NÍLTON CÉSAR FURTADO CANTO

**UMA ABORDAGEM EVOLUTIVA PARA IDENTIFICAÇÃO
DE PROCEDIMENTOS DE RACIOCÍNIO HUMANO**

São Paulo

2008

NÍLTON CÉSAR FURTADO CANTO

**UMA ABORDAGEM EVOLUTIVA PARA IDENTIFICAÇÃO
DE PROCEDIMENTOS DE RACIOCÍNIO HUMANO**

**Tese apresentada a Escola Politécnica
da Universidade de São Paulo para
obtenção do Título de Doutor em
Engenharia Elétrica**

São Paulo

2008

NÍLTON CÉSAR FURTADO CANTO

**UMA ABORDAGEM EVOLUTIVA PARA IDENTIFICAÇÃO
DE PROCEDIMENTOS DE RACIOCÍNIO HUMANO**

**Tese apresentada à Escola Politécnica
da Universidade de São Paulo para
obtenção do Título de Doutor em
Engenharia Elétrica**

**Área de concentração:
Sistemas Eletrônicos**

Orientador: Prof. Dr. Marcio Lobo Netto

São Paulo

2008

Este exemplar foi revisado e alterado em relação à versão original, sob responsabilidade única do autor e com a anuência de seu orientador.

São Paulo, 19 de dezembro de 2008.

Assinatura do autor: _____

Assinatura do orientador: _____

FICHA CATALOGRÁFICA

Edição Revisada

Canto, Nilton César Furtado

Uma abordagem evolutiva para identificação de procedimentos de raciocínio humano / N.C.F. Canto. -- ed.rev.-- São Paulo, 2008.

136 p.

Tese (Doutorado) - Escola Politécnica da Universidade de São Paulo. Departamento de Engenharia de Sistemas Eletrônicos.

**1.Algoritmos genéticos 2.Cognição 3.Inteligência artificial
I.Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia de Sistemas Eletrônicos II.t.**

DEDICATÓRIA

Aos meus pais, Waldemar e Nila.

À minha esposa Fátima.

Aos meus filhos Juliana e Pedro.

AGRADECIMENTOS

Ao meu orientador, Prof. Dr. Marcio Lobo Netto, pela segura orientação e apoio incansável à realização deste trabalho.

Gostaria de agradecer aos professores membros da banca de qualificação e defesa, Dr. Emilio Del Moral Hernandez, Dr. Flavio Soares Correa da Silva, Dr. João José Neto, Dr. Marco Túlio Carvalho de Andrade e Dr. Nizam Omar, que aceitaram participar de minha avaliação e com isto contribuir para a finalização deste trabalho.

Um agradecimento especial a todos os professores da Escola Politécnica, pelos valiosos ensinamentos indispensáveis à realização deste trabalho.

Aos colegas doutorandos Daniel Costa Paiva e Luciene Cristina Rinaldi Rodrigues pelo incentivo e colaboração na preparação da minha defesa.

A minha irmã, Cristiane, pelo incentivo e pela leitura e revisão de vários textos.

Aos designers gráficos e meus colegas de trabalho, Flávio e Félix, pela ajuda na produção de várias imagens.

Aos funcionários da secretária de pós-graduação, pelo sempre gentil e eficiente atendimento.

A todos aqueles que de forma direta ou indireta colaboraram para a realização desta tese, meu sincero agradecimento.

RESUMO

Neste trabalho, investigou-se a utilização de algoritmos evolutivos para identificação de procedimentos de raciocínio utilizados por humanos na construção de soluções para uma classe de problemas cuja principal característica é a utilização de raciocínio dedutivo. Para isso, utilizou-se uma abordagem que explora os diferentes níveis de complexidade do problema, partindo da análise das estratégias apresentadas por jogadores humanos. Foram realizados diversos ensaios que evoluíram primeiramente, para um modelo de solução puramente combinatória guiada por um algoritmo genético e independente do jogador humano, até atingir um modelo que procura identificar um procedimento de solução que guarde semelhanças com os procedimentos apresentados pelos jogadores humanos. Como resultado, apresentou-se um algoritmo denominado Classificador Genético –um sistema de operadores guiado por um algoritmo genético– capaz de identificar procedimentos de raciocínio para solução de combinações específicas do problema proposto. Os ensaios permitiram ainda identificar conjuntos de operadores que se combinados corretamente, através de um mecanismo que simule a tomada de decisão do jogador humano, são capazes de aumentar o potencial de identificação de soluções do algoritmo proposto. O estudo também revelou a importância dos mecanismos de memória no processo de solução do problema e as dificuldades em manipular operadores gerais com métodos puramente evolutivos. Foi possível ainda identificar de que modo jogadores humanos tratam os fatores relacionados à diversidade de possíveis encaminhamentos no processo decisório, que afetam a solução do problema proposto.

Palavras-chaves – algoritmos evolutivos, sistemas classificadores, algoritmos genéticos, sistemas cognitivos, inteligência artificial, raciocínio humano.

ABSTRACT

In this work we investigated the use of evolutionary algorithms to identify logic procedures used by humans in the construction of solutions of a class of problems in which the main characteristic is the use of deductive reasoning. In order to do that it was used an approach that explores the problem's different levels of complexity, starting from the strategies analysis presented by human players. Several experiments were carried out where at first moment used a model of solution that is strictly combinatorial guided by a genetic algorithm and independent of the human player that evolved to a model that tries to identify a solution procedure that maintains the similarities with the procedures presented by human players. As a result, we presented an algorithm denominated Genetic Classifier - a system of rules guided by a genetic algorithm - able to identify reasoning procedures for solution of specific combinations of the proposed problem. Moreover, the experiments allowed identifying clusters of rules that if combined correctly, through a mechanism that simulates the decision making performed by a human player, are capable of increasing the potential to identify the solutions of the proposed algorithm. The study also revealed the importance of the memory's mechanism in the process of solving the proposed problem and the difficulties to manipulate general rules with regular evolutionary methodologies. It was also possible to identify the way human players deal with the factors related to the diversity of possible directions in the decision process.

Keywords: evolutionary algorithms, classifier systems, genetic algorithms, cognitive systems, artificial intelligence, human reasoning.

LISTA DE ILUSTRAÇÕES

Figura 3.1	- Detalhes de um lance para versão clássica do <i>Mastermind</i>	15
Figura 3.2	- Relação entre jogadores e estratégias.	18
Figura 3.3	- Exemplo de partida em que o jogador utiliza uma estratégia denominada Ótima.	19
Figura 4.1	- Diagrama de fluxo de um algoritmo genético simples.	25
Figura 4.2	- Representação inteira (a) e binária (b) de um indivíduo.	27
Figura 4.3	- Processo de seleção pelo método da roleta para uma população de 4 indivíduos.	29
Figura 4.4	- (A) Dois pais selecionados para passarem pelo processo de recombinação e o ponto de corte, escolhido ao acaso, entre os genes 3 e 4. (B) Filhos provenientes dessa recombinação.	31
Figura 4.5	- (A) Duas árvores representando os indivíduos de uma população e seus respectivos pontos de corte. (B) Resultado da operação de <i>crossover</i> .	33
Figura 4.6	- (A) Um indivíduo da população e o nodo escolhido para mutação. (B) Resultado da mutação, em que o operador multiplicação foi substituído pelo operador adição.	34
Figura 5.1	- Resumo dos ensaios realizados.	36
Figura 5.2	- Principais abordagens utilizadas para neste trabalho.	37
Figura 5.3	- Interface do sistema desenvolvido para realização dos ensaios	43
Figura 5.4	- Módulos do sistema e suas interconexões.	44
Figura 5.5	- Lances de uma partida executada pelo algoritmo genético.	47
Figura 5.6	- Geração da população inicial contendo n indivíduos.	48
Figura 5.7	- Exemplo de cálculo do <i>fitness</i> para um indivíduo algoritmo genético.	50
Figura 5.8	- Exemplo de aplicação do <i>crossover</i> condicional.	53
Figura 5.9	- Exemplo de aplicação do <i>crossover</i> condicional para o caso em que todas as cores do código já foram descobertas.	53

