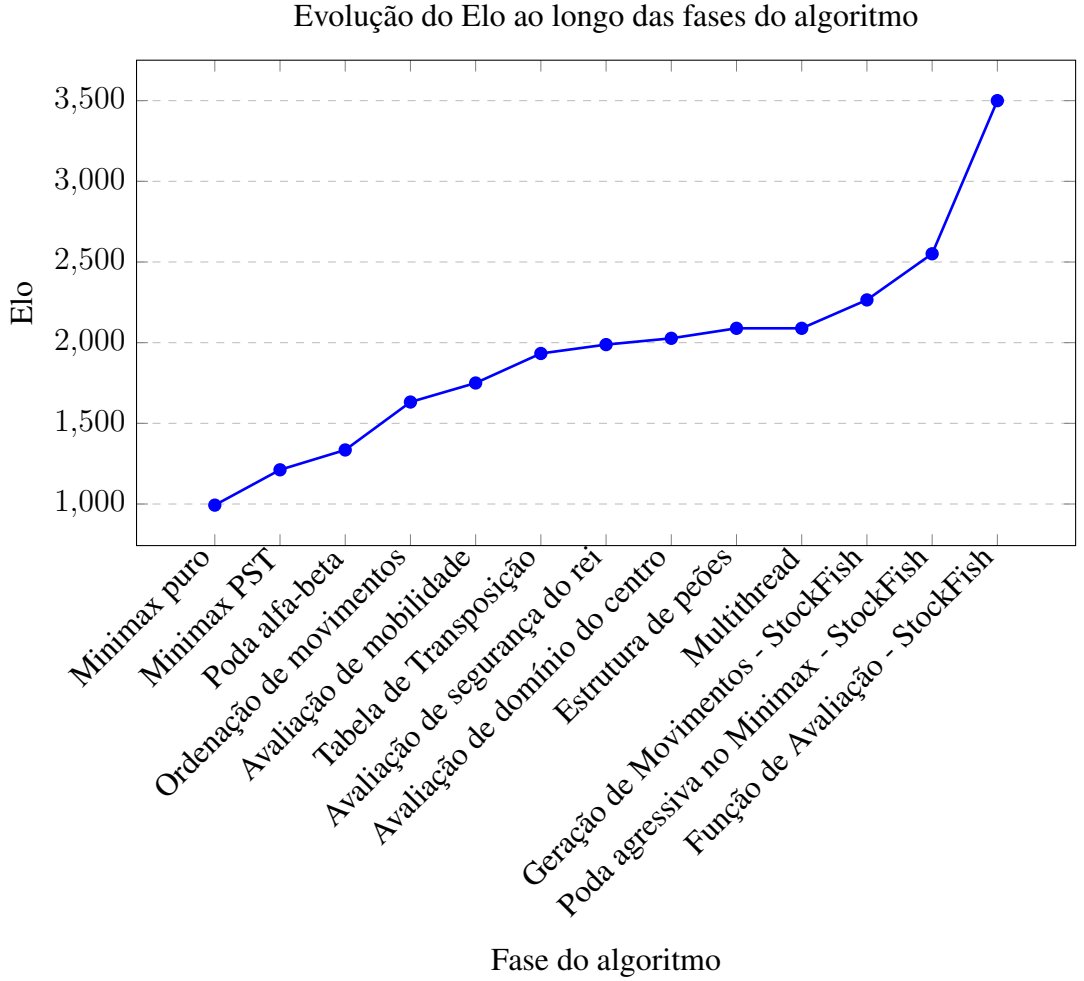
Fase do Algoritmo	Nível	Incremento Elo	Elo	Tempo jogada (s)
Geração de Movimentos	12	0	2265	36
Poda agressiva no Minimax	16	285	2551	42
Função de Avaliação	19	»949	»3500	58

Apesar dessas diferenças, a incorporação das técnicas principais do Stockfish permitiu uma melhoria significativa na eficiência e precisão da nosso agente. O processo de validação reforçou a robustez da adaptação e garantiu que o nosso agente mantivesse um alto nível de desempenho e confiabilidade.

O Elo máximo dos bots na plataforma **Chess.com** é 3200. Por essa razão, nosso agente foi testada com uma profundidade de busca no nível 13. Em um total de 100 partidas disputadas, o agente obteve **77 vitórias**, **9 empates** e **14 derrotas**, alcançando um desempenho equivalente a um **Elo de 3506**. O tempo médio por jogada em nosso equipamento foi de **8,3 segundos**. Quando nosso agente opera com niveis de profundidade superiores a 13, então ela sempre vence. Por isso, na tabela 6.2 a avaliação Elo e incremento foi estimado em alores muito maiores que 949 e 3500. Quando um jogador não perde nenhuma partida, o desempenho é considerado infinito na teoria do sistema Elo porque não há uma forma de calcular a razão W/L finita. Na prática, porém, sistemas de rating limitam esse desempenho a um valor extremamente elevado, mas não infinito. Algumas abordagens assumem que o jogador tem um desempenho "superior ilimitado"em

comparação ao oponente. O gráfico 6.9 apresenta a evolução geral do agente até a consolidação das técnicas assimiladas do motor Stockfish.

Figura 6.1: Evolução do Elo ao longo das fases do algoritmo com técnicas adicionais



6.10 Reflexão

Os testes demonstraram que o agente evoluiu progressivamente com cada melhoria implementada. Além disso, a escolha por realizar partidas contra bots de Elo fixo garantiu uma métrica consistente para avaliar a performance do agente. O cálculo do Elo permitiu medir quantitativamente o impacto das técnicas utilizadas, corroborando a eficácia das abordagens implementadas e servindo como base para estudos e otimizações futuras.

7 ADAPTAÇÃO DO AGENTE PARA JOGOS PRÉ-ENXADRISTICOS

Métodos ou jogos pré-enxadristicos refere-se a atividades, exercícios ou abordagens pedagógicas utilizadas para preparar iniciantes no xadrez, especialmente crianças ou pessoas que ainda não têm familiaridade com as regras e conceitos básicos do jogo. Esses métodos ajudam a desenvolver habilidades cognitivas, motoras e estratégicas antes mesmo de ensinar o jogo em si.

Ensinar xadrez para crianças é uma atividade de grande relevância no desenvolvimento cognitivo. O jogo de xadrez é amplamente reconhecido como uma ferramenta eficaz para aprimorar o raciocínio lógico, a tomada de decisões, a concentração e até mesmo a criatividade. Além disso, o xadrez promove habilidades sociais importantes, como a paciência, o respeito às regras e a resiliência ao lidar com vitórias e derrotas. Sua aplicação no ambiente pedagógico tem mostrado resultados positivos em diversas faixas etárias, especialmente na educação infantil e fundamental. Contudo, o ensino tradicional do xadrez pode apresentar desafios para crianças, pois a complexidade inicial do jogo, com suas 32 peças e movimentos variados, pode ser intimidadora ou levar à dispersão.

Um problema comum é que as crianças, por vezes, se encantam por peças específicas, como o cavalo, com seu movimento peculiar em "L", ou a torre, pela sua habilidade de atravessar o tabuleiro em linha reta. Essa atração, embora natural, pode distrair do objetivo principal do jogo e levar a um comportamento repetitivo, como o uso excessivo de uma única peça, em detrimento do aprendizado estratégico e global do jogo. Assim, há uma necessidade de adaptar a experiência de ensino para tornar o xadrez mais acessível e engajador para crianças.

Nosso agente, ao contrário de agentes tradicionais como o Stockfish, oferece a flexibilidade necessária para atender a essas demandas pedagógicas tal como o método que descreveremos nas próximas seções. Com pequenas adaptações em nosso agente, podemos criar um ambiente de aprendizado lúdico e progressivo, no qual as crianças possam interagir com um xadrez simplificado, adequado ao seu nível de compreensão e atenção.

7.1 Tabuleiro Reduzido e Regras Simplificadas

Uma das maiores inovações possíveis com nosso agente é a capacidade de suportar configurações de tabuleiros reduzidos, com menos linhas e colunas, além de permitir

partidas com um número limitado de peças. Agentes tradicionais, como o Stockfish, dependem da presença de peças como o rei e não conseguem avaliar tabuleiros que fogem ao formato padrão de 8x8. Em contrapartida, nosso agente pode ser facilmente adaptado para lidar com diferentes configurações, permitindo uma abordagem escalonada no ensino do xadrez.

Por exemplo, uma atividade inicial pode ser a **batalha de peões**. Nesse jogo, o tabuleiro é reduzido (como 4x4 ou 6x6), e as únicas peças presentes são os peões. O objetivo é simples: promover um peão ao final do tabuleiro adversário. Esse formato introduz as crianças às regras básicas do movimento e captura dos peões, sem sobrecarregá-las com a complexidade das demais peças. À medida que a criança progride, pode-se aumentar o número de peões ou ampliar o tamanho do tabuleiro.

7.2 Peças Auxiliares e Estratégias Avançadas

Outra adaptação possível é a introdução de **peças auxiliares** que servem como "ajudantes"para os peões. Essas peças especiais poderiam incluir, por exemplo:

- **O Arqueiro**: Uma peça com movimentos limitados em linha reta, capaz de "atirar"em peças adversárias a uma distância curta.
- **O Guardião**: Uma peça que se move como um rei, mas só pode proteger peões, bloqueando capturas de adversários.

Essas peças não podem vencer o jogo por si só, pois o objetivo permanece ser a promoção de um peão. Contudo, sua inclusão adiciona camadas estratégicas e mantém o interesse da criança, ao mesmo tempo em que introduz conceitos mais avançados, como proteção e controle do tabuleiro.

7.3 Interação Entre Crianças e Contra o Agente

Nosso agente pode ser configurada para partidas entre crianças ou contra o próprio agente. Essa funcionalidade é essencial em ambientes pedagógicos, pois permite que os alunos aprendam no próprio ritmo. No modo **Player vs Player**, as crianças podem competir entre si em partidas rápidas e divertidas, enquanto no modo **Player vs Agente**, a dificuldade pode ser ajustada automaticamente para desafiar as habilidades do jogador

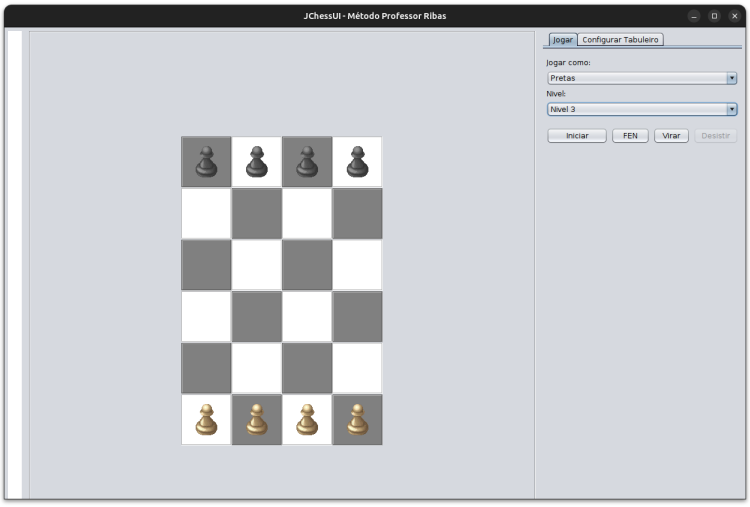
sem desmotivá-lo.

Além disso, nosso agente pode registrar o desempenho de cada jogador, fornecendo relatórios sobre áreas de força e aspectos que precisam ser melhorados. Isso oferece uma ferramenta valiosa para educadores acompanharem o progresso de seus alunos de forma personalizada.

7.4 Interface Adaptada

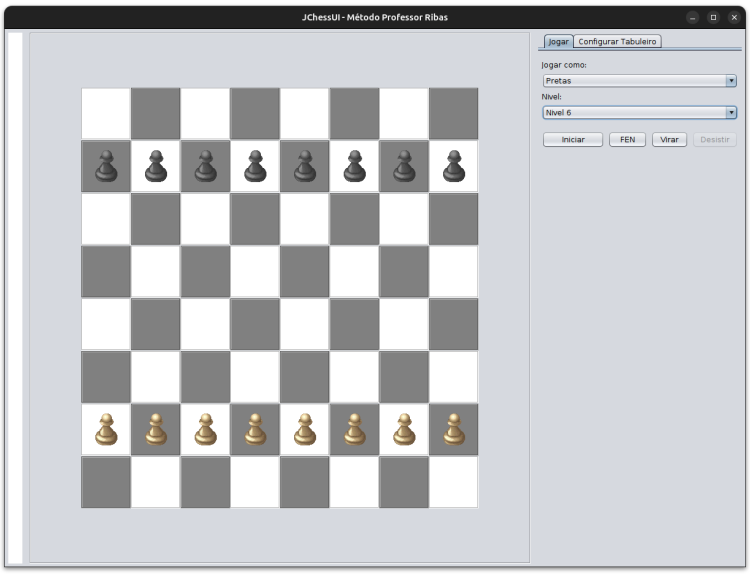
A simplificação reduz a quantidade de peças e regras em jogo, tornando mais fácil para os jovens compreenderem conceitos fundamentais, como controle de colunas e bloqueios. Essa abordagem cria um ambiente mais acessível e menos intimidador, onde as crianças podem focar em desenvolver habilidades específicas de planejamento e tomada de decisão sem se sentirem sobrecarregadas pela complexidade do xadrez completo.

Figura 7.1: Interface Gráfica Adaptada: tabuleiro reduzido.



Fonte: O autor

Figura 7.2: Interface gráfica adaptada: tabuleiro 8x8.



Fonte: O autor

A utilização de um agente para simular jogadas permite que os pequenos aprendam de forma interativa, enfrentando um "oponente"que pode ser ajustado em diferentes níveis de dificuldade, incentivando o progresso gradual e promovendo o engajamento por meio de desafios adequados à sua habilidade. As Figuras 7.1 e 7.2 apresentam um protótipo de interface modificada para um método do pedagógico, demonstrando a evolução de jogos (desafios) pré-enxadrísticos em relação ao tamanho do tabuleiro e número de peças.

7.5 Benefícios Pedagógicos

Essa abordagem oferece inúmeros benefícios:

- **Aprendizado Escalonado**: Ao começar com regras simples e tabuleiros reduzidos, as crianças podem se familiarizar com o jogo de forma gradual e sem pressão.
- **Desenvolvimento Cognitivo**: Os formatos adaptados mantêm o foco em habilidades importantes, como planejamento, antecipação e resolução de problemas.
- **Engajamento Lúdico**: O uso de peças auxiliares e desafios progressivos mantém o interesse das crianças ao longo do aprendizado.
- **Inclusão Social**: A simplicidade das regras adaptadas permite que crianças de diferentes idades e níveis de habilidade joguem juntas, promovendo a colaboração e a inclusão.

7.6 Reflexão

A capacidade de adaptação de nosso agente para projetos pedagógicos é um diferencial que a destaca no mercado. Ao permitir jogos simplificados, personalizáveis e voltados para o aprendizado, oferecemos uma ferramenta poderosa para escolas, clubes de xadrez e iniciativas sociais. Essa flexibilidade é um reflexo direto do design modular e inovador da nossa implementação, que prioriza não apenas a performance, mas também a acessibilidade.

O xadrez é mais do que um jogo; é uma oportunidade de transformar vidas. Com nosso agente, podemos levar esse impacto a mais crianças, ajudando a formar não apenas melhores jogadores, mas também indivíduos mais concentrados, criativos e resilientes.

8 CONCLUSÃO

O presente trabalho abordou o desenvolvimento de um agente de xadrez, explorando as técnicas e heurísticas computacionais aplicadas ao jogo, assim como os benefícios do xadrez em contextos educacionais e sociais. O estudo permitiu compreender tanto a complexidade algorítmica envolvida na implementação de um programa que simula partidas de xadrez, quanto a relevância do jogo em outras áreas, como a pedagogia e a interação social.

Do ponto de vista técnico, foi possível explorar e implementar algoritmos como o *minimax* com poda alfa-beta, heurísticas de avaliação posicional e conceitos avançados, como tabelas de transposição com *Zobrist Hashing* e pesquisas de quiescência. Essas técnicas não apenas permitiram o desenvolvimento de um agente funcional, mas também evidenciaram o desafio de encontrar soluções computacionalmente eficientes para um jogo com tamanha complexidade combinatória. Além disso, o aprofundamento nesses métodos representa uma excelente oportunidade de aprendizado em áreas como inteligência artificial e desenvolvimento de software.

No aspecto pedagógico, o xadrez mostrou-se uma poderosa ferramenta para o desenvolvimento de habilidades cognitivas em crianças, como o raciocínio lógico, a concentração e a tomada de decisões. A implementação de uma ferramenta digital baseada no xadrez, voltada para o ensino em escolas, pode potencializar ainda mais esses benefícios, democratizando o acesso ao aprendizado do jogo e tornando-o mais interativo e atraente para crianças. Assim, o xadrez não é apenas um jogo, mas um recurso didático que pode contribuir diretamente para o desenvolvimento intelectual de jovens estudantes.

Além disso, o estudo reforçou a importância do xadrez como instrumento sociocultural, promovendo a interação entre pessoas de diferentes idades, origens e contextos sociais. Em um mundo cada vez mais digitalizado, o xadrez mantém seu papel de integração, proporcionando um espaço de socialização e aprendizado mútuo, seja em torneios presenciais, ambientes escolares ou plataformas virtuais.

As perspectivas futuras deste trabalho incluem o aprofundamento nas técnicas e heurísticas envolvidas no desenvolvimento de agente de xadrez, visando aprimorar a eficiência dos algoritmos e explorar novas abordagens de inteligência artificial, como o aprendizado por reforço. Além disso, pretende-se desenvolver uma ferramenta pedagógica voltada especificamente ao ensino de xadrez para crianças, com o objetivo de promover o desenvolvimento do raciocínio lógico e habilidades correlatas no ambiente escolar. Essa

aplicação poderá contribuir significativamente para a educação básica, oferecendo uma abordagem inovadora e acessível para o aprendizado de conteúdos interdisciplinares.