Figura 5.10	- Cálculo do <i>fitness</i> para um indivíduo da população de soluções.	58
Figura 5.11	- Representação gráfica da medida da similaridade entre os lances de indivíduo qualquer da população de soluções e os lances do jogador humano.	59
Figura 5.12	- (A) mostra o processo de recombinação e o ponto de corte, escolhido ao acaso, entre os genes 3 e 4. (B) Filhos provenientes dessa recombinação.	60
Figura 6.1	- Distribuição do número de lances necessários para solução do jogo em 20 partidas realizadas pelo jogador 1 (iniciante).	62
Figura 6.2	- Variação do desempenho do jogador 1 (iniciante) em 20 partidas.	63
Figura 6.3	- Distribuição do número de lances necessários para solução do jogo em 20 partidas realizadas pelo jogador 2 (intermediário).	63
Figura 6.4	- Variação do desempenho do jogador 2 (intermediário) em 20 partidas.	64
Figura 6.5	- Distribuição do número de lances necessários para solução do jogo em 20 partidas realizadas pelo jogador 3 (avançado).	64
Figura 6.6	- Variação do desempenho do jogador 3 (avançado) em 20 partidas.	65
Figura 6.7	- Resultado da execução do algoritmo genético versão AG2.	69
Figura 6.8	- Resultado da execução do algoritmo genético versão AG3.	70
Figura 6.9	- Procedimento obtido depois de 100 gerações pelo CG3S.	77
Figura 6.10	Comparação entre os resultados dos algoritmos CG1S (ensaio III-a) e CG1M (ensaio III-b) obtidos em 100 partidas com o mesmo código secreto.	81
Figura 6.11	Comparação entre os resultados dos algoritmos CG3S (ensaio III-a) e CG3M (ensaio III-b) obtidos em 100 partidas com o mesmo código secreto e o mesmo número de lances.	84
Figura 6.12	- Lances gerados a partir da aplicação do operador i.	85

Figura 6.13	- Representação da árvore de decisão do jogador humano (A).	86
Figura 6.14	- Representação da árvore de decisão do jogador humano (B).	87
Figura 6.15	- Execução das estratégias 1 e 2 da tabela 7.16.	92
Figura 6.16	- Execução de um conjunto de operadores que soluciona o problema em cinco lances, mas não configura uma estratégia.	92
Figura 6.17	- Representação da interseção de soluções para o código (4,1,5).	94
Figura 6.18	- Representação da interseção das soluções encontradas para os códigos gerados a partir da tripla (4,1,5).	96
Figura 6.19	- Solução apresentada pelo jogador humano para solução do código em cinco lances.	97
Figura 6.20	- Execução do conjunto de operadores do jogador virtual (lado esquerdo) que reproduz os lances do jogador humano (lado direito) para a solução do código (5,2,1).	97
Figura 6.21	- Execução dos lances do jogador virtual (lado esquerdo) que reproduz a pontuação do jogador humano (lado direito).	98
Figura 7.1	- Indivíduos formados por conjuntos de operadores.	105
Figura 7.2	- Máquina de estados para seleção de operadores.	106
Figura 7.3	- Máquina de estado para seleção automática de operadores.	107
Figura A.1	- Distribuição do número de lances necessários para solução do jogo em 100 partidas utilizando o Algoritmo Genético AG1 com função de similaridade baseada na função de ponderação e uma população de 100 indivíduos.	119
Figura A.2	- Distribuição do número de lances necessários para solução do jogo em 100 partidas utilizando o Algoritmo Genético AG1 com função de similaridade calculada a partir da distância entre pontos e uma população de 100 indivíduos.	120

LISTA DE TABELAS

Tabela 3.1	- Termos utilizados na descrição do jogo <i>Mastermind</i> .	16
Tabela 4.1	- Estrutura básica de um Algoritmo Evolutivo.	23
Tabela 5.1	- Descrição dos três jogadores avaliados em 20 partidas (ensaio I).	38
Tabela 5.2	- Descrição do ensaio II - Algoritmo Genético.	39
Tabela 5.3	- Estrutura de um arquivo texto contendo os dados referentes a duas partidas de um mesmo jogador.	45
Tabela 5.4	- Tabela de codificação de cores para o <i>Mastermind</i> .	45
Tabela 5.5	- Tabela auxiliar para o cálculo do <i>fitness</i> .	50
Tabela 5.6	- Algoritmo executado pelo módulo classificador genético.	55
Tabela 5.7	- Lista de operadores para solução do <i>Mastermind</i> .	56
Tabela 6.1	- Resumo dos ensaios e algoritmos.	61
Tabela 6.2	- Desempenho dos jogadores em 20 partidas sucessivas.	67
Tabela 6.3	- Parâmetros e resultados de simulação para o <i>Mastermind</i> (4x6).	68
Tabela 6.4	- Parâmetros e resultados de simulação para o <i>Mastermind</i> (5x8).	68
Tabela 6.5	- Parâmetros e resultados de simulação com o CG1.	73
Tabela 6.6	- Descrição da sequência de operadores gerada pelo CG1S na simulação número 1.	73
Tabela 6.7	- Parâmetros e resultados de simulação com o CG2S.	75
Tabela 6.8	- Parâmetros e resultados de simulação com o CG3S.	76
Tabela 6.9	- Análise da similaridade entre os lances do jogador humano e virtual.	78
Tabela 6.10	- Parâmetros e resultados da simulação do algoritmo CG1M.	80

Tabela 6.11	- Parâmetros e resultados da simulação CG2M.	82
Tabela 6.12	- Parâmetros e resultados da simulação do algoritmo CG3M.	83
Tabela 6.13	- Análise da pontuação possível de ser obtida em uma partida.	88
Tabela 6.14	- Conjunto de operadores instanciados.	88
Tabela 6.15	- Parâmetros e resultados de simulação do algoritmo CG1E.	91
Tabela 6.16	- Análise da formação de <i>clusters</i> em dez soluções para o código (4,1,5).	93
Tabela 6.17	- Soluções para a as seis possíveis combinações do código (4,1,5).	94
Tabela 6.18	- Análise da formação de <i>clusters</i> nas soluções encontradas para os seis códigos gerados a partir da tripla (4,1,5).	95
Tabela 6.19	- Análise da similaridade entre os lances do jogador humano e virtual.	98
Tabela A.1	Dados estatísticos comparando as duas metodologias utilizadas para o cálculo do <i>fitness</i> dos indivíduos de uma população.	120

LISTA DE ABREVIATURAS

AG	Algoritmo Genético
CBT	<i>Computer Based Training</i>
CE	Computação Evolutiva
EE	Estratégias Evolutivas
CG	Classificador Genético
GPS	<i>General Problem Solver</i>
IA	Inteligência Artificial
PE	Programação Evolutiva
PG	Programação Genética
RNA	Rede Neural Artificial
SC	Sistema Classificador
SIs	Sistemas inteligentes
TM	Taxa de Mutação

SUMÁRIO

RESUMO	i
ABSTRACT	ii
LISTA DE ILUSTRAÇÕES	iii
LISTA DE TABELAS	vi
LISTA DE ABREVIATURAS	viii

CAPÍTULO 1

CONSIDERAÇÕES INICIAIS	1
1.1 JUSTIFICATIVA PARA A ESCOLHA DO TEMA	2
1.2 OBJETIVOS	4
1.2.1 Objetivos Gerais	4
1.2.2 Objetivos Específicos	4
1.3 ORGANIZAÇÃO DO TRABALHO	4