Por fim, o estudo realizado evidencia que o xadrez vai muito além de suas origens históricas como um jogo de tabuleiro: ele representa um recurso técnico, pedagógico e social de grande relevância, com o potencial de transformar a forma como aprendemos, interagimos e pensamos.

APÊNDICE A — TEORIA DO XADREZ

A.1 O Desenho do Jogo

O visual das peças de xadrez sempre oscilou entre o simples e o ornamentado desde os tempos do Chaturanga (Figura A.1). Antes de 600 d.C., o design das peças começou a evoluir passando de formas básicas para conjuntos que representavam animais, guerreiros e nobres (MURRAY, 1913). Entre os séculos IX e XII, os muçulmanos adotaram um estilo mais simples e abstrato, frequentemente usando barro ou pedra esculpida. Tinham o cuidad de não representar formas de criaturas vivas, em respeito à proibição islâmica (HOOPER; WHYLD; WHYLD, 1987). Esse retorno ao estilo mais simbólico do Shatranj popularizou do jogo, já que peças menos detalhadas eram mais fáceis de produzir, e os jogadores podiam se concentrar na estratégia, em vez de na complexidade das peças (FISKE, 1913).

Quando o xadrez se espalhou pela Europa e Rússia, peças mais elaboradas, muitas vezes decoradas com pedras preciosas e semipreciosas, voltaram a fazer sucesso (EALES, 1993). Enquanto isso, os tabuleiros, que no mundo muçulmano usavam quadrados monocromáticos, ganharam padrões alternados de preto e branco, ou vermelho e branco, por volta de 1000 d.C., sendo feitos de materiais nobres como madeira ou mármore (YALOM, 2009). Já no século XVIII, Pedro I, o Grande, da Rússia, usava tabuleiros de campanha feitos de couro macio, que ele carregava em suas expedições militares. O design moderno das peças de xadrez foi criado por Nathaniel Cook em 1835 (SOLTIS, 2024), e, com a promoção de Howard Staunton, em 1849, esse modelo — conhecido como "Staunton"— se tornou o padrão oficial para competições internacionais (LINDER, 1994).

Figura A.1: Chaturanga

Fonte: (LUDOTEKA, 2024)

A.2 As Regras

O xadrez, como o conhecemos hoje, passou por uma longa transformação ao longo dos séculos, adaptando suas regras e peças para tornar o jogo mais dinâmico e estratégico. Desde as suas origens em versões antigas como o Shatranj, até a padronização das regras modernas, o xadrez evoluiu significativamente em termos de mecânica e complexidade. Essa evolução não só definiu a estrutura do jogo atual, mas também influenciou o modo como ele é jogado e estudado em todo o mundo.

A.2.1 A Evolução das Regras e Peças

O xadrez, em sua forma moderna, passou por um longo processo de evolução nas regras e na aparência das peças, com variações significativas em diferentes regiões. Por exemplo, em 1300, o peão ganhou a capacidade de mover-se duas casas em seu primeiro lance, em vez de apenas uma, como era no shatranj, a antiga versão do xadrez (MURRAY, 1913). No entanto, essa regra só foi amplamente aceita na Europa após mais de 300 anos (EALES, 1993). Até o século XV, o conselheiro (atual dama) movia-se apenas uma casa na diagonal e, mesmo quando um peão era promovido ao atingir a oitava fileira, ele só poderia se transformar em conselheiro, tornando a promoção um fator pouco decisivo no jogo (YALOM, 2009).

O maior progresso do xadrez ocorreu após duas mudanças cruciais nas regras, que se popularizaram a partir de 1475. O conselheiro foi transformado na poderosa dama, ganhando a capacidade de movimentar-se em todas as direções por várias casas, tornando-se a peça mais poderosa do tabuleiro (EALES, 1993). A peça chamada elefante, que anteriormente movia-se em um salto de duas casas na diagonal, foi transformada no bispo, dobrando seu alcance (DAVIDSON, 1949). Essas mudanças deram um novo dinamismo ao jogo, aumentando o valor da promoção de peão e permitindo checkmates em poucos movimentos, substituindo o antigo estilo de jogo que muitas vezes terminava com o rei desprotegido.

A.2.2 Regras Atuais

As regras modernas do xadrez continuaram a se consolidar com as adições do roque e da captura en passant. Embora essas regras fossem conhecidas desde o século XV, elas só foram amplamente aceitas a partir do século XVIII (MURRAY, 1913). Pequenas variações em outras regras persistiram até o final do século XIX; por exemplo, em algumas regiões da Europa, não era permitido promover um peão a dama se o jogador ainda possuía sua dama original (EALES, 1993).

O xadrez atual é jogado em um tabuleiro de 64 casas, com 16 peças para cada jogador, incluindo um rei, uma dama, duas torres, dois bispos, dois cavalos e oito peões (DAVIDSON, 1949). O objetivo do jogo é colocar o rei adversário em xeque-mate, uma posição em que ele está sob ataque e não pode escapar. O tempo de cada partida é controlado por um relógio, que determina um limite para cada jogador completar suas jogadas. A notação algébrica é usada para registrar os movimentos das peças, e as regras de empate, como o afogamento e a regra das cinquenta jogadas, garantem a uniformidade das competições (FIDE, 2021). A padronização moderna das regras e o uso do modelo Staunton para as peças são elementos que asseguram a consistência e a integridade das partidas em torneios oficiais ao redor do mundo (STAUNTON, 1849).

A.2.3 O Enfoque Prático

Após 1970, o pensamento sobre o xadrez passou a valorizar uma abordagem mais prática durante as competições. Os soviéticos defendiam que, ao criar desequilíbrios nas posições, forçavam os jogadores a encontrar os melhores lances, pois, em situações críticas, qualquer erro poderia inverter uma vantagem ou levar à perda da iniciativa (DVORETSKY, 2001). Esse enfoque se tornou evidente nas partidas de Fischer e Karpov, que acreditavam que as partidas eram decididas mais por erros sob pressão de tempo do que por jogadas brilhantes (KASPAROV, 2020). Assim, enquanto os soviéticos se concentravam no perfeccionismo durante o meio-jogo, os pragmatistas começaram a dar mais importância ao final da partida (SOLTIS; SOLTIS, 2014).

Entre as décadas de 1970 e 1990, houve mudanças no entendimento sobre as fases iniciais e intermediárias do jogo, como a desvalorização do bispo em comparação ao cavalo. Enquanto os jogadores do período pós-soviético trocavam bispos por cavalos por compensações mínimas, também não hesitavam em manter um bispo ruim, trocando o

bom, em busca de pequenos ganhos posicionais (SOLTIS; SOLTIS, 2014). Essas alterações no pensamento refletem um afastamento das ideias tradicionais e uma adaptação a um estilo mais pragmático e flexível.

A.2.4 Teoria do Ataque

No final do século XVIII, mestres italianos de Módena desafiaram as ideias de Philidor, que focava excessivamente na importância dos peões e negligenciava a eficácia de ataques diretos ao rei adversário. Eles mostraram que partidas poderiam ser vencidas em menos de 20 movimentos com uma rápida mobilização das peças, contrastando com o estilo mais lento e posicional de Philidor. As ideias desse grupo italiano levaram à popularização dos gambitos, sacrifícios de peões nas aberturas para obter rápida mobilização e abrir linhas para ataque. Esse estilo agressivo e com sacrifícios caracterizou muitos mestres do século XIX e ficou conhecido como escola Romântica (SHIBUT, 2012).

Paul Morphy, o primeiro americano reconhecido como o melhor jogador de xadrez do mundo, exemplificou essas ideias de ataque, apesar de sua breve carreira de menos de três anos e menos de 75 jogos sérios. Em 1858-1859, ele derrotou todos os principais jogadores europeus, exceto Howard Staunton. Sua habilidade em atacar no momento certo permitiu-lhe evitar derrotas em posições inferiores e vencer posições favoráveis. Sua superioridade em calcular combinações e compreender a iniciativa tática o diferenciava dos outros mestres da época. Após derrotar Adolf Anderssen por 7–2, Anderssen declarou que Morphy não lhe dava oportunidades para brilhantes sacrifícios, como ocorria com outros mestres (SHIBUT, 2012).

Morphy valorizava o desenvolvimento rápido das peças em posições abertas, sacrificando frequentemente peões para ativar suas peças mais poderosas, como rainhas, torres e bispos. Ele demonstrou que a superioridade de desenvolvimento nas primeiras 10 a 15 jogadas era crucial em posições abertas, onde a iniciativa poderia ser decisiva. Sua capacidade de manter e concretizar essa iniciativa foi a chave para sua grandeza e influência duradoura no jogo moderno (SHIBUT, 2012).

A Teoria do Ataque no xadrez enfatiza a importância de ações agressivas para pressionar o adversário, controlando o ritmo e forçando erros. Estrategicamente, é fundamental identificar fraquezas na posição inimiga e coordenar peças de maneira a maximizar a pressão. Os principais conceitos envolvem sacrifícios táticos, controle de colunas abertas e a criação de ameaças múltiplas. Jogadores como Garry Kasparov são conhecidos

por seu domínio em conduzir ataques precisos e devastadores, demonstrando a eficácia desta abordagem para desmantelar defesas bem estruturadas(SOLTIS, 2024).

A.2.5 Teoria do Equilíbrio

A Teoria do Equilíbrio, por outro lado, busca a manutenção de uma posição sólida e resiliente, onde não há vulnerabilidades óbvias que o oponente possa explorar. Esta abordagem é ideal para jogadores que preferem uma partida mais estratégica e menos arriscada, focando em manter a simetria posicional e trocar peças de forma a neutralizar possíveis ataques adversários. Grandes mestres como Anatoly Karpov ilustram a eficácia desta filosofia de jogo, priorizando o controle do centro e a flexibilidade posicional para capitalizar em erros do adversário ao invés de arriscar ataques prematuros (LINDER; LINDER, 2014).

Wilhelm Steinitz, o sucessor de Morphy, foi campeão mundial até 1894, desenvolvendo princípios do meio-jogo que influenciaram profundamente o xadrez posicional. Steinitz afirmou que o resultado natural de uma partida é o empate, devido ao equilíbrio inerente entre as forças de Brancas e Pretas, e que o objetivo final do jogo é o xeque-mate, mas não necessariamente o primeiro (SHIBUT, 2012). No início de sua carreira, Steinitz era um jogador tático e combinacional, no estilo de Morphy, mas, ao atingir a maturidade, ele começou a valorizar características posicionais sutis, especialmente em estruturas de peões bloqueadas. Ele procurou explicar por que alguns ataques eram bem-sucedidos enquanto outros falhavam, independentemente da habilidade dos jogadores, e concluiu que um ataque só era justificado quando o equilíbrio de forças era rompido (KEENE, 1999).

Steinitz observou que pequenas vantagens, como o controle de casas críticas ou a melhor estrutura de peões, poderiam justificar um ataque. Exemplos incluem ter dois bispos contra dois cavalos do oponente ou ter um cavalo bem posicionado em uma casa sem proteção de peões adversários. Ele foi o primeiro a usar o termo "buraco"para designar casas vulneráveis no campo do adversário (Coombe, 2020). Steinitz também popularizou a importância de explorar fraquezas estruturais nos peões, como peões isolados ou dobrados. Essas fraquezas poderiam ser exploradas no final do jogo, onde vantagens sutis, como maioria de peões em um flanco, podiam ser decisivas (LINDER; LINDER, 2014). Além disso, ele introduziu o conceito de "ataque minoritário", onde o jogador com menos peões em um flanco avança para criar fraquezas na estrutura do oponente.

Steinitz foi um pioneiro na defesa, muitas vezes assumindo posições passivas com

as peças brancas para atrair ataques prematuros do adversário, que ele punia com contragolpes pacientes. Sua abordagem científica ao xadrez, equilibrando defesa e ataque com base no entendimento posicional, o diferenciou dos românticos anteriores. Ele consistentemente derrotou os principais jogadores da época, como Adolf Anderssen e Mikhail Chigorin, consolidando seu legado como o primeiro jogador a tratar o xadrez de maneira científica (SHIBUT, 2012). Steinitz combinou teoria e prática em suas partidas e escritos, deixando uma herança duradoura para o desenvolvimento do jogo moderno(SOLTIS, 2024).

A.2.6 Xadrez 960

Xadrez 960, também conhecido como Fischer Random Chess, é uma variante do xadrez tradicional criada por Bobby Fischer em 1996. Seu principal objetivo é reduzir a importância da memorização de aberturas e aumentar a criatividade e o pensamento estratégico desde os primeiros lances. No Xadrez 960, a disposição das peças na primeira fileira de cada jogador é aleatória, resultando em 960 possíveis configurações iniciais, daí o nome da modalidade. Com isso, os jogadores são forçados a depender mais de sua habilidade de adaptação e cálculo tático, pois não podem confiar em padrões estabelecidos de jogo. Essa inovação serve como um antídoto para o crescente foco em preparação teórica, comum no xadrez tradicional de alto nível (FIDE, 2021).

As regras do Xadrez 960 seguem as mesmas do xadrez clássico, com exceção da disposição inicial das peças. Cada jogador começa com uma fileira de peões na segunda linha e as peças maiores (torres, cavaleiros, bispos, dama e rei) distribuídas aleatoriamente na primeira linha. A única restrição para a distribuição das peças é que os bispos devem estar em casas de cores opostas, e o rei deve estar posicionado entre as duas torres para possibilitar o roque. O roque funciona de maneira semelhante ao xadrez tradicional, mas depende da posição inicial do rei e das torres. Todas as outras regras — como o xeque, o xeque-mate e a promoção de peões — permanecem inalteradas. A aleatoriedade da posição inicial elimina o estudo de linhas de abertura específicas e promove uma maior ênfase na compreensão tática e estratégica (FIDE, 2023).

A.2.7 Outras Notações de Xadrez

Além das notações padrão, há outras notações específicas ou historicamente relevantes:

A.2.7.1 Notação Descritiva

A notação descritiva era comumente utilizada antes da notação algébrica se tornar o padrão. Ela se baseia em nomes das colunas e fileiras a partir da perspectiva de cada jogador. Em vez de letras, as colunas eram nomeadas de acordo com as peças na posição inicial, como `QR` (torre da dama), `QN` (cavalo da dama) e assim por diante. Essa notação foi amplamente substituída pela notação algébrica, que é mais objetiva.

A.2.7.2 Notação ICCF

A notação ICCF, usada em competições de xadrez por correspondência, é uma versão modificada da notação algébrica, onde as jogadas são indicadas por coordenadas das casas de origem e destino, sem o uso de letras para as peças. Essa notação é especialmente útil para partidas realizadas sem um tabuleiro físico, como em torneios de longa distância ou via correspondência eletrônica.

APÊNDICE B — XADREZ COMO ESPORTE

B.1 Torneios

Os torneios de xadrez modernos evoluíram consideravelmente desde o início do século XX, acompanhando o crescimento do xadrez como esporte e sua profissionalização. Com o advento de sistemas de classificação, como o Elo Rating System, desenvolvido por Arpad Elo em 1960, os torneios passaram a ter uma estrutura mais organizada e competitiva (GLICKMAN, 1995). Esses sistemas não só permitiram uma melhor organização dos torneios, mas também proporcionaram uma maneira objetiva de medir a força dos jogadores, facilitando o estabelecimento de categorias e títulos como Mestre Internacional e Grande Mestre. A profissionalização do xadrez também levou ao surgimento de circuitos internacionais, como o Grand Chess Tour, que reúne a elite mundial em competições de alto nível (BLOKDYK, 2020).

A influência dos torneios online e híbridos também se tornou um aspecto significativo dos torneios modernos. A pandemia de COVID-19 acelerou a popularidade dos torneios online, oferecendo uma plataforma para a continuidade das competições em um momento em que torneios presenciais eram inviáveis (ILIESCU, 2020). O advento das tecnologias digitais e plataformas como *chess.com* e *lichess*, jogadores de diferentes partes do mundo podem competir entre si sem as barreiras logísticas de viagens, e isso ampliou a acessibilidade ao xadrez competitivo. Torneios híbridos, que combinam partidas presenciais e online, também surgiram como um novo formato, permitindo maior flexibilidade e inclusão no cenário competitivo global.

B.1.1 FIDE

A FIDE, ou Federação Internacional de Xadrez, é a entidade responsável pela regulamentação e promoção do xadrez em nível mundial. Fundada em 1924, a FIDE estabelece as regras do jogo, organiza torneios internacionais, e confere títulos como Grande Mestre (GM) e Mestre Internacional (IM) aos jogadores que atendem a critérios específicos de desempenho. A FIDE também é responsável pela classificação Elo, que mede a força relativa dos jogadores em competições(FIDE, 2024).

A organização atua em várias frentes, incluindo a promoção do xadrez nas escolas, a organização de eventos de alto nível como o Campeonato Mundial e a supervisão das

federações nacionais. Em anos recentes, a FIDE tem se concentrado em expandir a base de jogadores e tornar o xadrez mais acessível globalmente, utilizando plataformas online e promovendo o jogo entre as novas gerações.

B.1.2 O Sistema de Ranking Elo

O sistema de ranking Elo é um método estatístico amplamente utilizado para calcular a força relativa entre jogadores de xadrez. Desenvolvido pelo físico húngaro Arpad Elo, esse sistema é a base para a classificação de jogadores em diversas plataformas e federações de xadrez ao redor do mundo, incluindo a Federação Internacional de Xadrez (FIDE, 2024).

B.1.2.1 Funcionamento

O sistema Elo atribui a cada jogador um número, chamado rating, que representa sua força de jogo. Esse rating é dinâmico e se ajusta a cada partida disputada. A ideia central é que a probabilidade de um jogador vencer outro é uma função direta da diferença entre seus ratings (ELO, 1978).