CAPÍTULO 2

COGNIÇÃO E SOLUÇÃO DE PROBLEMAS	5
2.1 HISTÓRICO DO DESENVOLVIMENTO DA CIÊNCIA COGNITIVA	5
2.2 TEORIAS DA APRENDIZAGEM	6
2.2.1 Epistemologia Genética de Piaget	7
2.2.2 Aprendizagem Significativa de Ausubel	8
2.3 APRENDIZAGEM EM SISTEMAS DE INTELIGÊNCIA ARTIFICIAL	9
2.4 ABORDAGENS COGNITIVAS PARA SOLUÇÃO DE PROBLEMAS	9
2.5 PROCEDIMENTOS PARA SOLUÇÃO DE PROBLEMAS	13

CAPÍTULO 3

JOGOS DE RACIOCÍNIO	14
3.1 <i>MASTERMIND</i>	14
3.2 <i>MASTERMIND</i> : O PROBLEMA.	16
3.3 CARACTERÍSTICAS COGNITIVAS DO JOGADOR	17
3.4 ALGORITMOS E DESEMPENHOS	18

CAPÍTULO 4

COMPUTAÇÃO EVOLUTIVA	21
4.1 BASE BIOLÓGICA	21
4.2 PRINCÍPIOS DE FUNCIONAMENTO	23
4.3 PRINCIPAIS PARADIGMAS EVOLUTIVOS	24
4.4 ALGORITMOS GENÉTICOS	24
4.4.1 Métodos de Busca e Otimização	25
4.4.2 AG – Implementação de Algoritmos Genéticos	26
4.4.2.1 Codificação dos Indivíduos	27
4.4.2.2 População Inicial	28
4.4.2.3 Cálculo da Função de Avaliação (Fitness)	28
4.4.2.4 Métodos de Seleção	28
4.4.2.5 Elitismo	30
4.4.2.6 Recombinação (crossover)	30
4.4.2.7 Mutação	31
4.4.2.8 Critérios de Parada do algoritmo genético	31
4.4.3 O Uso de AGs na Solução de Problemas	32
4.5 SISTEMAS CLASSIFICADORES	32
4.6 PROGRAMAÇÃO GENÉTICA	33
4.7 ESTRATÉGIAS EVOLUTIVAS	34
4.8 PROGRAMAÇÃO EVOLUTIVA	34

CAPÍTULO 5

MÉTODOS E ENSAIOS	36
5.1 INTRODUÇÃO	36
5.2 JOGADOR HUMANO	38
5.3 JOGADOR VIRTUAL	38
5.4 DESCRIÇÃO DO SISTEMA PROPOSTO	42
5.4.1 Módulo Especialista	44
5.4.2 Módulo de Controle	44
5.4.3 Módulo algoritmo genético	45
5.4.4 Módulo classificador genético	46
5.5 DETALHES DO ALGORITMO GENÉTICO	46
5.5.1 Codificação	47
5.5.2 Seleção	48
5.5.3 Elitismo	48
5.5.4 Reprodução	48
5.5.5 Função Fitness	49

5.5.6	Crossover Condicional	52
5.5.7	Mutação	53
5.5.8	Heurísticas	54
5.6	DETALHES DO CLASSIFICADOR GENÉTICO	54
5.6.1	Algoritmo do Classificador Genético	54
5.6.2	População Inicial	56
5.6.3	Função <i>Fitness</i>	57
5.6.3.1	CG3S - Cálculo da Função <i>Fitness</i>	57
5.6.4	Crossover	59
5.6.5	Mutação	60
5.6.6	Crítérios de Parada	60

CAPÍTULO 6

RESULTADOS E AVALIAÇÕES DOS ENSAIOS		61
6.1	ENSAIO I – ANÁLISE DO DESEMPENHO DE HUMANOS	62
6.2	ENSAIO II – ALGORITMO GENÉTICO	67
6.2.1	AG1 (Algoritmo Genético) – Análise dos Resultados	67
6.2.2	AG2 (Algoritmo Genético) – Análise dos Resultados	69
6.2.3	AG3 (Algoritmo Genético) – Análise dos Resultados	70
6.3	ENSAIO III-a – CLASSIFICADOR GENÉTICO (operadores sem memória)	71
6.3.1	CG1S – Análise dos Resultados	71
6.3.2	CG2S – Análise dos Resultados	74
6.3.3	CG3S – Análise dos Resultados	75
6.4	ENSAIO III-b – CLASSIFICADOR GENÉTICO (operadores com memória)	79
6.4.1	CG1M – Análise dos Resultados	80
6.4.2	CG2M – Análise dos Resultados	82
6.4.3	CG3M – Análise dos Resultados	83
6.5	ENSAIO IV - CLASSIFICADOR GENÉTICO (conjunto de operadores instanciados)	87
6.5.1	CG1E – Análise dos Resultados	90
6.5.2	CG2E – Análise dos Resultados	96
6.5.3	CG3E – Análise dos Resultados	98

CAPÍTULO 7

CONCLUSÃO E TRABALHOS FUTUROS		101
7.1	DISCUSSÃO DOS RESULTADOS	101

7.2	CONCLUSÕES FINAIS	104
7.3	TRABALHOS FUTUROS	105
BIBLIOGRAFIA CONSULTADA		108
BIBLIOGRAFIA COMPLEMENTAR		115
APÊNDICE A		
MÉTODOS DE CÁLCULO PARA A FUNÇÃO <i>FITNESS</i>		117
A.1	<i>FITNESS</i> CONDICIONAL	117
A.2	COMPARAÇÃO ENTRE MÉTODOS PARA CÁLCULO DO <i>FITNESS</i>	119

CAPÍTULO 1

CONSIDERAÇÕES INICIAIS

A construção de sistemas inteligentes (SIs) tem sido objeto de inúmeras pesquisas ao longo das últimas décadas. Avanços nas áreas de Neurociência e Ciência Cognitiva têm contribuído para o desenvolvimento de arquiteturas que procuram modelar a inteligência em sistemas artificiais a partir de analogias com o processo cognitivo humano [NETTO2007; PYLYSHYN1980]. O principal desafio dos modelos propostos está na construção de sistemas com flexibilidade e autonomia cognitiva, capazes de resolver problemas ou tomar decisões para as quais não foram explicitamente programados. Embora não exista uma definição universalmente aceita para o conceito de inteligência, pode-se enumerar uma série de tarefas e habilidades desenvolvidas pelo homem e associadas à presença dela: flexibilidade, adaptabilidade, capacidade de resolver problemas, agir em ambientes complexos, operar em tempo real, usar símbolos e abstrações, usar linguagem natural, aprender com a experiência, viver em sociedade, ter consciência de sua própria existência, entre outras. Essas habilidades demonstram a profunda capacidade que o cérebro humano possui de manipular símbolos e adquirir novos conhecimentos. Por outro lado, sabe-se que a inteligência de um indivíduo está diretamente associada ao conhecimento que o mesmo possui e que este conhecimento representa uma forte influência no processo de aprendizagem de novos conceitos e em habilidades específicas, como por exemplo, a capacidade de resolver problemas [MOREIRA1999]. São essas características do comportamento inteligente que os pesquisadores têm procurado entender e implementar em sistemas artificiais. Como exemplo pode-se citar os trabalhos de Allen Newell sobre teorias da cognição, sistemas de conhecimentos e sistemas de resolução geral e automática de problemas – o GPS (*General Problem Solver*) [NEWELL1963]. Newell se destacou pela sua teoria do conhecimento segundo a qual o conhecimento está ligado à racionalidade. Para Newell, a inteligência de um sistema pode ser definida pela capacidade que ele possui de usar todo o seu conhecimento na obtenção dos

seus objetivos. Esses conceitos tiveram grande influência nos primeiros trabalhos em inteligência artificial (IA) os quais tenderam para uma simulação direta dos procedimentos mentais humanos, tais como estes eram entendidos, e que tratavam essencialmente de questões de lógica. Com os avanços nas pesquisas, as tentativas de simulação direta dos procedimentos mentais foram sendo revistas, enquanto novos paradigmas, mais flexíveis e com características adaptativas, foram surgindo e ganhando espaço no cenário acadêmico, dentre os quais está a computação evolutiva (CE). A computação evolutiva compreende um grupo de métodos computacionais que incorporam mecanismos evolutivos presentes na natureza e descritos pela teoria da evolução natural. Como exemplo, tem-se os algoritmos genéticos (AGs) propostos inicialmente por John Holland em 1962 [HOLLAND1975] e suas variantes – os Sistemas Classificadores (SC) – e a Programação Genética (PG) [KOZA1992a], os quais têm se mostrado promissoras ferramentas para análise e solução de problemas nas mais diversas áreas do conhecimento humano.