B.1.2.2 Fórmula

A fórmula do Elo, embora possa parecer complexa à primeira vista, é uma ferramenta poderosa para calcular a força relativa de jogadores em diversos jogos, principalmente no xadrez. Ela permite que jogadores sejam comparados de forma objetiva e que seus ratings sejam ajustados após cada partida, refletindo com precisão suas habilidades. A fórmula básica para calcular a nova classificação (Rn) de um jogador após uma partida é a seguinte:

$$R_{\text{novo}} = R_{\text{antigo}} + K \cdot (S - E)$$

Onde:

- R_{novo} = Novo rating
- R_{antigo} = Rating inicial (1613)
- K = Fator K (determina o peso de cada partida; valores comuns são 32 ou 40)
- S = Pontuação real
- E = Pontuação esperada

O fator *K* determina a quantidade de pontos que um jogador pode ganhar ou perder em uma partida. Ele é geralmente maior para jogadores com menos partidas disputadas e menor para jogadores mais experientes. O *W* Representa o resultado da partida de forma simples: vitória, empate ou derrota. A pontuação esperada *E* é o ponto-chave do sistema Elo. Ela indica a probabilidade de um jogador vencer uma partida, com base na diferença de rating entre os dois jogadores.

B.1.2.3 Pontuação Esperada

Antes de uma partida, o sistema calcula a pontuação esperada (E) de cada jogador. Essa pontuação representa a probabilidade de um jogador vencer, somada à metade da probabilidade de um empate. Após a partida, o rating de cada jogador é atualizado com base no resultado real e na pontuação esperada. Se um jogador com rating inferior vence um jogador com rating superior, seu rating aumentará significativamente, enquanto o rating do jogador mais forte diminuirá. A fórmula para calcular a pontuação esperada é mais complexa:

$$E = \frac{1}{1 + 10^{(R_a - R_b)/400}}$$

Onde:

- E = Pontuação esperada
- R_a = Rating do oponente A
- R_b = Rating do oponente B

B.1.2.4 Exemplo

Considere um cenário onde o Jogador A, com um rating inicial de 1613, participa de um torneio de cinco rodadas (CONTRIBUTORS, 2024d). Os resultados das partidas são os seguintes:

- **Rodada 1:** Derrota para um jogador com rating 1609.
- **Rodada 2:** Empate com um jogador com rating 1477.
- **Rodada 3:** Vitória contra um jogador com rating 1388.
- **Rodada 4:** Vitória contra um jogador com rating 1586.
- **Rodada 5:** Derrota para um jogador com rating 1720.

Para calcular o novo rating do Jogador A, utilizamos a fórmula do sistema de rating Elo descrita anteriormente:

$$R_{\text{novo}} = R_{\text{antigo}} + K \cdot (S - E)$$

Calculando a Pontuação Real (S):

A pontuação real do Jogador A é a soma dos pontos obtidos em cada partida (onde o valor *1* representa vitória, *0* derrota e *0,5* empate):

$$S = 0 + 0,5 + 1 + 1 + 0 = 2,5$$

Calculando a Pontuação Esperada (E):

A pontuação esperada é determinada com base no rating dos oponentes e no rating inicial do jogador. A fórmula para calcular a pontuação esperada contra um único oponente é:

$$E_i = \frac{1}{1 + 10^{(R_i - R_{\text{A}})/400}}$$

Onde:

- E_i = Pontuação esperada contra o oponente i
- R_i = Rating do oponente i
- R_{A} = Rating do jogador A

Aplicando esta fórmula a cada partida:

$$\begin{aligned}
E_1 &= \frac{1}{1 + 10^{(1609-1613)/400}} \approx 0,506 \\
E_2 &= \frac{1}{1 + 10^{(1477-1613)/400}} \approx 0,686 \\
E_3 &= \frac{1}{1 + 10^{(1388-1613)/400}} \approx 0,785 \\
E_4 &= \frac{1}{1 + 10^{(1586-1613)/400}} \approx 0,539 \\
E_5 &= \frac{1}{1 + 10^{(1720-1613)/400}} \approx 0,351
\end{aligned}$$

Somando esses valores:

$$E = E_1 + E_2 + E_3 + E_4 + E_5 \approx 0,506 + 0,686 + 0,785 + 0,539 + 0,351 = 2,867$$

Determinando o Fator K (K):

O fator K determina o peso de cada partida na atualização do rating. Um valor comum para o K é 32 (CONTRIBUTORS, 2024d).

Cálculo Final:

Substituindo os valores na fórmula do Elo:

$$R_{\text{novo}} = 1613 + 32 \cdot (2,5 - 2,867) = 1613 + 32 \cdot (-0,367) = 1613 - 11,744 \approx 1601$$

Portanto, o novo rating do Jogador A após o torneio será aproximadamente 1601.

Este exemplo ilustra como o rating de um jogador pode diminuir, mesmo com duas vitórias e um empate, devido ao nível dos adversários enfrentados.

B.1.2.5 Vantagens do sistema Elo

O sistema Elo oferece uma forma objetiva de comparar jogadores de diferentes níveis e em diferentes momentos. Ao longo do tempo, o sistema tende a fornecer uma classificação precisa dos jogadores, refletindo suas habilidades reais. O sistema também pode ser adaptado para outros jogos e esportes, além do xadrez (ELO, 1978).

B.1.2.6 Limitações

Jogadores com poucos jogos disputados podem ter seus ratings mais voláteis. Portanto o rating Elo do jogador deve ser seguido de um longo histórico para ser mais preciso que também ajuda a eliminar o fator "sorte"que pode ocorrer eventualmente.

B.1.2.7 Aplicações do sistema Elo

O sistema Elo é o padrão para classificar jogadores de xadrez em todo o mundo. Também é utilizado em jogos como Go, Shogi e videogames competitivos. Algumas modalidades esportivas, como tênis e futebol, utilizam sistemas de ranking inspirados no Elo.

B.1.3 O Fator Tempo nos Torneios

O fator tempo desempenha um papel crucial nas competições e torneios de xadrez, influenciando tanto a dinâmica do jogo quanto as estratégias dos jogadores. Existem diversos controles de tempo, como partidas clássicas, rápidas e blitz, que afetam a preparação e o estilo de jogo dos participantes. Em partidas com controle de tempo reduzido, como o blitz, a pressão para tomar decisões rápidas muitas vezes leva a erros táticos, tornando o jogo mais imprevisível e emocionante (HARREVELD; WAGENMAKERS; MAAS, 2007). Além disso, o tempo disponível impacta diretamente a capacidade de planejamento estratégico, forçando os jogadores a equilibrarem velocidade e precisão(JOHNSON, 2024).

Em 1988, Bobby Fischer propôs um incremento incondicional por movimento, não importando se o atraso foi esgotado ou não. Com o tempo de Fischer, pode-se, portanto, aumentar o tempo restante se alguém se mover mais rápido do que o atraso (CONTRIBUTORS, 2024b). Com o advento dos relógios digitais, o incremento de tempo após cada jogada, conhecido como "delay"ou "increment", tornou-se comum, permitindo que os jogadores não percam apenas por tempo em posições ganhas, o que traz uma dimensão adicional de complexidade ao gerenciamento do relógio durante o jogo.

B.1.4 Hierarquia e Ranking

A hierarquia no xadrez moderno é definida principalmente por um sistema de classificações, que categoriza jogadores com base em suas habilidades e conquistas em torneios. O sistema de pontuação Elo, desenvolvido por Arpad Elo, é a ferramenta mais utilizada para medir o nível de habilidade dos enxadristas, atribuindo a cada jogador uma pontuação que varia conforme seu desempenho em competições oficiais. Esta pontuação é dinâmica, aumentando ou diminuindo dependendo do resultado contra adversários com classificações superiores ou inferiores (EALES, 1993). A Federação Internacional de Xadrez (FIDE) utiliza esse sistema para determinar títulos honoríficos como Mestre Internacional (IM) e Grande Mestre (GM), com critérios específicos que incluem alcançar um determinado rating Elo e obter resultados consistentes em torneios reconhecidos (GLICKMAN, 1995). Além do Elo, a hierarquia é também influenciada pela performance em eventos de elite como o Campeonato Mundial, o Torneio de Candidatos e o Grand Chess Tour, onde apenas os melhores jogadores competem, elevando seu status

e prestígio dentro da comunidade enxadrística (KEENE, 2002). A estrutura hierárquica não apenas classifica os jogadores, mas também influencia convites para torneios e oportunidades de patrocínio, tornando-se um reflexo da evolução e competitividade no xadrez moderno.

A FIDE oferece outras classificações baseadas no sistema de rating Elo. Estas classificações servem para reconhecer diferentes níveis de habilidade no xadrez competitivo:

- Grande Mestre (GM): É o mais alto título concedido pela FIDE, exigindo um rating Elo de pelo menos 2500 e performances de alto nível em torneios internacionais.
- Mestre Internacional (IM): Um título abaixo de GM, geralmente exigindo um Elo de pelo menos 2400 e também performances consistentes em torneios.
- Mestre FIDE (FM): Um título intermediário com um rating mínimo de 2300, concedido a jogadores que demonstram bom desempenho em competições.
- Candidato a Mestre (CM): Para jogadores com rating de pelo menos 2200, representando um bom nível competitivo, mas abaixo dos padrões profissionais.
- Grande Mestre Feminino (WGM): Similar ao GM, mas adaptado para jogadoras do sexo feminino, exigindo um rating mínimo de 2300.
- Mestre Internacional Feminino (WIM): Equivalente ao IM, com um rating mínimo de 2200.
- Mestre FIDE Feminino (WFM) e Candidato a Mestre Feminino (WCM): Exigindo ratings de 2100 e 2000, respectivamente.

Além dos títulos oficiais, há classificações não oficiais que categorizam jogadores como:

- Iniciante: Elo abaixo de 1200, representando jogadores que estão aprendendo as regras e conceitos básicos.
- Intermediário: Elo entre 1200 e 1600, com compreensão básica de táticas e estratégias.
- Avançado: Elo entre 1600 e 2000, compreendendo bem teorias de abertura, meio-jogo e finais.

Estas classificações ajudam a medir o progresso de jogadores amadores e a determinar o nível competitivo. Os títulos, como GM e IM, são vitalícios, enquanto o rating Elo pode variar com o desempenho em torneios (ELO, 2008).

B.1.5 Jogadores de Relevânia Internacional

Aqui está uma lista com alguns dos jogadores mais relevantes da história do xadrez, incluindo tanto figuras históricas quanto jogadores contemporâneos. A lista inclui suas classificações e pontuações Elo.

B.1.5.1 Jogadores Históricos

1. **Garry Kasparov (GM)** Elo: 2851 Período ativo: 1980-2005 - Notável por ser um dos campeões mundiais mais dominantes da história do xadrez.
2. **Bobby Fischer (GM)** Elo: 2785 (pico em 1972) Período ativo: 1957-1975 - Famoso por vencer o Campeonato Mundial de 1972 contra Boris Spassky.
3. **Anatoly Karpov (GM)** Elo: 2725 (pico em 1994) Período ativo: 1970-2007 - Campeão mundial de 1975 a 1985, conhecido por seu estilo posicional.
4. **Mikhail Botvinnik (GM)** Elo: 2720 (pico em 1964) Período ativo: 1927-1970 - Considerado o pai da escola soviética de xadrez.

B.1.5.2 Jogadores Contemporâneos

1. **Magnus Carlsen (GM)** Elo: 2861 (pico em 2019) Período ativo: 2004-presente -Atual campeão mundial e amplamente considerado o melhor jogador da história.
2. **Hikaru Nakamura (GM)** Elo: 2800 (pico em 2021) Período ativo: 2003-presente - Reconhecido por suas habilidades em blitz e xadrez online.
3. **Fabiano Caruana (GM)** Elo: 2844 (pico em 2019) Período ativo: 2010-presente - Um dos principais concorrentes ao título mundial e conhecido por suas táticas agudas.
4. **Judit Polgar (GM)** Elo: 2735 (pico em 2005) Período ativo: 1988-2014 - A melhor jogadora de xadrez da história, superando muitos jogadores masculinos de elite.

Esses jogadores contribuíram e têm contribuído de maneira significativa para a história e evolução do xadrez, cada um à sua maneira (FIDE, 2024).

B.2 Xadrez na Sociedade

Ao permitir a interação entre pessoas de diferentes idades e origens, o xadrez também atua como uma ferramenta de socialização, ajudando a quebrar barreiras sociais e culturais. Em ambientes mais amplos, torneios e clubes de xadrez podem servir como espaços de respeito e competição saudável, contribuindo para a construção de um senso de comunidade e camaradagem entre os participantes.

B.2.1 Xadrez como Ferramenta Pedagógica

O xadrez tem se consolidado como uma ferramenta pedagógica valiosa nas escolas, oferecendo uma variedade de benefícios que vão além do simples entretenimento. Estudos mostram que ao introduzir o xadrez no ambiente escolar, os alunos desenvolvem habilidades cognitivas essenciais, como o raciocínio lógico, a resolução de problemas e a tomada de decisões (KAZEMI; YEKTAYAR; ABAD, 2012). Essas competências são fundamentais não apenas para o aprendizado do jogo, mas também para a formação de uma base sólida em disciplinas como matemática e ciências, onde a lógica e a estratégia são frequentemente aplicadas (SALA et al., 2017). O xadrez também estimula a paciência e a concentração, características vitais para o sucesso acadêmico (BURGOYNE et al., 2018).

Outro aspecto importante do xadrez como ferramenta pedagógica é a promoção de habilidades sociais e emocionais. O jogo ensina respeito e disciplina, uma vez que os alunos devem aprender a lidar com vitórias e derrotas de maneira saudável (ACIEGO; GARCÍA; BETANCORT, 2012). A prática do xadrez em grupo fomenta a interação entre os estudantes, promovendo um ambiente colaborativo e de respeito mútuo. Ao trabalhar em conjunto em competições ou em partidas rápidas, as crianças aprendem a se comunicar efetivamente e a desenvolver empatia, habilidades que serão essenciais em sua vida pessoal e profissional (SCHOLZ; ROCHA; BÖTTCHER, 2016). Dessa forma, o xadrez se revela não apenas um jogo estratégico, mas um poderoso aliado na formação integral dos estudantes.

B.2.2 Mulheres no Xadrez

A história das mulheres no xadrez remonta ao século XVI, quando a introdução da rainha no jogo tornou o xadrez mais rápido e competitivo, características então associadas à masculinidade. No século XIX, o surgimento de clubes de xadrez e publicações femininas, como o ABC of Chess de H.I. Cooke, trouxe as mulheres para o cenário enxadrístico. Jogadoras como Ellen Gilbert e Edith Winter-Wood destacaram-se em partidas por correspondência e na composição de problemas. A britânica Vera Menchik se tornou a primeira campeã mundial de xadrez feminino em 1927 e, ao derrotar vários mestres masculinos, quebrou barreiras e colocou o nome das mulheres no xadrez competitivo(SOLTIS, 2024).

O incentivo governamental, especialmente na União Soviética, ajudou a consolidar o xadrez feminino nas décadas seguintes, com campeonatos nacionais massivos que geraram talentos como Ludmilla Rudenko e Nona Gaprindashvili, a primeira mulher a receber o título de Grande Mestre Internacional. A ascensão de Judit Polgár nas décadas de 1980 e 1990, juntamente com suas irmãs Susan e Zsófia, revolucionou o esporte, desafiando a noção de que o xadrez competitivo é um domínio exclusivamente masculino. Polgar alcançou o status de Grande Mestre aos 15 anos, derrotando alguns dos maiores jogadores da história, como Garry Kasparov e Anatoly Karpov. Sua ascensão ao topo do ranking mundial de xadrez masculino demonstrou que as mulheres podem competir em igualdade de condições com os homens. Além de Polgar, várias outras jogadoras contemporâneas, como Hou Yifan, também têm se destacado e contribuído para a expansão do xadrez feminino globalmente (SOLTIS, 2024).

Figura B.1: Judit Polgar

Fonte: (FRIEDEL, 2024)

Na contemporaneidade, jogadoras como Hou Yifan e Koneru Humpy continuam a elevar o padrão, enquanto iniciativas como a FIDE WOM e a Fundação Susan Polgar

promovem maior inclusão e oportunidades. Embora desafios como a sub-representação e preconceitos de gênero ainda existam, o futuro das mulheres no xadrez parece promissor, com programas de base e maior visibilidade, inspirando uma nova geração de jogadoras (SCHAEFER, 2024).

B.2.3 Xadrez na Interação e Inclusão Social

O xadrez, além de ser um esporte intelectual, desempenha um papel crucial na interação e inclusão social, especialmente em comunidades marginalizadas. Sua natureza acessível e as poucas barreiras para a entrada o tornam uma ferramenta poderosa para unir indivíduos de diferentes origens e níveis socioeconômicos. O xadrez pode ser ensinado em ambientes diversos, como escolas, centros comunitários e prisões, promovendo habilidades cognitivas e de resolução de problemas, ao mesmo tempo que ajuda a construir redes sociais e a reduzir o isolamento social (BURGOYNE et al., 2018).

Em particular, programas de xadrez têm sido usados como ferramentas de intervenção social, especialmente com jovens em risco e populações vulneráveis. Estudos indicam que a prática do xadrez pode melhorar a atenção, o planejamento e o controle emocional, fatores que influenciam o comportamento e as tomadas de decisão. Esses benefícios são especialmente úteis para jovens em ambientes educacionais desafiadores ou com pouca supervisão familiar, promovendo comportamentos mais cooperativos e redução de conflitos (SALA et al., 2017). Além disso, iniciativas de xadrez em prisões demonstraram uma diminuição na reincidência de crimes, oferecendo aos presos uma atividade que promove disciplina e pensamento crítico (Schmidt, 2015).