Neste trabalho, propõe-se o desenvolvimento de um algoritmo evolutivo para identificação de procedimentos utilizados por humanos na solução de problemas categorizados como problemas de raciocínio dedutivo. Apresenta-se um algoritmo evolutivo denominado Classificador Genético (CG) que procura, através da recombinação de operadores e de medidas de semelhança com procedimentos apresentados por um jogador humano, construir sua própria seqüência de procedimentos para a solução do problema proposto. A partir da análise das simulações e da confrontação com os resultados obtidos pelo jogador humano, discutem-se os mecanismos envolvidos na resolução dessa classe de problemas, analisa-se o desempenho do algoritmo proposto e as possíveis semelhanças com o procedimento apresentado pelo jogador humano.

1.1 JUSTIFICATIVA PARA A ESCOLHA DO TEMA

A proposta nasceu de inúmeras discussões sobre como são construídos os procedimentos humanos para a solução de problemas e como tais procedimentos podem contribuir para o desenvolvimento de sistemas inteligentes. A temática abordada pelo trabalho é a autonomia cognitiva em sistemas artificiais, mais especificamente, a autonomia na solução de problemas de raciocínio dedutivo em

sistemas artificiais. Nesse contexto, propõe-se o uso de um jogo pertencente à classe de jogos de raciocínio dedutivo como plataforma de teste e validação para o algoritmo proposto. Tal escolha deve-se a algumas das características apresentadas por este jogo:

- Trata-se de um jogo que pode ser formulado genericamente como um problema de busca de um código secreto em um espaço de estados [KNUTH1977; STUCKMAN2005].
- A solução requer do jogador humano: boa memória para reter todas as informações recebidas durante a partida, raciocínio dedutivo para interpretar as informações sobre seus acertos e erros, e a criação de alguma estratégia como, por exemplo, estabelecer objetivos intermediários ou eliminar possibilidades que permitam restringir o espaço de busca e encontrar a solução do problema em um número aceitável de lances.
- A solução do jogo escolhido guarda semelhanças com outros jogos também categorizados como problemas de raciocínio dedutivo, como por exemplo, o Sudoku e o Campo Minado. Os problemas tratados nesses jogos surgem em aplicações práticas, como em projeto de redes de telecomunicação e de circuitos VLSI, escalonamento e roteamento de veículos, etc. Outras áreas de aplicação incluem economia (matrizes de entrada/saída), física (estados de energia mínima), estatística (análise de dados), a biologia molecular (alinhamento de DNA e proteínas, inferência de padrões). As conclusões obtidas para este trabalho poderão também ser aplicadas, embora não diretamente, na solução desses problemas [GAREY1979; GUTIN2002; KELLERER2005; MARTELLO1990; WHITLEY1989].

O jogo escolhido é portanto utilizado como problema motivador e plataforma de validação dos conceitos e objetivos investigados nesse trabalho. Espera-se que os resultados da pesquisa possam contribuir para o desenvolvimento de sistemas com maior flexibilidade na aquisição de conhecimento e autonomia na resolução de problemas.

1.2 OBJETIVOS

O projeto proposto contempla estudos nas seguintes áreas de pesquisa: aprendizagem humana, solução automática de problemas e computação evolutiva. Os objetivos gerais e as delimitações desse trabalho são descritos a seguir.

1.2.1 Objetivos Gerais

Neste trabalho investiga-se a utilização de modelos evolutivos na identificação de procedimentos para a solução de problemas de raciocínio dedutivo.

1.2.2 Objetivos Específicos

Desenvolvimento de um algoritmo evolutivo para identificação de procedimentos humanos para solução de um jogo de raciocínio dedutivo (*Mastermind*®). Para isso, utiliza-se uma abordagem que explora os diferentes níveis de complexidade do problema, partindo de uma proposta de solução puramente combinatória guiada por um algoritmo genético (AG) e independente do procedimento apresentado pelo humano, indo até um modelo baseado na observação dos procedimentos do humano que procura extrair um procedimento de solução que guarde semelhanças com o procedimento apresentado pelo humano.

1.3 ORGANIZAÇÃO DO TRABALHO

Este trabalho encontra-se estruturado em sete capítulos. O Capítulo 2 aborda as principais teorias da aprendizagem, suas idéias fundamentais e explora os processos cognitivos de resolução de problemas. O Capítulo 3 introduz o problema central, detalha o estudo de caso utilizado e explora os requisitos exigidos de um jogador humano na solução do problema proposto. No Capítulo 4 são apresentados os fundamentos da Computação Evolutiva, destacando-se os principais paradigmas e suas aplicações. O estudo deste domínio gerou a proposta do classificador genético descrita no Capítulo 5. O Capítulo 6 discute os resultados obtidos nos diferentes ensaios e simulações e o Capítulo 7 apresenta as conclusões e os trabalhos futuros.

CAPÍTULO 2

COGNIÇÃO E SOLUÇÃO DE PROBLEMAS

Uma das principais habilidades do cérebro humano é a sua capacidade de resolver problemas. A maioria das decisões a serem tomadas pelo cérebro envolve a resolução de algum tipo de problema. Alguns exemplos de problemas triviais encontrados diariamente são: decidir a melhor rota para o trabalho, encontrar uma vaga no estacionamento e priorizar tarefas para serem executadas em certo dia. A solução de problemas é um processo cerebral executado nas mais altas instâncias e consiste na busca por uma solução para um dado problema ou em um caminho para alcançar um determinado objetivo [WANG2004]. O estudo dos processos envolvidos na solução de problemas tem interessado pesquisadores das mais diferentes áreas que buscam um modelo formal e axiomático para esse mecanismo cerebral o qual poderá servir como base para várias teorias e sistemas automáticos para solução de problemas.

As pesquisas desenvolvidas na área de psicologia com foco no estudo das habilidades cognitivas para a solução de problemas tiveram seu início nos anos setenta e têm demonstrado os efeitos desses processos cognitivos nos mecanismos de aprendizagem. Além disso, pesquisadores têm reconhecido os benefícios do uso de simulações em computador na validação de teorias que muitas vezes não podem ser efetivamente provadas, mas podem ser demonstradas por simulação em computador [CHIEW2004]. Neste capítulo, apresenta-se um breve histórico do desenvolvimento da ciência cognitiva descrevendo as principais teorias da aprendizagem, suas idéias fundamentais e algumas abordagens cognitivas para a solução de problemas.

2.1 HISTÓRICO DO DESENVOLVIMENTO DA CIÊNCIA COGNITIVA

Os estudos sobre o desenvolvimento cognitivo e das capacidades cognitivas do homem têm origens na antiguidade, com os gregos como Aristóteles tratando de

questões relacionadas ao pensamento. Descartes, no século XVII, já mencionava a diferença entre alma e corpo, que para ele seriam entidades independentes que interagiam através da glândula pineal. Nesta época, séculos XVII e XVIII, os estudos da mente eram dominados por duas correntes de pensamento, os behavioristas (também conhecidos como comportamentalistas) que se concentravam no estudo do comportamento e, influenciados pelos resultados de Newton, procuravam adaptar modelos comportamentais a modelos da Física Clássica, e os estruturalistas que praticavam uma pré-psicanálise de introspecção a partir dos elementos básicos da consciência [CROOKS1988].

Durante o século XIX e parte do século XX os behavioristas ganharam força. Autores como Ivan Petrovich Pavlov (1849-1936), John Broadus Watson (1878-1958) [BAUM1999] e Burrhus Frederic Skinner (1904-1990) [SKINNER1953] procuraram explicar o comportamento humano através de modelos baseados em arco-reflexo. Segundo estes autores todo comportamento poderia ser entendido como resultado de fatores do meio ambiente que condicionavam o indivíduo a reagir de um determinado modo.

Os modelos behavioristas predominaram no cenário científico entre os anos de 1920 e 1940, quando então começaram a entrar em declínio. Nesse momento, modelos de representações mentais e conceitos como imaginação, desejos e crenças, até então descartados como não científicos, passaram a ser intensamente discutidos, surgindo novas teorias sobre o cérebro, seu funcionamento e sobre mecanismos de aprendizagem, inclusive estudos comparando o cérebro ao recém inventado computador.

A seguir, discute-se o conceito de aprendizagem e são apresentadas algumas contribuições importantes dadas ao tema.

2.2 TEORIAS DA APRENDIZAGEM

A aprendizagem pode ser definida como a aquisição de novas capacidades e comportamentos através da experiência. A maior parte do conhecimento que o ser humano possui sobre o mundo não nasce com ele, mas é adquirida através da experiência. Este conhecimento é armazenado na memória pelo cérebro ficando

disponível para futuros acessos. Um exemplo importante de modelo de aprendizagem é a chamada aprendizagem associativa, na qual o aprendiz é exposto a pares de eventos ou estímulos com a finalidade de fixar a relação entre os mesmos [ANDREAE1995]. O estudo da aprendizagem associativa tem proporcionado inúmeras descobertas sobre os princípios da aprendizagem, os quais são aplicáveis a espécies tão diferentes quanto os insetos e os humanos e a comportamentos tão diversos como a salvação e o medo.

A inteligência artificial também tem se beneficiado com avanços obtidos pelos estudos na área de aprendizagem, uma vez que um dos seus objetivos é implantar em sistemas artificiais a capacidade de aprendizado dos organismos inteligentes. Neste ponto, destacam-se a teoria da Epistemologia Genética de Jean Piaget e a teoria da Aprendizagem Significativa de David Ausubel, por se debruçarem sobre a aprendizagem cognitiva, dando ênfase ao estudo dos processos de aquisição de conhecimento [MOREIRA1999].

2.2.1 Epistemologia Genética de Piaget

Um dos grandes responsáveis pelo abandono do paradigma behaviorista foi o suíço Jean Piaget (1896-1980) que desenvolveu uma teoria, sobre o desenvolvimento cognitivo humano, denominada Epistemologia Genética. Jean Piaget foi um dos grandes estudiosos da aprendizagem humana, fornecendo importantes contribuições à psicologia e a epistemologia. Ele conseguiu em seu trabalho traduzir questões filosóficas em questões empíricas, estabelecendo padrões para a natureza e o desenvolvimento do conhecimento.

Piaget propôs que o desenvolvimento intelectual é uma continuidade da adaptação biológica realizada por outros meios. Para ele, a atividade intelectual é uma espécie de órgão, possuindo um lado funcional e um lado estrutural. Mas, diferente de outros órgãos que possuem estrutura e funcionalidades fixas, os órgãos cognitivos apresentam funcionalidade contínua com estruturas descontínuas. A funcionalidade contínua estaria ligada ao crescimento do conhecimento durante a evolução do indivíduo e as estruturas descontínuas seriam as diferentes formas de aquisição de conhecimento durante o seu crescimento, como as influências culturais. A teoria cognitiva de Piaget não admite que a desenvolvimento intelectual seja

inerente à própria vida, mas assume que o desenvolvimento intelectual é uma forma de adaptação criada pela vida durante o processo evolutivo. A adaptação é um processo de transformação desencadeado pelo organismo visando o equilíbrio e, por conseguinte, a sua conservação no meio em que vive. Desta forma, a construção do intelecto se dá na medida em que novos patamares de equilíbrio adaptativo são alcançados [PIAGET1982]. As idéias contidas na teoria de Piaget têm influenciado inúmeros trabalhos que buscam a modelagem de sistemas artificiais baseados em hipóteses da aprendizagem humana [CONEN1995; DRESCHER1991; PEROTTO2004; WAZLAWICK1997].

2.2.2 Aprendizagem Significativa de Ausubel

Ausubel em sua teoria denominada Aprendizagem Significativa, descreve o que ele chama de Estrutura Cognitiva. A estrutura cognitiva é uma estrutura hierárquica de conceitos que são representações das experiências do indivíduo. Segundo Ausubel a aprendizagem é a organização e integração de toda a informação armazenada por um indivíduo nesse sistema mental ordenado. O conteúdo previamente adquirido pelo indivíduo representa um forte influenciador do processo de aprendizagem. Novas informações são assimiladas e armazenadas na razão direta da qualidade da estrutura cognitiva prévia do aprendiz. Esse conhecimento anterior resultará em pontos de ancoragem nos quais as novas informações irão encontrar um modo de se integrar àquilo que o indivíduo já conhece. Apesar de a estrutura prévia orientar o modo de aquisição de novas informações, estas também influenciam o conhecimento já armazenado, alterando quando necessário a estrutura cognitiva, resultando num processo evolutivo do conhecimento presente na estrutura cognitiva. Esse processo de associação de informações interrelacionadas denomina-se Aprendizagem Significativa. Assim como a teoria de Piaget, os estudos de Ausubel também têm influenciado muitos trabalhos na área de inteligência artificial, os quais procuram modelar a inteligência a partir de analogias com a teoria de Ausubel [MAGUITMAN2004; MUNTJEWERFF1999].

2.3 APRENDIZAGEM EM SISTEMAS DE INTELIGÊNCIA ARTIFICIAL

A Aprendizagem de Máquina constitui uma sub-área da Inteligência Artificial e tem como principal objetivo o estudo e a compreensão dos processos de aprendizagem e o desenvolvimento de algoritmos que suportem estes processos para aplicação em sistemas computacionais. Entende-se por aprendizagem a capacidade do sistema de utilizar suas percepções do mundo não só para agir imediatamente, mas também para melhorar suas ações no futuro. De forma geral, os algoritmos de aprendizagem de máquina devem proporcionar mudanças no conteúdo, na estrutura de organização e na manipulação do conhecimento dos sistemas melhorando o seu desempenho [REZENDE2003; RUSSEL1995]. Como melhoria no desempenho do sistema entende-se a apresentação de um comportamento mais eficiente, com redução no tempo de execução de tarefas, maior confiabilidade nas relações entre sistemas, aquisição de novas habilidades e alcance satisfatório de objetivos [DIETTERICH2003; JONES2000; POSNER1993; ROISENBERG1998; SANDOZ1993; WILSON1999]. Entre os diversos paradigmas da Aprendizagem de Máquina destacam-se os algoritmos evolutivos discutidos no capítulo 4.

2.4 ABORDAGENS COGNITIVAS PARA SOLUÇÃO DE PROBLEMAS

O mecanismo mental utilizado na solução de problemas pode ser definido como resultado da aplicação de diferentes processos cognitivos, tais como: capacidade de memorização, aquisição de conhecimento e capacidade de adaptação a novas situações, entre outros. A aquisição de conhecimento é um processo cognitivo descrito pela teoria de Ausubel (seção 2.22) e classificado como um processo de transferência [MUNTJEWERFF1999], uma vez que, o conteúdo aprendido por um indivíduo em uma determinada situação irá afetar o processo de aprendizagem e execução em uma situação futura [MAGUITMAN2004]. Desta forma, a solução de problemas pode ser considerada como um tipo de processo de transferência, uma vez que envolve a utilização de conhecimento adquirido previamente em uma nova situação [COX2005].