O impacto social do xadrez também se estende à inclusão de pessoas com deficiências. O xadrez adaptado tem possibilitado que pessoas com deficiências visuais, físicas e cognitivas participem e se envolvam com o jogo, promovendo uma cultura de inclusão e superação. Estudos mostram que essa prática não só melhora as capacidades cognitivas desses indivíduos, mas também aumenta a autoestima e a percepção de inclusão na sociedade (BURGOYNE et al., 2018). Assim, o xadrez continua a desempenhar um papel vital na construção de comunidades mais inclusivas e resilientes.

APÊNDICE C — COMPLEMENTAÇÃO DA INFRAESTRUTURA DE UM AGENTE

C.1 Estágios da Geração de Movimentos

O método de geração de movimentos percorre diversos estágios, cada um dos quais foca em um tipo de movimento específico, desde capturas favoráveis a movimentos de evasão. Esse pipeline permite uma seleção eficiente dos melhores movimentos durante a pesquisa. Os estágios principais incluem:

- MAIN-TT, EVASION-TT, QSEARCH-TT e PROBCUT-TT: Movimentos relacionados à tabela de transposição (TT), que armazena resultados de posições já avaliadas anteriormente, evitando recomputações desnecessárias.
- CAPTURE-INIT, PROBCUT-INIT, QCAPTURE-INIT: Estágios iniciais de captura, onde são gerados os movimentos de captura, que são de interesse por potencialmente resultar em uma vantagem material imediata.
- GOOD-CAPTURE e BAD-CAPTURE: Movimentos de captura são divididos em bons e ruins, sendo os primeiros aqueles que capturam peças mais valiosas ou com risco reduzido de perder material em trocas subsequentes.
- QUIET-INIT e QUIET: Movimentos não capturadores são gerados nesses estágios e, posteriormente, ordenados com base em heurísticas como o histórico de bons movimentos em posições anteriores.
- EVASION-INIT e EVASION: Movimentos evasivos são gerados quando o rei está sob ataque, sendo priorizados os que removem a ameaça direta de xeque.
- PROBCUT, QCAPTURE, QCHECK-INIT e QCHECK: Relacionados à busca de profundidade limitada (quiescence search), onde apenas capturas e cheques são explorados, visando estabilizar posições voláteis.

C.2 Outras operações de Bitboard

C.2.0.1 Operações Morfológicas

As operações morfológicas são técnicas matemáticas aplicadas a conjuntos binários, frequentemente utilizadas no processamento de imagens, mas também adaptáveis

a estruturas como bitboards em xadrez. Essas operações são baseadas em um conceito fundamental de dilatação e erosão. A dilatação expande as áreas marcadas por bits "ativos"(bits com valor 1), o que pode ser entendido como a "expansão"do controle de uma peça no tabuleiro. Em um bitboard de xadrez, por exemplo, a dilatação pode simular a expansão do raio de alcance de uma peça, como uma torre ou um bispo, onde, a cada aplicação, o movimento potencial da peça pode ser estendido para cobrir mais casas. Já a erosão, ao contrário, reduz as áreas ativas, ou seja, limita os bits ativos no bitboard. No contexto de bitboards, isso pode ser útil para identificar casas que ainda estão sob controle após remover posições de fronteira. A erosão pode ser aplicada para diminuir o raio de ação de uma peça ou refinar o controle do tabuleiro em situações específicas.

C.2.0.2 Operação de Rotação

A operação de rotação é utilizada principalmente em agentes de xadrez para acelerar o cálculo de movimentos de peças como as torres e os bispos, cujas ações se propagam em linhas e diagonais. A rotação de bitboards envolve a "rotação"dos bits para a esquerda ou para a direita ao longo de um eixo específico (horizontal, vertical, ou diagonal) e pode ser usada para transformar o tabuleiro de modo a facilitar cálculos e comparações. A operação de rotação de bits em um bitboard consiste em deslocar todos os bits em uma direção específica e "reaparecer"do outro lado do bitboard (wrap around), como se os bits fossem organizados circularmente. No contexto de um tabuleiro de xadrez, isso pode ser visualizado como um deslocamento das casas do tabuleiro de modo que os bits que saem pela borda "retornam"do lado oposto. Embora não seja uma operação intrínseca a todos os processadores, é possível emular a rotação com combinações de deslocamentos e operações lógicas como OR e AND.

A rotação de bitboards é particularmente útil para calcular movimentos de peças que se deslocam em múltiplas direções. Por exemplo, ao calcular ataques de bispos, é necessário avaliar diagonais em direções diferentes, e a rotação de bitboards permite transformar o tabuleiro de modo que todas as diagonais ou antidiagonais sejam tratadas como colunas ou linhas. Isso facilita o uso de operações de deslocamento simples para calcular os movimentos.

Outro exemplo é no cálculo dos ataques de cavalos, onde a rotação pode ser usada para facilitar o mapeamento dos movimentos possíveis de um cavalo em múltiplas direções. Em vez de avaliar todas as direções separadamente, um bitboard rotacionado permite aplicar a mesma lógica de deslocamento em diferentes orientações.

C.3 Função Zobrist Hashing

A função Zobrist é uma técnica de hashing amplamente utilizada em agentes de xadrez e outros jogos de tabuleiro para representar o estado de um tabuleiro de forma compacta e eficiente. Ela é essencial em técnicas como a Tabela de Transposição, que armazena resultados de posições já calculadas, evitando a necessidade de recalcular jogadas já analisadas. Esse método de hashing é popular porque gera uma representação única e rápida de cada posição possível no tabuleiro de xadrez.

C.3.1 Como a função Zobrist funciona?

A técnica de hashing Zobrist funciona atribuindo a cada possível elemento em um tabuleiro (uma peça em uma determinada casa) um valor aleatório de bits. Esses valores são então combinados usando a operação XOR (ou exclusivo). O estado de um tabuleiro de xadrez pode ser representado pelo resultado dessa série de operações XOR, que leva em conta todas as peças e suas posições no tabuleiro. A grande vantagem do uso da operação XOR é que ela permite calcular de forma incremental o novo valor de hash à medida que o estado do tabuleiro muda.

Por exemplo, se uma peça for movida de uma casa para outra, o valor de hash pode ser atualizado de maneira muito eficiente sem precisar recalcular o hash inteiro. Basta aplicar o XOR para "remover"o valor associado à peça na casa antiga e "adicionar"o valor da peça na casa nova.

C.3.2 Cálculo do Hash Zobrist

O cálculo do hash Zobrist envolve três passos principais:

1. **Inicialização:** Para cada combinação possível de peça e posição no tabuleiro, um número aleatório de 64 bits é gerado. Como existem 6 tipos de peças (rei, dama, torre, bispo, cavalo e peão), 2 cores (brancas e pretas) e 64 casas no tabuleiro, você precisa de $6 \times 2 \times 64 = 768$ valores aleatórios para cobrir todas as possibilidades. Além disso, valores aleatórios adicionais são gerados para representar o roque, a regra do en passant e a vez do jogador (brancas ou pretas).
2. **Cálculo Inicial do Hash:** O valor de hash do tabuleiro é calculado realizando a

operação XOR entre os números aleatórios associados a cada peça e sua posição no tabuleiro. Isso gera um número único que representa o estado atual do jogo.

3. **Atualização do Hash:** À medida que as peças se movem no tabuleiro, o valor de hash pode ser atualizado de maneira incremental. Ao mover uma peça, realiza-se o XOR com o valor correspondente à peça na posição antiga para "remover"esse valor, e realiza-se o XOR com o valor correspondente à peça na nova posição para "adicionar"o novo valor.

C.3.3 Exemplo de Cálculo

Imagine que no tabuleiro de xadrez há um peão branco na posição e2. O valor de hash inicial é o resultado do XOR de todos os valores aleatórios de peças nas suas respectivas casas. Agora, se esse peão for movido de e2 para e4, para atualizar o hash, você faria o seguinte:

- XOR com o valor aleatório correspondente ao peão branco na posição e2 (removendo-o do hash).
- XOR com o valor aleatório correspondente ao peão branco na posição e4 (adicionando-o ao hash).

Dessa forma, a função Zobrist permite que as atualizações no estado do tabuleiro sejam extremamente rápidas e eficientes, o que é crucial para agentes de xadrez que precisam avaliar milhões de posições rapidamente.

C.3.4 Vantagens do Zobrist Hashing

Eficiência: A atualização incremental do hash é extremamente rápida, já que basta realizar operações XOR para adicionar ou remover peças. Compactação: O hash Zobrist condensa a complexidade de um tabuleiro de xadrez em um número de 64 bits. Minimização de Colisões: Como os números aleatórios gerados são distribuídos uniformemente, as chances de duas posições diferentes resultarem no mesmo hash são mínimas.

C.4 Fases da Pesquisa de Quiescência

Analisando o código da função Quiescência em agentes modernos como o Stockfish, é possível subdividir o método em 5 fases bem definidas.

1. **Verificação Inicial**: A pesquisa de quiescência geralmente começa quando a pesquisa principal atinge uma profundidade zero. O agente avalia a posição atual e determina se a avaliação está acima do valor *beta* (corte beta) ou abaixo de *alpha* (corte alpha), baseados nos limites da poda alfa-beta.
2. **Avaliação Estática Inicial** (Stand Pat): A primeira coisa que o agente faz na pesquisa de quiescência é realizar uma avaliação estática da posição. Este valor, chamado de "*stand pat*", é uma tentativa de estimar a avaliação da posição atual sem realizar qualquer movimento. Se esse valor já for suficiente para cortar a pesquisa (isto é, se for maior ou igual a beta), o agente retorna esse valor imediatamente. Isso é conhecido como o *corte beta*.
3. **Exploração de Capturas e Cheques**: Se o valor de "*stand pat*"não for suficiente para cortar a pesquisa, o agente gera todos os movimentos táticos relevantes, como capturas, promoções e cheques. Cada um desses movimentos é executado e a posição resultante é avaliada recursivamente com uma nova chamada à pesquisa de quiescência.
4. **Futuro de Capturas Não Favoráveis (Futility Pruning)**: Algumas técnicas de poda são aplicadas durante a pesquisa de quiescência, como o "futility pruning". Se o agente identificar que uma captura tem pouco ou nenhum potencial de melhorar a avaliação, pode ignorá-la sem gastar mais tempo avaliando a posição resultante. Um exemplo clássico é quando a captura de uma peça de valor menor por uma de valor maior não gera um ganho substancial de material.
5. **Retorno da Melhor Avaliação**: Após a exploração de todas as jogadas de quiescência possíveis, o agente retorna o melhor valor encontrado, que é a avaliação mais precisa da posição em questão.

C.5 Exemplo de Funcionamento do Algoritmo Minimax com Poda Alfa-Beta

O algoritmo Minimax com poda Alfa-Beta é uma técnica fundamental em jogos de dois jogadores, utilizada para determinar a melhor jogada possível. Apresentamos, a

seguir um detalhamento passo a passo de seu funcionamento.

C.5.1 Conceitos Fundamentais

Revisando alguns conceitos-chave:

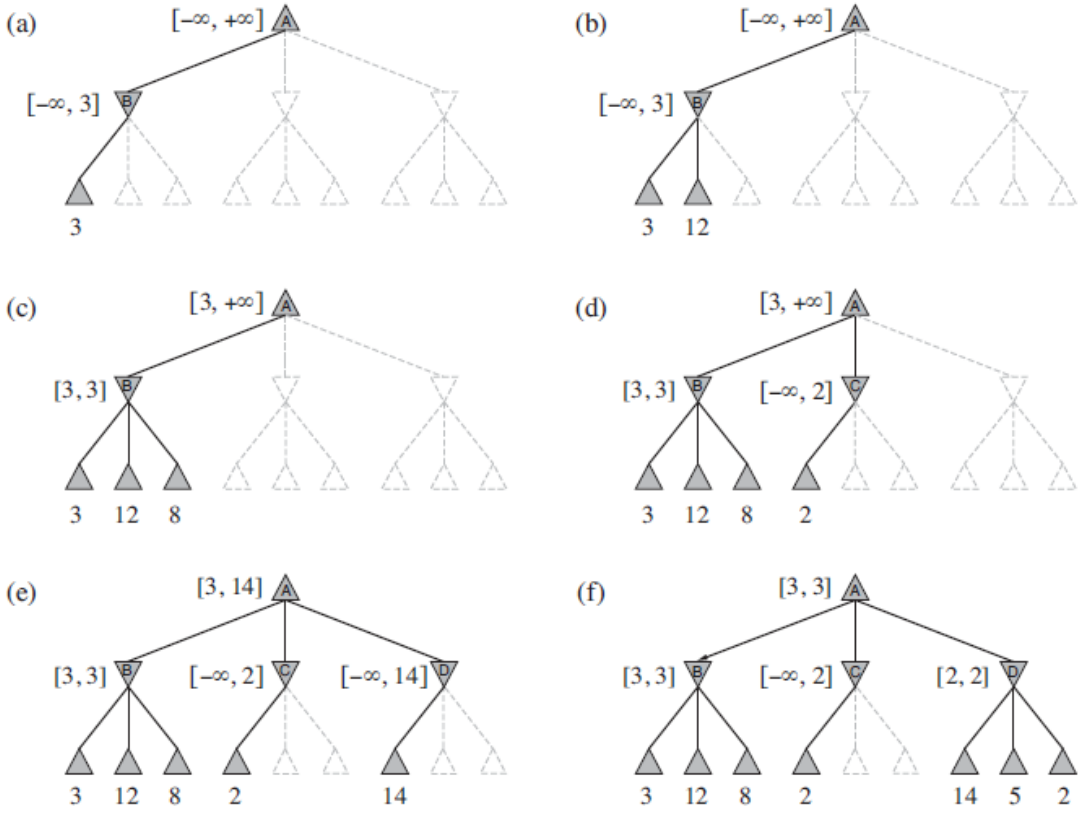
- **Nó:** Representa um estado possível do jogo.
- **MAX:** Nó onde o jogador que maximiza o valor realiza uma jogada.
- **MIN:** Nó onde o jogador que minimiza o valor realiza uma jogada.
- **Alfa:** Melhor valor encontrado até o momento para o jogador MAX.
- **Beta:** Melhor valor encontrado até o momento para o jogador MIN.
- **Poda:** Eliminação de ramos da árvore de busca que não influenciam a decisão final.

C.5.2 Funcionamento do Algoritmo

A figura C.2 apresenta um exemplo de funcionamento do algoritmo minimax com poda alfa-beta

1. **Inicialização:**
    - **Alfa:** Inicializado com o menor valor possível (negativo infinito).
    - **Beta:** Inicializado com o maior valor possível (positivo infinito).
2. **Busca em Profundidade:**
    - A partir do nó raiz, o algoritmo realiza uma busca em profundidade na árvore de jogo.
    - Ao chegar em uma folha (estado final do jogo), um valor é atribuído a ela, geralmente representando a utilidade daquele estado para o jogador MAX.
3. **Propagação de Valores:**
    - Os valores das folhas são propagados para cima na árvore.
    - **Nós MAX:** O valor de um nó MAX é o máximo dos valores de seus filhos.
    - **Nós MIN:** O valor de um nó MIN é o mínimo dos valores de seus filhos.
4. **Atualização de Alfa e Beta:**
    - **Alfa:** Ao visitar um nó MAX, o valor de alfa é atualizado para o máximo entre

Figura C.1: Exemplo Minimax



Exemplo de algoritmo Minimax com poda alfa-beta (RUSSELL; NORVIG, 2016)

o valor atual de alfa e o valor do nó MAX.

- **Beta:** Ao visitar um nó MIN, o valor de beta é atualizado para o mínimo entre o valor atual de beta e o valor do nó MIN.

5. **Poda:**

- **Poda Alfa:** Se o valor de um nó MIN for menor ou igual a alfa, todos os seus descendentes podem ser podados, pois eles não podem levar a um valor maior para o jogador MAX.
- **Poda Beta:** Se o valor de um nó MAX for maior ou igual a beta, todos os seus descendentes podem ser podados, pois eles não podem levar a um valor menor para o jogador MIN.

Figura C.2: Algoritmo Minimax

```
function ALPHA-BETA-SEARCH(state) returns an action
  v ← MAX-VALUE(state, −∞, +∞)
  return the action in ACTIONS(state) with value v

function MAX-VALUE(state, α, β) returns a utility value
  if TERMINAL-TEST(state) then return UTILITY(state)
  v ← −∞
  for each a in ACTIONS(state) do
    v ← MAX(v, MIN-VALUE(RESULT(s,a), α, β))
    if v ≥ β then return v
    α ← MAX(α, v)
  return v

function MIN-VALUE(state, α, β) returns a utility value
  if TERMINAL-TEST(state) then return UTILITY(state)
  v ← +∞
  for each a in ACTIONS(state) do
    v ← MIN(v, MAX-VALUE(RESULT(s,a) , α, β))
    if v ≤ α then return v
    β ← MIN(β, v)
  return v
```

Algoritmo Minimax com poda alfa-beta (RUSSELL; NORVIG, 2016)

C.6 Redes Neurais e Algoritmos Baseados em Aprendizado de Máquina

Redes neurais são estruturas computacionais inspiradas na neurociência, com o objetivo de simular o comportamento de neurônios biológicos, mas em um ambiente digital. Cada rede neural é composta de "neurônios" artificiais interconectados em camadas que trabalham em conjunto para processar informações, identificar padrões e gerar respostas a partir de dados complexos. Em particular, redes neurais são amplamente utilizadas em problemas onde a complexidade dos dados exige uma abordagem de modelagem não linear, como reconhecimento de imagem, processamento de linguagem natural e tomada de decisão em jogos.

O funcionamento de uma rede neural ocorre através de várias camadas: *camada de entrada* (input layer), *camadas ocultas* (hidden layers) e *camada de saída* (output layer). A camada de entrada recebe os dados brutos e os passa para as camadas ocultas, onde ocorre a transformação não linear, e, finalmente, a camada de saída gera o resultado processado, que pode ser uma classificação, uma previsão ou qualquer outra resposta programada.