Um problema consiste basicamente de três componentes: dados, objetivos e operadores. Dados são informações a respeito do problema. Objetivos são definidos como um estado ou meta a ser alcançado para solução do problema. Operadores são ações que podem ser executadas para alcançar o objetivo do problema. Além disso, para cada problema existe um espaço de estados associado, que corresponde a todas as possibilidades de caminhos e soluções possíveis para alcançar os objetivos desejados. A definição do espaço de estados é por si só um problema – fatores como: especificação insuficiente do problema, objetivos ambíguos ou mesmo a inexistência de uma sequência de ações que resolva o problema (algoritmo), pode levar a definição de um espaço de estado que não contemple a solução do problema levando a não-solução do mesmo.

Uma vez definidos os termos do problema e seu espaço de estados, o próximo passo é explorar as possíveis abordagens para o problema. Psicólogos têm proposto várias abordagens cognitivas para a solução de problemas:

- **Fatos** – uso de solução direta sem esforço de busca. É uma solução simbólica, na qual o indivíduo analisa os fatos (verdades incondicionais) e conclui uma solução.
- **Subida da Encosta** – em alusão ao processo de subida de uma encosta, em que o alpinista procura sempre fazer um movimento que o aproxime do seu objetivo que é o topo. Este processo consiste em fazer a cada interação algum movimento que aproxime a solução proposta do objetivo do problema.
- **Encadeamento para trás** – partindo-se da solução procura-se o caminho que levou até ela.
- **Busca Exaustiva** – busca-se sistematicamente em todo o espaço de estado até encontrar a solução do problema.
- **Solução por Analogia** – buscam-se analogias com outros problemas já resolvidos para encontrar a solução do problema corrente.

Existem também as abordagens heurísticas, que são soluções que englobam conhecimento específico a respeito do problema com o objetivo de restringir o

espaço de busca ou selecionar estratégias com maior probabilidade de êxito. Dentre estas seguem alguns exemplos:

- **Regra Prática** – busca seletiva de uma região do espaço de busca, escolhida a partir da experiência.
- **Solução por Partes** – consiste em dividir o problema principal em sub-problemas sempre que possível.
- **Brain Storming** – uma abordagem heurística que consiste em apresentar inicialmente, sem nenhum tipo de restrição, possíveis soluções para o problema, numa verdadeira tempestade de idéias para depois em sucessivos refinamentos chegar a uma solução ótima ou aceitável.

A utilização dessas abordagens não garante que o objetivo do problema possa ser alcançado, especialmente se a solução não estiver dentro do espaço de busca. Do ponto de vista do processamento de informação, existem outros fatores importantes que podem impedir que a solução seja encontrada. Entre eles estão os relacionados à capacidade de memória do indivíduo, como por exemplo:

- **Quantidade de Memória de Trabalho** – A memória de trabalho está ligada à capacidade de reter informação e detalhes sobre o problema para processamento imediato. Muitas vezes, a solução do problema requer que um grande número de informações, detalhes e resultados de pré-processamentos sejam combinados para se inferir a solução. A memória de trabalho é, portanto, um limitador da capacidade de processamento do ser humano. Entretanto, algumas habilidades cognitivas envolvidas na resolução de problemas podem ser aprendidas bem o suficiente para se tornarem automáticas, requerendo o mínimo de utilização de memória. A memória de trabalho também pode ser ampliada através do uso de mecanismos externos, como por exemplo, lápis e papel.
- **Memória de Longa Duração** - A solução de problemas complexos requer, na sua maioria, a recuperação de informações gravadas na memória de longa duração. As informações gravadas nessa área de memória dizem respeito às lembranças e conceitos aprendidos anteriormente pelo indivíduo. Muitas vezes, essas informações podem ser codificadas deficientemente ou

mesmo sofrer alterações ao longo do tempo, o que causa perda de detalhes importantes a respeito do problema a ser solucionado.

Outras diferenças individuais, como por exemplo, a presença ou falta de habilidades específicas e experiência adquirida em situações passadas, podem influenciar na capacidade de indivíduos solucionarem determinados tipos de problemas. De modo geral, as características presentes nos indivíduos considerados especialistas na solução de problemas são:

- **Expertise** – excepcional desempenho no tratamento de questões de determinado domínio do conhecimento.
- **Base de Conhecimento** – informações, descrições, hipóteses, conceitos, teorias, princípios e procedimentos que são ou úteis ou verdadeiros e acumulados através da experiência.
- **Memória** – capacidade cerebral superior relacionada ao processo de retenção de dados, informações e conhecimentos, obtidos em experiências vividas.
- **Categorização** – capacidade de classificar objetos atribuindo-os a certas classes por prototipicidade ou semelhança com elementos daquela classe.
- **Representação** – capacidade de extrair e reconhecer as qualidades sensíveis dos objetos.
- **Biblioteca de Estratégias** – conhecimento de diferentes abordagens de solução de problemas e habilidade para selecionar a melhor opção para um dado problema.
- **Velocidade e Precisão** – rapidez e precisão no planejamento e solução de problemas adquiridos pela prática.
- **Metacognição** – conhecimento dos processos cognitivos utilizados, o que permite ao especialista o monitoramento, julgamento e detecção de erros mais rapidamente [COX2005].

2.5 PROCEDIMENTOS PARA SOLUÇÃO DE PROBLEMAS

Pode-se definir os procedimentos para solução de problemas como processos de introspecção e observação informal. Diferente de uma máquina, as habilidades de um indivíduo para solução de problemas podem mudar ao longo de sua vida. Uma pessoa mais velha pode ter uma base de conhecimento mais ampla permitindo um desempenho melhor na solução de problemas. Entretanto, a capacidade para resolver problemas pode ser afetada por aspectos biológicos e psicológicos, como por exemplo, um dano cerebral ou pressão social. De forma geral, os procedimentos para solução de problemas podem ser associados às seguintes etapas:

- **Fase de Preparação ou Entendimento do Problema** – identificação do problema, seus dados e possíveis representações.
- **Fase de Planejamento** – determinação da seqüência de procedimentos para solução do problema.
- **Fase de Execução** – execução das ações planejadas.
- **Fase de Avaliação dos Resultados** – avaliação os resultados da abordagem executada com o objetivo de aprender sobre como problemas similares podem ser resolvidos em situações futuras.

São esses procedimentos gerais para solução de problemas que os pesquisadores têm procurado decifrar e implementar em sistemas artificiais, utilizando-se para isso, as mais diversas abordagens computacionais. Nos próximos capítulos, será apresentado com detalhes um problema de raciocínio dedutivo e a proposta de uma série de algoritmos evolutivos que procuram identificar procedimentos para solução do problema.

CAPÍTULO 3

JOGOS DE RACIOCÍNIO

Os jogos sempre estiveram presentes nas pesquisas de inteligência artificial. Nos anos 50 Claude Shannon e Alan Turing escreveram alguns artigos propondo a criação de computadores que poderiam desafiar humanos em jogos de tabuleiro [SHANNON1950]. Mais tarde, ainda nos anos 50, Alex Bernstein da IBM e Allen Newell, Cliff Shaw e Herbert Simon do Carnegie Institute of Technology desenvolveram um programa capaz de jogar xadrez [NEWELL1955; NEWELL 1958; NEWELL1959] e Arthur Samuel da IBM um programa capaz de jogar damas [SAMUEL1967]. Nos anos 60, Newell, Shaw e Simon utilizaram vários jogos de raciocínio para o desenvolvimento e validação do *General Problem Solver - GPS* [NEWELL1963]. Os jogos têm sido considerados como valiosos testes para pesquisas em inteligência artificial, uma vez que exigem do jogador a execução de tarefas cognitivas para a sua solução e por serem de difícil solução, mas suficientemente circunstanciados para permitirem testes de fácil entendimento que permitam explorar questões fundamentais em inteligência artificial. Desta forma, muitas das idéias desenvolvidas para a solução de jogos têm encontrado importantes aplicações em outras áreas da inteligência artificial. Este capítulo descreve o jogo *Mastermind* utilizado como estudo de caso e plataforma para validação de uma série de variantes de um algoritmo evolutivo para identificação de procedimentos de raciocínio humano.