Com o avanço das redes neurais profundas, surgiram novas abordagens para o

desenvolvimento de agentes de xadrez. Diferente do algoritmo minimax clássico, que avalia as posições de jogo com funções heurísticas, as redes neurais permitem uma avaliação mais refinada ao aprender padrões complexos e abstratos de posições e movimentos, aumentando a precisão e o desempenho dos agentes de xadrez.

C.6.1 Processo de Aprendizado em Redes Neurais

O processo de aprendizado em redes neurais ocorre principalmente de forma supervisionada, onde a rede aprende a partir de um conjunto de dados rotulado. Esse aprendizado é facilitado através de um processo de treinamento que utiliza o algoritmo de retropropagação (*backpropagation*) e otimizações por meio de gradiente descendente, ajustando os pesos e biases em cada nó da rede para minimizar a função de perda.

Durante o treinamento, o ajuste dos pesos ocorre iterativamente, onde cada ajuste busca reduzir o erro entre a saída esperada e a saída real da rede. Ao final do treinamento, a rede neural deve ser capaz de generalizar o aprendizado para novos dados, o que permite sua aplicação em problemas diversos, como classificação de imagens, predição de séries temporais, entre outros.

As primeiras aplicações de redes neurais no xadrez começaram de forma limitada devido à capacidade computacional restrita. No entanto, com a chegada de técnicas como redes neurais convolucionais (CNNs) e redes de memória de curto e longo prazo (LSTM), foi possível criar modelos que capturam padrões de tabuleiro e avaliam as posições em profundidade. Um dos marcos desse avanço foi a criação da AlphaZero, um motor desenvolvido pela DeepMind, que utiliza redes neurais profundas em conjunto com um algoritmo de busca de Monte Carlo.

C.6.2 Arquiteturas de Redes Neurais

Existem várias arquiteturas de redes neurais, cada uma projetada para lidar com tipos específicos de dados e desafios. As mais comuns incluem:

- **Redes Neurais Convolucionais (CNNs):** Ideais para processamento de imagens e reconhecimento de padrões visuais, as CNNs utilizam filtros convolucionais que extraem características locais, como bordas e texturas, nas imagens.
- **Redes Neurais Recorrentes (RNNs):** Utilizadas principalmente para dados se-

quenciais, como processamento de linguagem natural e previsão de séries temporais. As RNNs mantêm uma "memória" dos estados anteriores através de laços de retroalimentação, permitindo que a rede capture dependências temporais.

- **Redes Adversárias Generativas (GANs):** Consistem em duas redes que competem entre si: uma rede geradora, que cria dados falsos, e uma rede discriminadora, que tenta distinguir entre dados reais e falsos. Essa competição gera modelos que podem produzir dados sintéticos de alta qualidade, sendo amplamente utilizadas para criação de imagens e vídeos.
- **Rede Neural Eficientemente Atualizável (NNUE):** A NNUE usa uma rede densa (feedforward) em vez de convolucional, que se adapta bem ao cálculo de avaliações de posição em tempo real. Essa rede processa informações sobre as peças e a posição no tabuleiro de forma eficiente, calculando uma avaliação numérica que orienta o algoritmo de busca. É a arquitetura utilizada pelo Motor Stockfish.

Cada uma dessas arquiteturas é projetada para resolver problemas específicos, sendo selecionada com base nas necessidades do sistema em questão. A combinação de diferentes arquiteturas também é comum, visando maximizar a precisão e a eficácia dos modelos.

As primeiras aplicações de redes neurais no xadrez começaram de forma limitada devido à capacidade computacional restrita. No entanto, com a chegada de técnicas como redes neurais convolucionais (CNNs) e redes de memória de curto e longo prazo (LSTM), foi possível criar modelos que capturam padrões de tabuleiro e avaliam as posições em profundidade. Um dos marcos desse avanço foi a criação da AlphaZero, uma agente desenvolvido pela DeepMind, que utiliza redes neurais profundas em conjunto com um algoritmo de busca de Monte Carlo.

Em agentes modernos de xadrez, a arquitetura de rede neural mais amplamente adotada é a Rede Neural Convolucional (CNN), combinada com a Pesquisa de Monte Carlo (MCTS) para avaliar posições e selecionar movimentos. Exemplos notáveis incluem AlphaZero e Leela Chess Zero, que utilizam CNNs profundas com camadas convolucionais ajustadas especificamente para aprender padrões espaciais nas posições do tabuleiro.

C.6.3 Estrutura e Componentes das CNNs em Agentes de Xadrez

As CNNs são usadas devido à sua capacidade de capturar a estrutura espacial do tabuleiro de xadrez, interpretando a posição das peças e as relações entre elas de forma hierárquica. Cada camada convolucional aprende um conjunto específico de características, desde posições simples até padrões estratégicos complexos.

Já na AlphaZero, a rede é construída com várias *camadas residuais*, inspiradas pela arquitetura ResNet. Essas camadas ajudam a rede a lidar com redes muito profundas, permitindo a reutilização de informações das camadas anteriores e acelerando o aprendizado.

Os agentes de xadrez também uilizam as estruturas *Política de Rede* e *Polítia de Valor* onde a CNN é dividida em duas saídas: uma rede de política que sugere os movimentos mais promissores e uma rede de valor que estima a probabilidade de vitória para a posição atual. A rede de valor substitui a função de avaliação heurística usada em agentes clássicos, enquanto a rede de política orienta a pesquisa Monte Carlo em direção aos movimentos mais favoráveis

C.6.4 Algoritmos de Aprendizado de Máquina

O aprendizado de máquina (Machine Learning) é uma área da inteligência artificial que envolve a criação de algoritmos que podem aprender e fazer previsões com base em dados. Os algoritmos de aprendizado de máquina podem ser classificados em três categorias principais:

1. **Aprendizado Supervisionado:** Algoritmos supervisionados são treinados com um conjunto de dados rotulado, onde a saída correta é conhecida. Modelos supervisionados incluem regressão linear, árvores de decisão, máquinas de vetores de suporte (SVM) e redes neurais. Esses modelos são amplamente utilizados em classificação e predição.
2. **Aprendizado Não Supervisionado:** Algoritmos não supervisionados são aplicados a dados não rotulados, explorando o conjunto de dados para encontrar padrões e estruturas subjacentes. Algoritmos populares incluem clustering (*k-means*) e redução de dimensionalidade (*PCA*). Eles são úteis para análise de dados exploratória, onde as relações entre os dados não são conhecidas previamente.

3. **Aprendizado por Reforço:** Algoritmos de aprendizado por reforço operam com um agente que interage com o ambiente, recebendo recompensas ou penalidades baseadas nas ações executadas. Através da maximização das recompensas, o agente aprende uma política ótima de ações. Esse tipo de aprendizado é comumente aplicado em robótica e jogos complexos.

C.6.5 Integração de Redes Neurais com Aprendizado por Reforço

Recentemente, a integração de redes neurais com aprendizado por reforço, conhecida como aprendizado profundo por reforço (*Deep Reinforcement Learning*), tem permitido avanços significativos em tarefas complexas como jogos e controle de robôs. Um exemplo notável é o *Deep Q-Network* (DQN), que utiliza uma rede neural profunda para aproximar a função Q, que determina a ação ótima para cada estado em ambientes de tomada de decisão sequencial.

No xadrez e em outros jogos, redes neurais com aprendizado por reforço têm mostrado grande eficácia. A rede neural treina continuamente enquanto joga partidas simuladas, melhorando progressivamente sua habilidade de prever jogadas e desenvolver estratégias complexas. Esse é o princípio utilizado pelo AlphaZero, um dos modelos mais avançados, que combina o aprendizado de redes neurais com MCTS para maximizar o desempenho em jogos.

C.6.6 Arquitetura NNUE do Stockfish

As últimas versões do Stockfish, a partir da versão 12, combinam a tradicional árvore de busca Alfa-Beta com uma Rede Neural Leve NNUE. Essa rede foi projetada para complementar o mecanismo de busca minimax do Stockfish, focando na avaliação de posições. A abordagem NNUE é diferente das redes convolucionais (CNNs) usadas em moteres como AlphaZero e Leela Chess Zero, pois foi desenvolvida para ser altamente eficiente em CPUs, aproveitando características que tornam o treinamento e a inferência rápidos e precisos.

Ao contrário das CNNs, a rede NNUE é otimizada para rodar em CPUs, aproveitando operações rápidas de SIMD (Single Instruction, Multiple Data), permitindo que o Stockfish funcione com alta performance em CPUs comuns, sem a necessidade de GPUs.

A rede NNUE gera avaliações que são integradas ao tradicional algoritmo de busca Alfa-Beta do Stockfish. Com isso, o agente mantém a profundidade e o controle do algoritmo clássico, ao mesmo tempo que ganha uma avaliação mais precisa das posições através da rede neural.

O termo "Efficiently Updatable"de NNUE significa que a rede neural do Stockfish pode ser atualizada incrementalmente em função de mudanças na posição do tabuleiro, o que reduz a necessidade de recalcular completamente as avaliações para cada novo movimento. Isso torna o agente ainda mais eficiente e rápido.

C.6.7 Aplicações Futuras e Potenciais de Redes Neurais

Com o avanço das técnicas de aprendizado de máquina e o surgimento de arquiteturas de redes neurais mais eficientes, espera-se que as aplicações continuem a se expandir. Em jogos, redes neurais combinadas com aprendizado por reforço abrem possibilidades para desenvolvimentos mais complexos e estratégias de jogo quase perfeitas. No campo da saúde, redes neurais auxiliam na detecção precoce de doenças a partir de imagens médicas. E na área de processamento de linguagem, possibilitam a geração automática de texto e compreensão de linguagem natural com níveis de precisão sem precedentes.

C.6.8 Limitações

Apesar de seus muitos benefícios, redes neurais e algoritmos de aprendizado de máquina enfrentam diversos desafios:

- **Superfitting e Generalização:** Um dos principais problemas no treinamento de redes neurais é o *overfitting*, onde o modelo aprende detalhes específicos do conjunto de treinamento e falha em generalizar para novos dados. Técnicas como *dropout* e regularização ajudam a mitigar esse problema, mas o ajuste dos parâmetros de um modelo ainda é um desafio central.
- **Transparência e Explicabilidade:** Redes neurais são frequentemente consideradas “caixas pretas” devido à dificuldade de interpretar seus processos internos. Esse problema é significativo em áreas onde a interpretabilidade é crucial, como na saúde e na justiça, onde decisões precisam ser justificadas de forma clara. Pouco agentes

de xadrêz são capazes de explicar jogadas ou estratégias.

- **Requisitos Computacionais:** O treinamento de redes neurais profundas pode ser intensivo em termos de tempo e recursos computacionais, especialmente em grandes conjuntos de dados. Hardware especializado, como GPUs e TPUs, é frequentemente necessário para suportar o processamento.

C.7 Detalhamento da Função de Avaliação de Agentes Modernos

A avaliação de tabuleiros é um dos aspectos fundamentais para o funcionamento de um agente de xadrez. Por meio dela, o agente calcula o valor numérico de uma posição para decidir quais movimentos executar. Este capítulo aborda detalhadamente os fatores considerados por agentes como o Stockfish, categorizados em aspectos estruturais, posicionais, dinâmicos e específicos de finais.

A combinação de todos esses fatores gera uma avaliação precisa e numérica para cada posição, permitindo que motores como Stockfish escolham os melhores movimentos. A profundidade da análise e o ajuste fino desses parâmetros são o que diferenciam agentes de alto nível, permitindo-lhes competir em níveis super-humanos.

C.7.1 Estrutura de Peões

Uma das métricas mais comuns para avaliar a estrutura de peões é a identificação de peões dobrados, isolados ou atrasados. Peões dobrados, que ocupam a mesma coluna, geralmente representam uma fraqueza porque não podem se proteger mutuamente e podem ser alvos fáceis para ataques. Peões isolados, que não têm peões vizinhos na mesma fileira para protegê-los, também são considerados vulneráveis. Já os peões atrasados, que não conseguem avançar de forma segura devido à falta de suporte dos peões adjacentes, podem limitar a mobilidade das próprias peças e expor a posição.

Outro aspecto importante na avaliação de estruturas de peões é a formação de cadeias de peões. Uma cadeia bem estruturada, onde cada peão é protegido por outro na diagonal, é altamente valorizada, pois oferece estabilidade e controla importantes casas no tabuleiro. A presença de peões passados, que não enfrentam peões adversários na mesma coluna ou em colunas adjacentes, é especialmente benéfica, já que representam uma ameaça constante de promoção.

Além disso, a ocupação e o controle do centro com peões são analisados cuidadosamente. Peões centrais fortes permitem um controle significativo sobre o tabuleiro, restringindo os movimentos das peças adversárias e facilitando o desenvolvimento de peças maiores. Por outro lado, peões mal posicionados podem obstruir o movimento das próprias peças e criar buracos na posição, que podem ser explorados pelo oponente.

C.7.2 Controle do Tabuleiro

O controle do centro é um dos aspectos fundamentais na avaliação de tabuleiros em xadrez, desempenhando um papel crucial na dominância posicional e na mobilidade das peças. Tradicionalmente, as casas centrais e4, d4, e5 e d5 são vistas como o coração do tabuleiro, pois permitem que as peças exerçam influência sobre uma ampla área, facilitando ataques e defesa. Peões e peças que ocupam ou controlam essas casas oferecem ao jogador uma vantagem estratégica significativa, restringindo o movimento do adversário e criando oportunidades de ataque.

Além do controle das casas centrais, a avaliação leva em consideração o domínio de fileiras e colunas abertas ou semiabertas. Torres e damas posicionadas em colunas abertas exercem pressão sobre as posições adversárias, tornando possível invadir o território inimigo ou atacar alvos estratégicos. O controle de fileiras, especialmente a sétima e oitava fileiras para as pretas, ou a segunda e primeira fileiras para as brancas, pode ser decisivo. Torres na sétima fileira, por exemplo, ameaçam peões não defendidos e restringem o rei adversário, conferindo uma vantagem poderosa ao controlador dessa área.

Casas críticas, que são aquelas cuja posse oferece benefícios substanciais, como suporte a peças ou avanços de peões, também desempenham um papel crucial na avaliação. Exemplos incluem casas avançadas protegidas por peões, conhecidas como "postos avançados", que servem como pontos de ancoragem ideais para cavalos ou outras peças menores. O domínio de casas críticas permite ao jogador criar planos de longo prazo e restringir a liberdade das peças inimigas. Um cavalo em um posto avançado pode ser especialmente devastador, controlando múltiplas casas enquanto permanece difícil de desalojar.

Outro aspecto importante do controle do centro e das casas críticas é a possibilidade de criar buracos na posição adversária. Buracos são casas que não podem ser facilmente defendidas por peões e, portanto, se tornam pontos fracos crônicos na estrutura do adversário. Explorar essas debilidades, colocando peças em posições fortes, pode

ser a chave para construir um ataque bem-sucedido.

Por fim, o controle dinâmico do centro e das casas críticas é frequentemente mais valioso do que o controle estático. Peças que ameaçam invadir o centro ou colunas abertas, mesmo sem ocupá-las imediatamente, obrigam o adversário a gastar recursos para defender essas áreas. Assim, a avaliação de controle do centro, fileiras, colunas e casas críticas não apenas considera a força atual de uma posição, mas também seu potencial de crescimento e a capacidade de restringir as opções do oponente.

C.7.3 Atividade e Mobilidade das Peças

A atividade e mobilidade das peças são fatores cruciais na avaliação de uma posição no xadrez, pois determinam a eficácia das peças em influenciar o jogo. A mobilidade refere-se ao número de casas para as quais uma peça pode se mover legalmente. Uma maior mobilidade geralmente indica que a peça possui mais opções estratégicas, seja para atacar, defender ou apoiar outras peças. A limitação da mobilidade, por outro lado, pode restringir a capacidade de resposta de um jogador e criar vulnerabilidades na posição.

A coordenação entre as peças é outro aspecto importante. Peças que trabalham em conjunto de forma eficiente podem criar ameaças mais poderosas, proteger pontos fracos e construir planos coesos. Por exemplo, a colaboração entre torres, especialmente quando dobradas em uma coluna aberta, ou entre um bispo e uma torre atacando pontos críticos, pode ser devastadora para a posição adversária. A falta de coordenação, por sua vez, pode levar a peças mal colocadas e planos ineficazes.

Torres em colunas abertas são altamente valorizadas na avaliação de uma posição. Uma torre posicionada em uma coluna sem peões exerce grande pressão sobre o adversário, controlando a linha vertical e ameaçando invadir o território inimigo. Quando duas torres estão conectadas em uma coluna aberta ou semiaberta, a pressão aumenta exponencialmente, tornando difícil para o adversário neutralizar a ameaça sem comprometer sua estrutura de peões ou outras peças.

Bispos em diagonais longas são outro elemento de grande relevância. Esses bispos têm o poder de controlar vastas áreas do tabuleiro, influenciando diretamente tanto o centro quanto o flanco. Eles podem ser particularmente perigosos em posições abertas, onde sua mobilidade não é restringida por peões. O domínio de diagonais críticas também cria ameaças de longo alcance e restringe o movimento das peças adversárias.

Cavalos posicionados em outposts – casas avançadas protegidas por peões ou que

o adversário não pode contestar com seus próprios peões – são uma força poderosa. Esses cavalos têm a capacidade de atacar múltiplas casas e muitas vezes se tornam peças centrais em um ataque. Eles podem ser difíceis de desalojar, obrigando o adversário a gastar recursos consideráveis para removê-los ou neutralizar sua influência.

No geral, a avaliação da atividade e mobilidade das peças leva em conta não apenas o potencial imediato das peças, mas também como elas interagem e se posicionam para o futuro. Peças ativas e bem coordenadas geralmente criam uma posição fluida e dinâmica, enquanto peças passivas e mal colocadas podem limitar severamente as opções de um jogador, eventualmente resultando em colapso posicional. Assim, otimizar a atividade e mobilidade é essencial para dominar o tabuleiro e buscar a vitória.