3.1 MASTERMIND

Mastermind é a versão comercial de um velho jogo, pertencente à categoria dos jogos de raciocínio dedutivo, chamado de “*Bulls and Cows*” [BOGOMOLNY1999], em que dois jogadores, o codificador e o decodificador avaliam suas habilidades cognitivas. O codificador elabora uma combinação secreta de pinos coloridos a qual fica escondida do decodificador. O

decodificador tem como objetivo descobrir o código secreto no menor número de lances possível. Para isso, o decodificador inicia o jogo propondo uma sequência de pinos coloridos para o código desconhecido. Em seguida, o codificador compara a tentativa com o seu código e responde com um marcador preto, para cada pino da solução proposta que possua a cor e a posição correta, e com um marcador branco, para cada pino que possua a cor correta, mas que esteja na posição errada. A figura 3.1 mostra a versão clássica do jogo utilizando uma combinação de 4 pinos extraído de um alfabeto de 6 cores (verde (1), amarela (2), laranja (3), turquesa (4), vermelha (5), azul (6)), a linha superior mostra um código secreto formado por 4 pinos com cores (verde (1), turquesa (4), azul (6) e azul (6)). A segunda linha apresenta uma tentativa do decodificador de descobrir o código secreto e ao seu lado a resposta do codificador representada pelos marcadores pretos e brancos. O *Mastermind* pode ser jogado também na versão “Super” cujo código secreto possui 5 pinos extraído de um alfabeto de 8 cores.

Figure 3.1 – Detalhes de um lance para versão clássica do *Mastermind*.

Ao decodificador é permitido fazer um número máximo de lances na tentativa de acerto, após as quais o codificador é declarado vencedor. Caso o decodificador encontre a combinação, ele acumulará um número de pontos igual ao número de lances efetuados. Em um torneio, será declarado vencedor o jogador que acumular menos pontos. A versão original do jogo “*Bulls and Cows*” era na verdade um pouco diferente – era jogado utilizando-se uma folha de papel e o código era formado obrigatoriamente por dígitos diferentes no intervalo de 0 a 9. A tabela 3.1 apresenta um resumo dos principais termos utilizados na descrição do jogo e que auxiliam no entendimento deste trabalho.

Tabela 3.1 – Termos utilizados na descrição do jogo *Mastermind*.

Termo	Descrição
Codificador	Jogador que elabora o código secreto a partir de um alfabeto de cores.
Decodificador	Jogador que tenta decifrar o código secreto, isto é, quais são as cores e as respectivas posições no código secreto.
Partida	Uma partida começa com a elaboração do código secreto pelo codificador e termina quando o decodificador decifra o código ou atinge o número máximo de lances sem decifrá-lo.
Lance	Um conjunto de cores apresentado pelo decodificador, referindo-se às diversas etapas de uma partida.
Torneio	Conjunto de partidas onde os jogadores se revezam no papel de codificador e decodificar. Em um torneio cada decodificador acumula em cada partida o número de lances utilizado para decifrar o código. No final do torneio é considerado campeão o jogador que acumulou o menor número de lances.
Pinos brancos	Marcador utilizado pelo codificador para indicar o número de cores corretas, mas em posições erradas, no lance efetuado pelo decodificador.
Pinos pretos	Marcador utilizado pelo codificador para indicar quantas cores estão na posição correta no lance efetuado pelo decodificador.

3.2 MASTERMIND: O PROBLEMA

De forma geral, o problema do *Mastermind* pode ser caracterizado como um problema de busca em um espaço de estados [CHEN1996; CHVATAL1983]. O código secreto é extraído de um alfabeto de cores com cardinalidade N , sendo que o código secreto e os lances apresentados pelo jogador (decodificador) têm comprimento N . Assim, o espaço de busca é dado por L^N o que, para a versão clássica do jogo $L=6$ e $N=4$, corresponde a um espaço de busca com 1296 estados possíveis e para a versão super com $L=8$ e $N=5$, a um espaço de busca com 32768 estados possíveis. Cada avaliação fornecida pelo codificador na forma de um marcador preto ou branco restringe o espaço de busca eliminando algumas

combinações possíveis. Pode-se também, formular o *Mastermind* como um problema de otimização combinatória, no qual cada lance possui uma pontuação definida através dos marcadores pretos e brancos e nesse caso a pontuação deve ser maximizada. Cabe ressaltar que com o uso de métodos adequados de avaliação da evolução do cenário do jogo (seqüência de lances) é possível resolver uma partida com um número de lances muito inferior ao número total de estados (possíveis combinações) existentes. A questão é portanto a de se identificar (ou descobrir por aprendizagem) quais são tais métodos.

3.3 CARACTERÍSTICAS COGNITIVAS DO JOGADOR

Embora de regras simples, algumas características do *Mastermind* fazem dele um jogo de difícil solução. Primeiro, porque existem muitas combinações possíveis, o que torna muito difícil a tarefa de encontrar a solução através de tentativas aleatórias. Isto significa que se o jogador testar uma jogada diferente a cada três segundos ele gastará mais de uma hora para esgotar todas as possibilidades na versão clássica e quase vinte e oito horas para a versão super. Um jogador experiente deve ser capaz de extrair o máximo de informação de cada avaliação e então combiná-las de forma a obter o máximo de conhecimento a respeito do código secreto. Uma criança, por exemplo, possui dificuldade em fixar mentalmente todas as informações recebidas e analisá-las conjuntamente para extrair as informações sobre o código secreto e provavelmente irá executar lances aleatórios até descobrir o código secreto. Já um jogador intermediário irá utilizar alguns operadores básicos para fazer os seus lances. Sua estratégia será melhor do que a da criança, mas ainda limitada.

Portanto, um bom jogador de *Mastermind* precisa possuir boa capacidade de memória para armazenar as informações de todas as avaliações, concentração para manter o foco nas informações relevantes e raciocínio para inferir a próxima jogada. Observa-se que tais características cognitivas não são exclusivas de bons jogadores de *Mastermind*, mas características gerais de especialistas em solução de problemas que exijam capacidade de raciocínio, conforme descrito na seção 3.2. Isto sugere a possibilidade de utilizar o *Mastermind*, em conjunto com outros instrumentos e técnicas, na avaliação da capacidade cognitiva de indivíduos, como por exemplo, na

seleção preliminar de candidatos a uma vaga de programador de computadores cujas características cognitivas gerais são essencialmente as mesmas. A figura 3.2 apresenta três diferentes níveis de jogadores e suas respectivas estratégias para o *Mastermind*.

Figura 3.2 - Relação entre jogadores e estratégias.

3.4 ALGORITMOS E DESEMPENHOS

Após o primeiro artigo publicado em 1977 por Donald Knuth [KNUTH1977], inúmeros trabalhos, com as mais diferentes abordagens, têm sido propostos para a solução do *Mastermind*. De maneira geral, as estratégias de solução algorítmicas para o problema podem ser categorizadas da seguinte forma:

Estratégia Ótima - Neste caso, lances que são sabidamente incorretos são jogados, com o objetivo de extrair informação sobre o código escondido e reduzir o espaço de busca. Por exemplo, no primeiro lance o jogador apresenta todas as cores iguais, suponha que ele receba um marcador preto como resposta (uma cor correta na posição correta). No segundo lance, o jogador mantém apenas um pino com a cor anterior e substituem todas as outras por uma nova cor, e assim sucessivamente até que todas as cores tenham sido testadas. Após esse ciclo de jogadas a solução do problema pode ser concluída analiticamente (figura 3.2).