C.7.4 Segurança do Rei

A segurança do rei é um dos fatores mais críticos na avaliação de posições no xadrez, pois um rei vulnerável pode rapidamente levar a ameaças táticas, ataques decisivos e, em última instância, ao xeque-mate. A avaliação da segurança do rei geralmente começa com a análise da estrutura de peões ao seu redor. Peões que permanecem intactos e bem organizados oferecem uma barreira protetora contra ataques diretos, enquanto fraquezas como peões avançados ou ausentes podem abrir linhas de ataque para as peças adversárias.

A posição do rei após o roque também é um elemento essencial na avaliação. O roque é uma manobra estratégica que geralmente move o rei para um local mais seguro, longe do centro e das principais linhas de ataque. No entanto, a segurança do rei após o roque depende da integridade da estrutura de peões no flanco para onde ele se deslocou. Por exemplo, avançar peões do roque de forma imprudente pode enfraquecer o abrigo do rei, criando buracos que podem ser explorados por peças adversárias. Adicionalmente, se o adversário controlar colunas abertas próximas ao rei roqueado, a segurança do monarca pode estar seriamente comprometida.

Outro aspecto importante na avaliação é a exposição do rei adversário. Um rei exposto, seja por falta de cobertura de peões ou por se encontrar no centro do tabuleiro, torna-se um alvo prioritário. Isso permite ao jogador criar ameaças de xeque, forçar trocas vantajosas ou até iniciar combinações táticas que resultem em material ganho ou mate. Por exemplo, se o adversário adianta muitos peões em busca de ataque, isso pode deixar seu próprio rei vulnerável a contra-ataques rápidos.

Além disso, a segurança do rei é frequentemente influenciada pelo estágio da partida. No meio-jogo, a proteção do rei é crucial, pois a maioria das ameaças diretas ocorre nesse momento. No entanto, no final do jogo, o rei muitas vezes se torna uma peça ativa, e sua segurança é avaliada em relação à sua capacidade de contribuir para o jogo sem correr risco imediato.

C.7.5 Material

A avaliação material é a base fundamental da maioria das funções de avaliação no xadrez, pois fornece uma medida quantitativa direta da força relativa de cada lado com base nas peças presentes no tabuleiro. Cada peça é atribuída um valor numérico que reflete sua utilidade e impacto no jogo. Por exemplo, peões geralmente valem 1, cavalos e bispos valem 3, torres valem 5 e a dama vale 9. Esses valores servem como um ponto de partida para avaliar a posição, embora o contexto posicional frequentemente modifique sua relevância.

Um aspecto importante da avaliação material é o equilíbrio entre peças menores e peças maiores. Um jogador pode sacrificar material menor, como peões, em troca de um ataque forte ou controle posicional superior. Além disso, o par de bispos é frequentemente avaliado como uma vantagem devido à sua capacidade de controlar longas diagonais em posições abertas. Da mesma forma, a troca de uma torre por duas peças menores, como um cavalo e um bispo, pode ser vantajosa dependendo da dinâmica da posição.

A mobilidade das peças afeta diretamente a avaliação material. Por exemplo, uma torre em uma coluna aberta ou semiaberta tem mais valor prático do que uma torre bloqueada por peões. De forma semelhante, um bispo em uma longa diagonal pode ser mais influente do que um cavalo com pouca mobilidade. Esses fatores posicionais podem justificar ajustes na avaliação material para refletir melhor a realidade da posição.

No final do jogo, a importância relativa das peças pode mudar. Um rei ativo, que inicialmente tem apenas valor defensivo, torna-se uma peça poderosa na fase final, influenciando diretamente a avaliação. Da mesma forma, o valor dos peões aumenta significativamente à medida que se aproximam da promoção, muitas vezes se tornando a peça mais importante no tabuleiro.

C.7.6 Avaliação Tática

A avaliação tática em uma posição de xadrez é um elemento essencial que mede as possibilidades imediatas de ataque, defesa e combinações táticas disponíveis para cada lado. Ao contrário de aspectos estratégicos, que geralmente têm efeitos a longo prazo, a avaliação tática foca em oportunidades de curto prazo que podem resultar em ganho de material, melhora da posição ou execução de ameaças diretas, como xeque-mate.

Um dos principais componentes da avaliação tática é a identificação de ameaças. Ameaças podem variar desde capturas diretas de peças até planos mais elaborados, como cravar uma peça adversária, explorar um rei mal protegido ou atacar peças indefesas. A função de avaliação deve ser capaz de reconhecer essas oportunidades e atribuir um peso apropriado a elas. Por exemplo, uma ameaça iminente de capturar uma peça de alto valor deve ser valorizada mais do que uma ameaça menos significativa.

O potencial de ataque é outro aspecto crítico. Ele avalia a capacidade de um jogador de organizar um ataque eficaz contra o rei ou outras peças adversárias. Peças ativas e coordenadas, como torres em colunas abertas, bispos em diagonais longas e damas em posições avançadas, contribuem para um alto potencial de ataque. Além disso, fatores como a presença de peões avançados ou controle de casas críticas próximas ao rei adversário aumentam a capacidade de ataque.

Por outro lado, a avaliação tática também considera o potencial de defesa. Uma boa posição defensiva pode neutralizar ameaças táticas e criar contra-ataques perigosos. A habilidade de defender peças importantes, cobrir buracos na estrutura de peões ou reorganizar peças para proteger o rei são elementos essenciais na avaliação da defesa. Além disso, a habilidade de bloquear linhas de ataque ou trocar peças ativamente para aliviar a pressão adversária é um indicador de uma posição defensiva sólida.

Combinações táticas, como garfos, cravadas, raios X e descobertas, desempenham um papel central na avaliação tática. A função de avaliação deve ser capaz de identificar essas oportunidades e dar um valor apropriado a elas, pois podem alterar significativamente o equilíbrio de uma partida em poucas jogadas.

C.7.7 Avaliação Posicional

A avaliação posicional no xadrez considera elementos de longo prazo que influenciam o potencial de cada lado na partida, mesmo na ausência de táticas imediatas ou

vantagens materiais significativas. Esses fatores posicionais fornecem um alicerce para avaliar a qualidade estratégica da posição e a capacidade de um jogador em explorar ou consolidar suas vantagens.

Espaço é um dos aspectos mais importantes na avaliação posicional. Um jogador com mais espaço tem maior liberdade para manobrar suas peças, coordená-las de forma eficaz e limitar as opções do adversário. Espaço é frequentemente determinado pelo alcance dos peões, pois eles formam barreiras que definem territórios no tabuleiro. Por exemplo, peões avançados que controlam casas no campo adversário podem restringir a mobilidade das peças inimigas e abrir caminhos para ataques. No entanto, controlar espaço requer cautela, pois peões avançados demais podem se tornar vulneráveis a ataques.

Casas fracas e fortes são um elemento crítico na avaliação posicional. Uma casa fraca é uma casa que não pode ser defendida por peões, tornando-se um ponto vulnerável que o adversário pode explorar. Por outro lado, uma casa forte é uma casa estratégica, muitas vezes localizada no campo adversário, onde uma peça pode ser colocada de forma segura e eficaz, como um cavalo em um outpost. Identificar e utilizar casas fortes, enquanto se minimiza a criação de casas fracas, é um componente chave de uma boa estratégia posicional.

A distinção entre peças ativas e peças passivas também é central na avaliação posicional. Peças ativas são aquelas que exercem influência significativa sobre o tabuleiro, atacando casas importantes, restringindo a mobilidade do adversário ou controlando pontos críticos. Já peças passivas são peças limitadas em sua mobilidade e impacto estratégico, geralmente devido a mau posicionamento ou restrições impostas pela estrutura de peões ou peças adversárias. Melhorar a atividade de peças passivas é muitas vezes um objetivo estratégico em posições equilibradas.

Além disso, desenvolvimento harmonioso e coordenação entre peças são fatores que complementam a avaliação posicional. Um jogador com peças melhor posicionadas pode construir ataques coesos ou consolidar uma defesa eficaz, enquanto posições descoordenadas tendem a ser mais vulneráveis.

C.7.8 Avaliação Dinâmica

A avaliação dinâmica no xadrez foca nas mudanças rápidas de características e possibilidades da posição, refletindo a capacidade de um jogador de tomar a iniciativa, criar ameaças e explorar fraquezas no campo adversário. Ao contrário de uma avalia-

ção mais estática, que se baseia em uma análise das peças e da estrutura de peões em determinado momento, a avaliação dinâmica considera o potencial de ação no curto e médio prazo. Ela envolve a capacidade de um jogador de influenciar a partida de maneira decisiva por meio de movimento rápido e a criação de oportunidades de ataque ou defesa.

Iniciativa é um dos pilares da avaliação dinâmica. Ter a iniciativa significa ser capaz de ditar o ritmo da partida, criando ameaças que o adversário precisa responder, muitas vezes de forma defensiva. A iniciativa pode surgir de uma variedade de fontes, como ataques diretos, controle de linhas abertas ou criação de ameaças múltiplas. A posição de um jogador com iniciativa é frequentemente mais flexível, pois ele tem o controle das ameaças e pode decidir quando e onde atacar. Isso força o adversário a reagir, muitas vezes em desvantagem. O controle de colunas abertas ou semiabertas, especialmente com torres e damas, é uma maneira comum de manter a iniciativa, pois essas peças podem exercer pressão constante sobre o oponente.

O potencial de ataque é outra faceta importante da avaliação dinâmica. Um jogador com um forte potencial de ataque tem a capacidade de criar ameaças tangíveis contra o rei adversário ou outras peças valiosas. O ataque não se resume apenas a uma série de movimentos ofensivos, mas à habilidade de manter a pressão e explorar as fraquezas na posição do oponente. Peças bem coordenadas, como torres em colunas abertas, bispos em diagonais longas e cavalos em posições avançadas, contribuem para um alto potencial de ataque. Além disso, a mobilização rápida de peças menores, como cavalo ou bispo, para apoiar a ofensiva, também aumenta o poder de ataque, mantendo o oponente em alerta constante.

Contra-ataque, por outro lado, é a capacidade de reverter a pressão adversária e aproveitar as fraquezas na posição do oponente para lançar um ataque. Um jogador habilidoso em contra-ataque é capaz de esperar pacientemente por um erro ou excesso de confiança do adversário, transformando uma posição defensiva em uma ofensiva. O contra-ataque pode ser uma resposta a um ataque agressivo do oponente, onde as peças inimigas se expõem ao ataque ao avançar sem o devido apoio. Em tais situações, a capacidade de explorar as vulnerabilidades, como a falta de proteção do rei ou o afastamento das peças defensivas, pode resultar em uma mudança decisiva na partida.

A avaliação dinâmica é essencialmente interligada com a análise tática, pois um bom entendimento das ameaças e oportunidades de ataque permite ao jogador se aproveitar de combinações táticas no momento oportuno. A transição entre a criação de iniciativas e ataques para a defesa contra ataques adversários é uma habilidade vital para

jogadores que buscam controlar as partidas de forma eficaz.

C.7.9 Avaliação de Finais

A avaliação de finais é uma parte crucial da análise de uma partida de xadrez, pois envolve a transição para a fase em que as peças restantes no tabuleiro determinam a possibilidade de vitória ou empate. Durante os finais, o jogo se torna mais focado na precisão, no posicionamento estratégico das peças e na utilização eficiente do espaço, especialmente em relação aos reis, peões passados e à atividade das peças. A avaliação de finais é frequentemente distinta da avaliação das fases iniciais e intermediárias, pois a dinâmica das posições muda drasticamente, e a capacidade de planejar movimentos mais simples e diretos ganha importância.

A posição dos reis no final do jogo é fundamental, pois o rei se torna uma peça mais ativa e muitas vezes participa diretamente na luta pela vitória. Em finais, os reis são frequentemente mais expostos, já que muitas peças foram trocadas, e a proteção em torno deles é reduzida. A capacidade de mover o rei para uma posição ativa, onde ele possa apoiar a promoção de peões ou proteger peões passados, é crucial. Um rei centralizado é geralmente mais forte, pois pode ajudar a controlar várias áreas do tabuleiro. Além disso, o rei também pode ser usado para bloquear peões adversários ou se colocar em uma posição de defesa, se necessário. Em finais de peões, a mobilidade do rei é frequentemente mais importante do que em outras fases do jogo, e um rei "passivo", restrito a movimentos defensivos, pode ser uma desvantagem.

Os peões passados são um dos fatores mais decisivos nos finais, pois são a principal ameaça de promoção, que pode resultar na vitória. A presença de um peão passado, especialmente perto da linha de promoção, é uma vantagem significativa. A avaliação de um peão passado depende de sua posição no tabuleiro, do apoio que ele tem (se é protegido por outras peças ou peões) e da capacidade do adversário de impedir sua promoção. Um peão passado que não pode ser bloqueado ou atacado de forma eficaz pelo adversário geralmente se torna o centro das operações no final, com o objetivo de promovê-lo a dama. Em finais, o controle de casas chave ao redor de peões passados — como as casas que eles controlam ou as casas que o rei adversário deve ocupar para bloquear sua promoção — é essencial para avaliar a posição.

A atividade das peças também é crucial no final do jogo. Peças ativas são aquelas que estão bem posicionadas para apoiar ataques, controlar importantes casas, defender

peões passados ou impedir a promoção dos peões adversários. Torna-se muito mais importante, em finais, ter peças que podem movimentar-se livremente e apoiar a ação, ao invés de peças passivas que ficam limitadas em movimentos por obstáculos ou falta de espaço. Por exemplo, torres em colunas abertas ou semiabertas podem ser extremamente poderosas, pois têm a capacidade de pressionar peões e restringir a mobilidade do adversário. Cavalos e bispos podem ganhar uma importância renovada devido à sua capacidade de controlar áreas amplas do tabuleiro, com os cavalos muitas vezes sendo mais valiosos em finais, devido à sua habilidade de saltar sobre peças adversárias e alcançar posições críticas.

Em finais, as trocas de peças tendem a ser mais favoráveis quando há a possibilidade de simplificar a posição e transformar uma vantagem material em uma vitória. O conceito de troca de peças vantajosa se torna mais relevante, pois a habilidade de eliminar peças adversárias, enquanto mantém ou cria ameaças de promoção de peões, pode ser a chave para a vitória.

C.7.10 Outros Fatores

Desenvolvimento de peças é um fator fundamental para garantir que um jogador tenha flexibilidade e controle sobre a partida. As peças bem desenvolvidas, centralizadas e coordenadas têm muito mais potencial para realizar ataques e defender contra ameaças. O desenvolvimento de peças deve ser eficiente, pois uma peça mal posicionada pode perder sua função e reduzir as opções estratégicas.

O par de bispos, por sua vez, é uma vantagem estratégica, especialmente em finais. Controlando grandes áreas do tabuleiro e com a capacidade de se mover livremente, o par de bispos pode criar uma vantagem tática significativa, proporcionando uma cobertura mais ampla e melhor capacidade de ataque e defesa. Embora nem sempre seja decisivo, o par de bispos é um elemento que deve ser levado em consideração ao avaliar a posição e as possibilidades no jogo.

Ambos os fatores, desenvolvimento das peças e o par de bispos, adicionam uma camada importante à avaliação estratégica de uma posição, muitas vezes determinando a eficiência das jogadas e a adaptabilidade do jogador a diferentes cenários durante a partida.

APÊNDICE D — INTERFACE GRÁFICA PADRONIZADA PARA O AGENTE

As interfaces gráficas para jogos de xadrez desempenham um papel crucial na experiência do usuário, possibilitando que os jogadores interajam de maneira intuitiva e acessível com o agente de xadrez. Este capítulo explora os elementos essenciais de uma interface gráfica para jogos de xadrez, incluindo a visualização do tabuleiro, ferramentas de análise, controle do tempo e os métodos de personalização. Interfaces bem projetadas facilitam o engajamento do jogador, tornando o xadrez acessível tanto para amadores quanto para profissionais (Pietrow e Sadler, 2018).

D.1 Elementos Essenciais de uma Interface Gráfica de Xadrez

Interfaces gráficas (GUI) desempenham um papel essencial no contexto dos jogos de xadrez, pois proporcionam uma ponte entre a complexidade dos agentes de xadrez e o usuário final, seja ele um jogador ou um analista (LEVY; NEWBORN, 1991). O objetivo principal das interfaces de xadrez é permitir que o usuário visualize o tabuleiro, mova peças e analise o jogo de maneira interativa. Com o desenvolvimento de interfaces gráficas específicas para o xadrez, houve uma evolução no modo como o usuário interage com agentes de alto desempenho e algoritmos de busca.

As interfaces modernas de xadrez, como *Arena*, *Fritz* e *ChessBase*, suportam interações com múltiplos agentes e permitem configurações detalhadas que maximizam a eficiência da análise (CAMPBELL; HOANE; HSU, 2002; SADLER; REGAN, 2019). Elas viabilizam o acompanhamento de partidas, o treinamento, a análise de posições, além de oferecerem suporte para diferentes formatos de arquivos de xadrez, como PGN (Portable Game Notation) e FEN (Forsyth-Edwards Notation) (PIETROW; SADLER, 2018).

Outro ponto relevante é o suporte ao protocolo UCI (Universal Chess Interface), amplamente adotado entre agentes modernos. Esse protocolo possibilita a integração de agentes distintos com interfaces de xadrez, independentemente do desenvolvedor ou da arquitetura subjacente do agente, tornando o UCI um dos padrões mais estabelecidos e flexíveis para interação (RUSSELL; NORVIG, 2010).

- **Visualização do Tabuleiro:** Um dos aspectos fundamentais de uma GUI de xadrez é a representação gráfica do tabuleiro e das peças, o que permite que o usuário acompanhe a partida em andamento de forma clara e intuitiva.

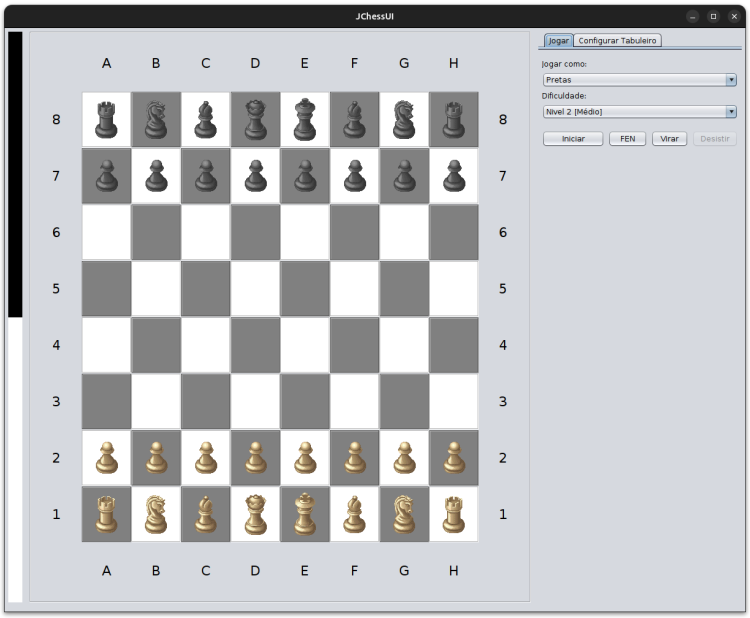
- **Movimentação das Peças:** O suporte a movimentação de peças por meio de cliques ou arrastamento é outra funcionalidade crucial, que oferece uma experiência mais interativa e próxima ao jogo de xadrez tradicional (LEVY; NEWBORN, 1991).
- **Análise e Configurações de Agente:** A maioria das GUIs permite o ajuste dos parâmetros do agente de xadrez, incluindo a profundidade de busca e o tempo máximo de análise, além de exibir uma avaliação em tempo real da posição no tabuleiro (SADLER; REGAN, 2019).

Esses recursos tornaram as GUIs uma ferramenta indispensável para jogadores profissionais e amadores, permitindo que motores avançados como *Stockfish* e *Leela Chess Zero* sejam operadas de forma amigável e acessível (CAMPBELL; HOANE; HSU, 2002).

D.2 Descrição da Interface do Agente de Xadrez

A interface do agente de xadrez foi projetada para ser intuitiva e funcional, proporcionando uma experiência de jogo agradável e eficiente. A seguir, descrevemos os principais componentes da interface.

Figura D.1: Interface JChess posição inicial



Tabuleiro Inicial

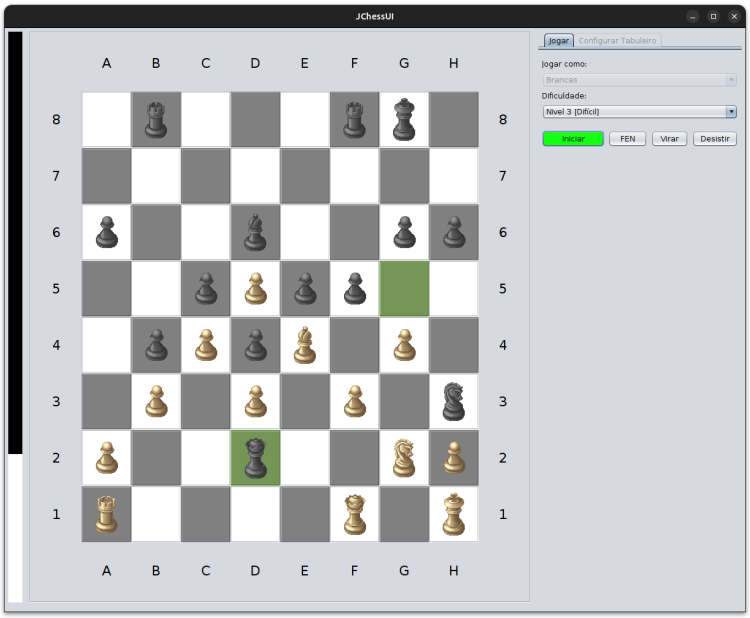
O formulário principal apresenta um tabuleiro que é o espaço onde ocorre a partida de xadrez. A direita existe uma TabView onde é possível selecionar o ambiente de jogo

(Jogar) ou o ambiente de configuração (Configuração Tabuleiro). No ambiente de jogo há uma combo box que permite ao jogador escolher se deseja jogar com as peças brancas ou pretas. Esta seleção deve ser feita antes do início da partida. Outra combo box está disponível para selecionar o nível de dificuldade da partida. Isso permite que o jogador ajuste o desafio de acordo com sua habilidade e experiência.

D.2.1 Ambiente de Jogo

Ainda na ambiente de jogo, existe um conjunto de botões de controle onde é possível iniciar um jogo ou desistir. A figura **??** apresenta demostra o estado inicial do tabuleiro com o botão Desistir desativado, dado que o jogo ainda não iniciou. Ao iniciar a partida não é possível clicar na tab para acessar o ambiente de configuração nem trocar a cor das peças escolhidas até que o jogador clique no botão Desistir, que agora está ativado, enquanto o botão Iniciar agora está desativado mas na cor verde, indicando que já foi dado início a partida, como na figura D.2 que demonstra uma situação jogo em andamento.

Figura D.2: Interface JChess durante uma partida



Exemplo de uma parida: As casas verdes denotam o úlimo movimento, das peças pretas: Dama G5 par D2

D.2.2 Botões de Controle de Jogo

- **Iniciar:** Este botão inicia a partida de xadrez. Ao iniciar a partida, o botão Desistir que estava inativado para a ficar ativado e o botão Iniciar passa para cor verde.
- **FEN:** Ao clicar neste botão, uma caixa de diálogo é exibida mostrando a posição atual do jogo no formato FEN (Forsyth-Edwards Notation).
- **Virar:** Este botão inverte o tabuleiro, colocando as peças brancas na parte superior e as pretas na parte inferior, ou vice-versa.
- **Desistir:** Este botão interrompe a partida atual e se inativa automaticamente equanto o botão Iniciar retoma a cor original.

À esquerda do tabuleiro, há uma barra de progressão que indica qual lado (branco ou preto) está em vantagem, de acordo com a avaliação do agente de xadrez. Esta barra fornece uma visualização rápida do estado atual do jogo. Por exemplo, na figura D.2, é possível verificar que, segundo o agente, as peças pretas estão em grande vantagem em relação as brancas.

D.2.3 Ambiente de Configuração do Tabuleiro

A tela de configuração do tabuleiro oferece uma interface detalhada para personalizar a disposição das peças e as condições de jogo. No centro da tela, há um tabuleiro de xadrez onde o usuário pode clicar para posicionar as peças. À direita do tabuleiro, encontra-se um painel onde é possível selecionar diferentes peças de cada cor. Ao selecionar uma peça e clicar no tabuleiro, o usuário pode desenhar qualquer configuração desejada. Há checkboxes que permitem indicar a situação de roque (O-O ou O-O-O) tanto para as peças brancas quanto para as pretas. Por meio de radiobuttons, o usuário pode indicar qual cor de peças fará o próximo movimento (brancas ou pretas).

D.2.4 Botões de Configuração

- **Limpar:** Este botão limpa a configuração atual, removendo todas as peças do tabuleiro.
- **Carregar FEN:** Permite carregar uma configuração de tabuleiro previamente defi-

nida ou configurada através de uma string FEN.

- **Posição Inicial:** Este botão redefine o tabuleiro para a posição inicial padrão do xadrez.
- **Mostrar FEN:** Exibe a string FEN correspondente à configuração atual do tabuleiro.
- **Avaliar:** Mostra uma pontuação de avaliação do tabuleiro configurado. Um valor positivo indica vantagem para as peças brancas, enquanto um valor negativo indica vantagem para as peças pretas.

Figura D.3: Interface JChess em configuração do tabuleiro

Ambiente de configuração do tabuleiro

Essa tela de configuração foi projetada para oferecer flexibilidade máxima ao usuário, permitindo a criação de cenários de jogo personalizados e a análise detalhada de diferentes posições.

D.3 Controle de Tempo

O controle de tempo é uma característica essencial nas interfaces de xadrez competitivas, que permite ao jogador definir o limite de tempo para uma partida, incluindo incrementos e decrementos de tempo. As interfaces geralmente incluem dois relógios visuais, um para cada jogador, que permitem acompanhar o tempo restante de maneira prática. Para que o xadrez competitivo seja justo, é importante que a interface apresente

um sistema de controle de tempo intuitivo e de fácil configuração, permitindo ao jogador selecionar modos como blitz (5 minutos) ou clássico (30 minutos ou mais) (Pietrow e Sadler, 2018). Optamos por não contemplar controle de tempo em nosso agente, tendo em vista que o objetivo principal é o aprendizado e não a pressão da competição em si.

D.4 Personalização e Configuração

Interfaces gráficas de xadrez modernas permitem uma ampla personalização visual e funcional, que abrange a escolha de temas de tabuleiro, estilos de peças e configurações de cores. Essas customizações ajudam a tornar a experiência mais confortável e permitem que o jogador personalize o ambiente de acordo com suas preferências. Adicionalmente, muitas interfaces permitem ajustar o nível de dificuldade do agente ou escolher diferentes motores para jogar, desde motores mais simples para iniciantes até agentes de nível de grande mestre, como o Stockfish e o AlphaZero (Vinyals et al., 2019).

D.5 Comandos Comuns em Interfaces de Xadrez

O protocolo UCI (Universal Chess Interface) define um conjunto de comandos para controlar agentes de xadrez, facilitando a interação entre o agente e as interfaces gráficas. Esses comandos permitem ao usuário configurar posições, definir parâmetros de análise e controlar a execução do agente em detalhes (RUSSELL; NORVIG, 2010). Abaixo, listamos alguns dos comandos mais comuns e suas funções principais:

D.5.1 Comando `position fen`

O comando `position fen` configura o tabuleiro em uma posição específica fornecida em Notação FEN (Forsyth-Edwards Notation), permitindo o carregamento de posições específicas para análise.

```
position fen rnbqkbnr/pppppppp/8/8/8/8/PPPPPPPP/RNBQKBNR w
                        KQkq - 0 1
```

Esse exemplo configura o tabuleiro na posição inicial de uma partida de xadrez. A notação FEN especifica a disposição das peças, o jogador a mover, e informações adi-

cionais, como roque e contagem de jogadas (LEVY; NEWBORN, 1991).

D.5.2 Comando `go depth 8`

O comando `go` inicia o cálculo do agente até uma profundidade especificada. A profundidade representa o número de meio-movimentos (*ply*) que o minimax deve avaliar.

```
go depth 8
```

Nesse exemplo, o agente analisa todos os possíveis movimentos até uma profundidade de 8 *ply*. Esse tipo de comando é útil para controlar a quantidade de processamento e tempo de análise (CAMPBELL; HOANE; HSU, 2002).

D.5.3 Comando `uci`

O comando `uci` inicializa o agente no modo UCI. Esse comando permite que a interface obtenha a lista de opções configuráveis do agente, incluindo parâmetros de análise e nível de dificuldade.

D.5.4 Comando `isready`

Este comando verifica se o agente está pronto para receber novos comandos. Após uma operação de análise extensa, `isready` confirma que o agente concluiu suas tarefas anteriores.

D.5.5 Comando `setoption name ...`

O comando `setoption` permite ajustar configurações específicas do agente, como profundidade de busca, uso de redes neurais e nível de habilidade.

```
setoption name Skill Level value 20
```

Esse exemplo configura o nível de habilidade do agente, ajustando sua competitividade e desempenho de acordo com as preferências do usuário (SADLER; REGAN, 2019).

D.5.6 Comando `stop`

O comando `stop` interrompe qualquer análise em andamento, sendo essencial para parar rapidamente o agente caso seja necessário ajustar uma configuração ou iniciar uma nova análise.

D.5.7 Comando `quit`

Esse comando encerra a sessão com o agente, liberando os recursos do sistema após a conclusão das análises ou jogos.

D.6 Desafios e Limitações das Interfaces de Xadrez

Apesar das inúmeras vantagens, as interfaces de xadrez enfrentam desafios no que diz respeito à usabilidade e à apresentação de informações complexas. Em partidas de alto nível, a visualização de dados e sugestões de movimento em tempo real pode sobrecarregar o jogador. Além disso, algumas interfaces exigem alto poder computacional para processar análises em profundidade, o que pode afetar o desempenho e a fluidez do jogo em dispositivos mais simples. Outro desafio envolve manter a simplicidade sem sacrificar funcionalidades avançadas, o que é especialmente difícil em interfaces que atendem tanto iniciantes quanto jogadores experientes (Sadler e Regan, 2019; Silver et al., 2018).

APÊNDICE E — ANÁLISE E DESEMPENHO DOS ALGORITMOS

A análise de algoritmos é essencial para compreender a eficiência e eficácia dos algoritmos em diferentes cenários e condições de uso, especialmente em contextos de jogos como o xadrez, onde respostas rápidas e precisas são críticas. Visualizaremos os principais aspectos envolvidos na avaliação de desempenho dos algoritmos, abordando métricas comuns, métodos de análise e desafios específicos.

E.1 Introdução à Análise de Desempenho

A análise de desempenho de algoritmos consiste em avaliar a quantidade de recursos necessários para que um algoritmo resolva determinado problema. Esses recursos incluem o tempo de execução e o uso de memória, conhecidos respectivamente como "tempo de execução" e "complexidade de espaço". Em jogos de tabuleiro complexos, como o xadrez, a avaliação do desempenho de algoritmos é particularmente importante, pois a capacidade de realizar cálculos profundos em tempo limitado é um fator decisivo na competitividade de um agente.

E.2 Métricas de Avaliação de Desempenho

As métricas mais comuns para a avaliação de desempenho dos algoritmos incluem:

- **Tempo de execução**: Mede o tempo necessário para que um algoritmo conclua suas operações para uma determinada entrada. Esse tempo pode ser avaliado em diferentes unidades, como segundos ou milissegundos, dependendo da natureza do problema e da precisão desejada (KNUTH, 1997).
- **Complexidade assintótica**: Representada pelas notações Big-O (O), Omega (Ω) e Theta (Θ), a complexidade assintótica descreve o comportamento do algoritmo quando o tamanho da entrada tende ao infinito. Por exemplo, um algoritmo de busca em árvore como o Minimax tem complexidade $O(b^d)$, onde b é o número médio de ramificações e d é a profundidade de busca.
- **Memória utilizada (complexidade de espaço)**: Refere-se ao espaço de memória que um algoritmo requer para executar. Em agentes de xadrez, o uso de tabelas de

transposição para armazenar posições já calculadas pode melhorar a eficiência em tempo, mas também aumenta a complexidade de espaço (CULBERSON, 1998).

- **Eficiência de CPU e cache**: Em algoritmos altamente otimizados para xadrez, como aqueles que implementam heurísticas e tabelas de transposição, o uso eficiente do cache da CPU e a minimização de falhas de cache também são métricas importantes (HEINZ, 2000).

E.3 Técnicas de Análise de Desempenho

Para avaliar o desempenho dos algoritmos, são utilizadas técnicas como análise empírica, análise assintótica e otimização de recursos computacionais.

E.3.1 Análise Empírica

A análise empírica envolve executar o algoritmo em diferentes conjuntos de dados e medir o tempo de execução e a memória utilizada. No caso de um agente de xadrez, testes podem ser realizados em posições complexas, como meio-jogo e final de jogo, para identificar em quais situações o algoritmo apresenta melhor ou pior desempenho. Essa análise empírica é frequentemente utilizada para verificar a precisão das estimativas teóricas de desempenho e detectar gargalos de processamento (AKL, 1987).

E.3.2 Análise Assintótica

A análise assintótica estuda o comportamento do algoritmo à medida que o tamanho de entrada cresce. Em agentes de xadrez, onde algoritmos como Minimax com poda alfa-beta são comuns, a análise assintótica ajuda a estimar como a complexidade do algoritmo cresce com o aumento da profundidade de busca. Por exemplo, em uma análise assintótica, a poda alfa-beta pode reduzir o número de nós a serem explorados de $O(b^d)$ para $O(b^{d/2})$ em casos ideais, praticamente dobrando a profundidade alcançável com a mesma quantidade de tempo (MARSLAND, 1986).

E.3.3 Profiling e Otimização de Código

O profiling é uma técnica que permite identificar as partes do código que mais consomem recursos. Em motores de xadrez, o profiling é particularmente útil para otimizar funções críticas, como a avaliação posicional e a geração de movimentos, de modo a reduzir o tempo de processamento por posição. Métodos de otimização incluem simplificação de cálculos matemáticos, substituição de estruturas de dados, e uso de algoritmos mais eficientes.

E.4 Desafios Específicos na Avaliação de Algoritmos de Xadrez

A avaliação de algoritmos para agentes de xadrez apresenta desafios específicos, incluindo a dificuldade de medir o impacto das heurísticas de avaliação e o balanceamento entre profundidade de busca e precisão das estimativas.

E.4.1 Complexidade da Heurística de Avaliação Posicional

A complexidade de uma heurística de avaliação posicional pode afetar significativamente o desempenho do algoritmo de busca. Quanto mais detalhada a heurística, maior o tempo de processamento, e isso pode comprometer o desempenho geral, especialmente em algoritmos como o Minimax, que exploram várias profundidades. Assim, encontrar um equilíbrio entre a precisão e o tempo de execução das heurísticas é um desafio importante (CAMPBELL; HOANE; HSU, 2002).

E.4.2 Profundidade de Busca e Recursos Computacionais

Em agentes de xadrez, a profundidade de busca tem um impacto direto na qualidade das decisões e no uso de recursos. Profundidades maiores aumentam a probabilidade de encontrar melhores movimentos, mas exigem um aumento exponencial de recursos computacionais. Esse compromisso entre profundidade de busca e tempo de execução é uma limitação importante que os desenvolvedores de agentes devem gerenciar.

E.4.3 Efeito das Tabelas de Transposição

As tabelas de transposição são estruturas de dados que armazenam posições já avaliadas para evitar reprocessamento, mas ocupam uma quantidade significativa de memória. Se uma tabela de transposição for insuficientemente dimensionada, pode ocorrer o efeito de "collision" (colisão), onde posições diferentes ocupam a mesma entrada na tabela, resultando em perda de eficiência (CULBERSON, 1998).

E.4.4 Poda alfa beta

A poda alfa-beta é uma técnica de otimização aplicada em algoritmos de busca de árvore, como o Minimax, cujo objetivo é reduzir o número de nós avaliados, aumentando a eficiência da busca. Esse mecanismo elimina ramos inteiros que não precisam ser explorados, uma vez que é possível provar que eles não influenciarão a decisão final do melhor movimento. A poda alfa-beta explora os limites superiores (beta) e inferiores (alfa) da função de avaliação durante a busca, evitando expandir nós cujos valores caem fora do intervalo permitido pelos limites (KNUTH; MOORE, 1975).

E.4.5 Funcionamento da Poda Alfa-Beta

Durante a execução do algoritmo de busca, a poda alfa-beta funciona ao manter duas variáveis principais: *alfa* e *beta*, que representam os valores mínimo e máximo aceitáveis de uma posição para o jogador atual. Se um nó gerar um valor fora desse intervalo, ele é podado (isto é, não é explorado), pois já se sabe que ele não influenciará a escolha do melhor movimento. Isso permite que a busca avance com menos recursos computacionais, alcançando profundidades maiores com o mesmo tempo disponível (CUNNINGHAM, 2012).

Em termos de complexidade, a poda alfa-beta pode melhorar o desempenho do Minimax de $O(b^d)$ para $O(b^{d/2})$, onde b é o fator de ramificação e d a profundidade da busca. No entanto, o benefício máximo dessa redução depende da ordenação dos nós.

E.4.6 Ordenação dos Nós para Eficiência da Poda

Para maximizar a eficiência da poda alfa-beta, a ordem de avaliação dos nós desempenha um papel crítico. Quando os melhores movimentos são avaliados primeiro, a chance de realizar cortes significativos no início da busca aumenta, reduzindo o número de nós explorados. Em implementações práticas de agentes de xadrez, técnicas como a *Movida de Captura* (que avalia primeiro as capturas ou jogadas taticamente fortes) e o uso de *tabelas de transposição* (para armazenar e reutilizar avaliações de posições já analisadas) são amplamente empregadas para melhorar essa ordenação (MARSLAND, 1986).

Algoritmos de ordenação como *Iterative Deepening* também são comuns em agentes de xadrez. Essa técnica explora a árvore de busca em profundidades sucessivas, refinando a ordem dos nós com base nos resultados de buscas anteriores. Dessa forma, a ordem de avaliação se torna cada vez mais precisa à medida que o algoritmo avança para profundidades maiores. Essa ordenação incremental melhora a eficácia da poda alfa-beta, tornando a busca mais rápida e eficiente (HEINZ, 2000).

APÊNDICE — REFERÊNCIAS

ACIEGO, R.; GARCÍA, L.; BETANCORT, M. The benefits of chess for the intellectual and social-emotional enrichment in schoolchildren. **The Spanish journal of psychology**, Cambridge University Press, v. 15, n. 2, p. 551–559, 2012.

AKL, S. G. **Parallel Sorting Algorithms**. [S.l.]: Academic Press, 1987.

BLOKDYK, G. **Chess Rating System A Complete Guide - 2020 Edition**. 5STARCOOKS, 2020. ISBN 9780655983316. Available from Internet: <https://books.google.com.br/books?id=EyFA0AEACAAJ>.

BROWNE, C. Bitboard methods for games. **ICGA journal**, IOS Press, v. 37, n. 2, p. 67–84, 2014. Available from Internet: <https://core.ac.uk/download/pdf/33500946.pdf>.

BURGOYNE, A. P. et al. "the relationship between cognitive ability and chess skill: A comprehensive meta-analysis": Corrigendum. **Intelligence**, Elsevier Science, 2018. Available from Internet: <https://psycnet.apa.org/record/2018-60092-012>.

CAMPBELL, M.; HOANE, A. J.; HSU, F.-h. Deep blue. **Artificial Intelligence**, Elsevier, v. 134, n. 1-2, p. 57–83, 2002. Available from Internet: <https://www.sciencedirect.com/science/article/pii/S0004370201001291>.

CAZAUX, J.; KNOWLTON, R. **A World of Chess: Its Development and Variations through Centuries and Civilizations**. McFarland, Incorporated, Publishers, 2017. ISBN 9780786494279. Available from Internet: <https://books.google.com.br/books?id=0O02DwAAQBAJ>.

CONTRIBUTORS, C. **Stockfish — Chess Programming Wiki**. 2024. Acessado em: 2 out. 2024. Available from Internet: <https://www.chessprogramming.org/Stockfish>.

CONTRIBUTORS, C. Time management in chess — chess programming wiki. **Chess Programming Wiki**, 2024. Acessado em: 30 setembro 2024. Available from Internet: <https://www.chessprogramming.org/Time_Management>.

CONTRIBUTORS, W. **AlphaZero — Wikipedia, The Free Encyclopedia**. 2024. Acessado em: 2 out. 2024. Available from Internet: <https://en.wikipedia.org/wiki/AlphaZero>.

CONTRIBUTORS, W. **Elo rating system — Wikipedia, The Free Encyclopedia**. 2024. Acessado em: 13 out. 2024. Available from Internet: <https://en.wikipedia.org/wiki/Elo_rating_system>.

CONTRIBUTORS, W. **Houdini (chess) — Wikipedia, The Free Encyclopedia**. 2024. Acessado em: 7 out. 2024. Available from Internet: <https://en.wikipedia.org/wiki/Houdini_(chess)>.

CONTRIBUTORS, W. **Komodo (chess) — Wikipedia, The Free Encyclopedia**. 2024. Acessado em: 7 out. 2024. Available from Internet: <https://en.wikipedia.org/wiki/Komodo_(chess)>.

CONTRIBUTORS, W. **Stockfish (chess) — Wikipedia, The Free Encyclopedia**. 2024. Acessado em: 2 out. 2024. Available from Internet: <https://en.wikipedia.org/wiki/Stockfish_(chess)>.

CULBERSON, J. C. Transposition table management for controlled resource allocation. **Artificial Intelligence**, v. 98, n. 1-2, p. 99–109, 1998.

CUNNINGHAM, P. Heuristic techniques in game tree search. **Proceedings of the ICGA Journal**, 2012.

DAVIDSON, H. A. **A Short History of Chess**. [S.l.]: Greenwood Press, 1949.

DVORETSKY, M. **School of Chess Excellence: Endgame analysis**. Edition Olms, 2001. (Progress in chess). ISBN 9783283004163. Available from Internet: <https://books.google.com.br/books?id=ztXdRgAACAAJ>.

EALES, R. **Chess, the History of a Game**. [S.l.]: University Microfilms, 1993. ISBN 9780835734790.

ELO, A. **The Rating of Chessplayers: Past and Present**. Ishi Press International, 2008. ISBN 9780923891275. Available from Internet: <https://books.google.com.br/books?id=syjcPQAACAAJ>.

ELO, A. R. **The rating of chessplayers, past and present**. Arco Pub. Co., 1978. ISBN 9780668047210. Available from Internet: <https://books.google.com.br/books?id=8pMnAQAAMAAJ>.

FIDE. **FIDE Laws of Chess: For Competitions Starting from 1 January 2022**. [S.l.]: Fédération Internationale des Échecs, 2021.

FIDE. **FIDE Handbook of Chess**. 2023. <https://handbook.fide.com/chapter/E012023>.

FIDE. **International Chess Federation**. 2024. Acessado em: 28 set. 2024. Available from Internet: <https://www.fide.com>.

FISKE, D. W. **Chess in Iceland and in Icelandic Literature**. Florence: Florentine Typographical Society, 1913.

FISKE, W. **Chess in Iceland and in Icelandic Literature with Historical Notes on Other Table-Games**. [S.l.]: Florence Press, 1905.

FRIEDEL, F. **Judit Polgar to retire from competitive chess**. 2024. Acessado em: 28 set. 2024. Available from Internet: <https://en.chessbase.com/post/judit-polgar-to-retire-from-competitive-chess>.

GLICKMAN, M. E. Chess rating systems. **American Chess Journal**, v. 3, p. 59–102, 1995. Available from Internet: <https://www.glicko.net/research/acjpaper.pdf>.

GOBET, F.; CAMPITELLI, G. Educational benefits of chess instruction: A critical review. In: **Chess and education: Selected essays from the Koltanowski conference**. [S.l.: s.n.], 2006. p. 124–143.

HARREVELD, F. V.; WAGENMAKERS, E.-J.; MAAS, H. L. V. D. The effects of time pressure on chess skill: An investigation into fast and slow processes underlying expert performance. **Psychological research**, Springer, v. 71, p. 591–597, 2007. Available from Internet: <https://link.springer.com/article/10.1007/s00426-006-0076-0>.

HEINZ, E. A. Scalable search in computer chess. **ICGA Journal**, v. 23, n. 4, p. 203–218, 2000.

HOOPER, D.; WHYLD, K.; WHYLD, K. **The Oxford Companion to Chess**. Oxford University Press, 1987. (Oxford Paperbacks). ISBN 9780192819864. Available from Internet: <https://books.google.com.br/books?id=h6crAQAAMAAJ>.

ILIESCU, D. M. D. The impact of artificial intelligence on the chess world. **JMIR serious games**, JMIR Publications Inc., Toronto, Canada, v. 8, n. 4, p. e24049, 2020. Available from Internet: <https://games.jmir.org/2020/4/e24049>.

JOHNSON, R. **The Effect Of Time Pressure On Chess Strategy**. 2024. Acessado em: 30 set. 2024. Available from Internet: <https://beginnerschessstrategies.com/the-effect-of-time-pressure-on-chess-strategy>.

JONES, P. The evolution of stockfish and its forks. **Chess Engine Quarterly**, v. 25, n. 4, p. 150–170, 2022.

KANNAN, P. Magic move-bitboard generation in computer chess. **Preprint**, 2007. Available from Internet: <http://pradu.us/old/Nov27_2008/Buzz/research/magic/Bitboards.pdf>.

KASPAROV, G. **Garry Kasparov on My Great Predecessors, Part Four**. Everyman Chess, 2020. (Everyman chess, p. 4). ISBN 9781781945186. Available from Internet: <https://books.google.com.br/books?id=zvqkxQEACAAJ>.

KAZEMI, F.; YEKTAYAR, M.; ABAD, A. M. B. Investigating the impact of chess play on developing meta-cognitive ability and math problem-solving power of students at different levels of education. **Procedia - Social and Behavioral Sciences**, v. 32, p. 372–379, 2012. ISSN 1877-0428. The 4th International Conference of Cognitive Science. Available from Internet: <https://www.sciencedirect.com/science/article/pii/S1877042812000572>.

KEENE, R. **Chess: An Illustrated History**. [S.l.]: Simon Schuster, 1999. ISBN 9780671708146.

KEENE, R. **How to Play the Opening in Chess**. New York: Sterling Publishing, 2002.

KLEIN, M. **Google's AlphaZero Destroys Stockfish In 100-Game Match**. 2017. Acessado em: 2 out. 2024. Available from Internet: <https://www.chess.com/news/view/google-s-alphazero-destroys-stockfish-in-100-game-match>.

KNUTH, D. E. **The Art of Computer Programming**. 2nd. ed. [S.l.]: Addison-Wesley, 1997.

KNUTH, D. E.; MOORE, R. W. An analysis of alpha-beta pruning. **Artificial Intelligence**, Elsevier, v. 6, n. 4, p. 293–326, 1975.

LEVY, D.; NEWBORN, M. **How Computers Play Chess**. [S.l.]: Computer Science Press, 1991.

LINDER, I. **The Art of Chess Pieces**. H.G.S. Publishers, 1994. ISBN 9785758803868. Available from Internet: <https://books.google.com.br/books?id=J1wOAgAACAAJ>.

LINDER, I.; LINDER, V. **Wilhelm Steinitz: 1st World Chess Champion**. Russell Enterprises, Incorporated, 2014. (World Chess Champion). ISBN 9781936490936. Available from Internet: <https://books.google.com.br/books?id=7pBXDwAAQBAJ>.

LUDOTEKA. **Imagem Chaturanga**. 2024. Acessado em: 1 out. 2024. Available from Internet: <https://www.ludoteka.com/clasika/chaturanga-en.html>.

MADAKE, J. et al. Chess ai: Machine learning and minimax based chess engine. **IEEE Conference Publication**, p. 1–6, 2023. Available from Internet: <https://ieeexplore.ieee.org/document/10080746>.

MARSLAND, T. A. A review of game-tree pruning. **ICCA Journal**, v. 9, n. 1, p. 3–19, 1986.

MARSLAND, T. A. **Computer chess and search**. Department of Computing Science, University of Alberta, 1991. Available from Internet: <https://webdocs.cs.ualberta.ca/~tony/RecentPapers/encyc.mac-1991.pdf>.

MURRAY, H. J. R. **A History of Chess**. [S.l.]: Oxford University Press, 1913.

PIETROW, I.; SADLER, M. **Understanding the Chessboard: User Interface Design for Competitive Play**. [S.l.]: Chess UI Press, 2018.

RUSSELL, S.; NORVIG, P. **Artificial Intelligence: A Modern Approach**. [S.l.]: Pearson, 2010.

RUSSELL, S.; NORVIG, P. **Artificial Intelligence: A Modern Approach**. [S.l.]: Pearson, 2016.

SADLER, M.; REGAN, N. **Game Changer: AlphaZero's Groundbreaking Chess Strategies and the Promise of AI**. [S.l.]: New in Chess, 2019.

SADMINE, Q. A.; HUSNA, A.; MüLLER, M. Stockfish ornbsp;leela chess zero? anbsp;comparison against endgame tablebases. In: **Advances in Computer Games: 18th International Conference, ACG 2023, Virtual Event, November 28–30, 2023, Revised Selected Papers**. Berlin, Heidelberg: Springer-Verlag, 2024. p. 26–35. ISBN 978-3-031-54967-0. Available from Internet: <https://doi.org/10.1007/978-3-031-54968-7_3>.

SALA, G. et al. Does chess instruction improve mathematical problem-solving ability? two experimental studies with an active control group. **Learning and Behavior**, Springer, v. 45, n. 4, p. 414–421, 2017. Available from Internet: <https://link.springer.com/article/10.3758/s13420-017-0280-3>.

SCHAEFER, M. **Women in Chess: Breaking Barriers and Making History**. 2024. Available from Internet: <https://digitalbooktalk.com/women-in-chess-breaking-barriers-and-making-history/>.

SCHOLZ, M.; ROCHA, A. F. da; BÖTTCHER, Y. More than a mere game: a longitudinal study on the moderating effect of chess training on math skills development in children. **International Journal of Science and Mathematics Education**, Springer, v. 14, p. 169–181, 2016.

SHANNON, C. Programming a computer for playing chess. **Philosophical Magazine**, 1950.

SHIBUT, M. **Paul Morphy and the Evolution of Chess Theory**. Dover Publications, 2012. (Dover Chess). ISBN 9780486149875. Available from Internet: <https://books.google.com.br/books?id=AVglRy1FBckC>.

SILVER, D. et al. A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. **Science**, American Association for the Advancement of Science, v. 362, n. 6419, p. 1140–1144, 2018.

SILVER, D. et al. A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. **Science**, v. 362, n. 6419, p. 1140–1144, 2018. Available from Internet: <https://www.science.org/doi/abs/10.1126/science.aar6404>.

SILVER DAVID; HUBERT, T. S. J. A. I. L. M. G. A. L. M. S. L. K. D. G. T. L. T. S. K. H. D. Mastering chess and shogi by self-play with a general reinforcement learning algorithm. **Science**, 2017. Available from Internet: <https://www.idi.ntnu.no/emner/it3105/materials/neural/silver-2017b.pdf>.

SOLTIS, A.; SOLTIS, A. **The Inner Game of Chess: How to Calculate and Win**. Mongoose Press, 2014. ISBN 9781936277605. Available from Internet: <https://books.google.com.br/books?id=Y7O6rQEACAAJ>.

SOLTIS, A. E. **Chess**. 2024. Available from Internet: <https://www.britannica.com/topic/chess>.

STAUNTON, H. **The Chess-Player's Handbook**. Henry G. Bohn, 1849. Available from Internet: <https://books.google.com.br/books?id=q7BAAAAAcAAJ>.

TANNOUS, S. Avoiding rotated bitboards with direct lookup. **ICGA Journal**, IOS Press, v. 30, n. 2, p. 85–91, 2007. Available from Internet: <https://ar5iv.labs.arxiv.org/html/0704.3773>.

Wikipedia contributors. **Mechanical Turk — Wikipedia, The Free Encyclopedia**. 2024. [Online; accessed 01-October-2024]. Available from Internet: <https://en.wikipedia.org/wiki/Mechanical_Turk>.

WINIEWSKA, J.; WOJCIK, P. Machine learning methods in game of chess implementation. **Computer Science and Mathematical Modelling**, v. 0, n. 13-14/2021, p. 61–69, 2022.

YALOM, M. **Birth of the Chess Queen: A History**. HarperCollins, 2009. ISBN 9780061913426. Available from Internet: <https://books.google.com.br/books?id=VgYHO1_-ApsC>.

ZOBRIST, A. L. A new hashing method with application for game playing. **ICGA Journal**, IOS Press, v. 13, n. 2, p. 69–73, 1990. Available from Internet: <https://minds.wisconsin.edu/bitstream/handle/1793/57624/TR88.pdf?sequence=1&isAllowed=y>.