Figura 3.3 – Exemplo de partida em que o jogador utiliza uma estratégia denominada Ótima.

Lance Ótimo - nessa estratégia cada lance é ótimo, no sentido que ele possui a maior probabilidade de ser o código secreto. Para isso, cada lance deve ser consistente com todos os lances feitos anteriormente. Por exemplo, suponha que o primeiro lance não apresente nenhuma cor correta (nenhum marcador branco e nenhum marcador preto). O próximo lance, para ser coerente com o primeiro, não deve possuir nenhuma das cores presentes no primeiro e assim sucessivamente. Este procedimento pode ser realizado da seguinte forma: enumeram-se todas as combinações possíveis, escolhe-se uma combinação para iniciar o jogo (essa combinação pode ser uma escolhida aleatória ou guiada por alguma heurística pré-definida). Em seguida, eliminam-se todas as combinações inconsistentes com a jogada anterior. Seleciona-se uma nova combinação e repete-se o processo até que o espaço de busca se reduza a uma única opção. Embora simples, já foi demonstrado que esta estratégia utiliza não mais que 6 lances para encontrar o código secreto [KNUTH1977] na versão 6x8. Outros trabalhos apresentam algoritmos melhorados que podem solucionar o *Mastermind* em 4,3 lances em média [KOYAMA1993; MITCHELL1999a].

Além das estratégias acima, diversos trabalhos têm explorado a utilização de algoritmos genéticos na solução do *Mastermind* obtendo resultados que, em média, são compatíveis com os valores obtidos por outras estratégias de busca heurística [AGOSTINHO2003; BERNIER1996; KABATIANSKI2000; MERELO1999; ROCHE1997; ROSA1999; VOMLEL2004]. Outras estratégias de inteligência artificial como as relacionadas acima foram propostas, mas elas foram criadas por mentes que conhecendo este jogo souberam descrever os meios para resolvê-lo, enquanto que a proposta deste trabalho é procurar obter os procedimentos para a solução do problema de forma emergente, ou seja, sem a intervenção direta de um programador que seja *expert* neste jogo.

No próximo capítulo, apresentam-se as bases da computação evolutiva e discorre-se sobre seus principais paradigmas.

CAPÍTULO 4

COMPUTAÇÃO EVOLUTIVA

Os principais paradigmas da inteligência artificial se desenvolveram baseados em uma variedade de modelos da natureza nos quais existem vários processos que podem ser denominados de “inteligentes”. Um ótimo exemplo são as Redes Neurais Artificiais (RNAs) cujas idéias principais inspiram-se no funcionamento dos neurônios do cérebro humano. Assim como as redes neurais artificiais a computação evolutiva utiliza-se de conceitos biológicos, como evolução e reprodução, para o desenvolvimento de sistemas cujo principal objetivo é a solução de problemas. O conceito principal por trás da computação evolutiva está na representação de estruturas de dados ou programas como populações de elementos que interagem entre si através de mecanismos denominados de operadores, sendo os genéticos um tipo particular de operadores. A aplicação de tais operadores é guiada por uma medida de qualidade dos elementos da população com relação aos objetivos do programa. As técnicas de computação evolutiva têm sido utilizadas nas mais variadas áreas de pesquisa em inteligência artificial, dentre as quais podemos citar os estudos que envolvem aprendizagem de máquina e resolução automática de problemas [BAUM1995; CASTILL02003]. Segundo Wolfgang Banzhaf [BANZHAF2000], embora as técnicas de computação evolutiva tenham evoluído rapidamente nas últimas décadas, elas ainda não estão suficientemente maduras e continuarão sendo um potencial campo de pesquisa para os próximos anos.

Nas seções seguintes, apresentam-se as bases da computação evolutiva e suas principais técnicas.

4.1 BASE BIOLÓGICA

A teoria da evolução publicada por Charles Darwin em 1859 propôs que a seleção natural é o mecanismo que distingue os sistemas biológicos dos demais sistemas físicos e químicos. Para Darwin a seleção natural conduz os indivíduos de

uma espécie, através de alterações sucessivas, a uma melhor adaptação ao meio em que vivem. Darwin descreveu em sua teoria um processo de seleção natural associado à diversidade das espécies para explicar o processo evolutivo. A seguir, as principais hipóteses presentes na teoria de Darwin:

1. Os filhos tendem a ser em maior número que os pais;
2. O número de indivíduos de uma espécie permanece aproximadamente constante;
3. Das hipóteses 1 e 2, pode-se concluir que haverá disputa pela sobrevivência;
4. Em uma mesma espécie, os indivíduos apresentam pequenas diferenças que, na maioria das vezes, também são encontradas nos pais.

Portanto, os indivíduos cujas variações se adaptam melhor às condições do ambiente em que vivem terão maior probabilidade de sobreviver e se reproduzir, sendo a evolução uma consequência direta da competição entre sistemas de reprodução.

As hipóteses anteriores explicam apenas os processos adaptativos em nichos ecológicos. Para uma teoria geral da origem das espécies são necessárias mais três hipóteses provenientes dos estudos da genética, a saber:

5. Algum processo de variação continuada deve ser responsável pela introdução de novas informações junto à carga genética dos organismos;
6. Não há limite para a sucessão de variações que podem ocorrer;
7. A seleção natural é o mecanismo para preservação das novas informações que correspondam a uma maior adaptação.

Das hipóteses anteriores, pode-se concluir que a seleção natural é probabilística, e seu objetivo primário é o indivíduo, mas seu efeito resultante manifesta-se na espécie como um todo, sendo ela beneficiária final do processo evolutivo [MAYR1988].

Os sistemas baseados em computação evolutiva são inspirados em tais hipóteses e utilizam uma terminologia baseada em analogias com entidades

biológicas reais, sendo que os modelos artificiais correspondem invariavelmente a estruturas simplificadas dos seus equivalentes biológicos.

4.2 PRINCÍPIOS DE FUNCIONAMENTO

O princípio de funcionamento dos sistemas de computação evolutiva baseia-se na interpretação das teorias de Darwin como um processo adaptativo de otimização. Dado um problema específico, os sistemas de computação evolutiva tentam encontrar uma solução a partir da criação e manipulação de uma população de estruturas candidatas à solução do problema. Cada indivíduo da população é submetido a processos de seleção baseados em medidas de adaptabilidade e a operadores genéticos (*crossover* e *mutação*) durante vários ciclos denominados gerações. Juntos, esses mecanismos visam criar uma população de indivíduos de alta adaptabilidade, ou seja, indivíduos que tenham alta probabilidade de ser a solução procurada do problema.

A estrutura básica de um algoritmo evolutivo é mostrada na tabela 4.1 [MICHALEWICZ1996]:

Tabela 4.1 – Estrutura básica de um Algoritmo Evolutivo.

```

procedimento algoritmo evolutivo

início
  Criar um conjunto inicial de estruturas;
  Avaliar a adaptabilidade de cada estrutura;
  Enquanto (condição de parada não for atingida) faça
    início
      Selecionar pares de estruturas;
      Submeter os pares aos operadores genéticos;
      Avaliar a adaptabilidade de cada estrutura;
    fim
  fim

```

O algoritmo evolutivo inicia-se com a criação de uma população de estruturas $P(t) = \{a'_1, a'_2, a'_3, a'_4, \dots, a'_n\}$, em que cada estrutura representa um candidato à solução do problema proposto. Cada solução a'_i é avaliada e produz alguma medida de adaptação ou *fitness*. Então, através desta medida os indivíduos mais adaptados

são selecionados e submetidos aos operadores genéticos dando origem a uma nova população $t+1$. O processo se repete até que a condição de parada seja atingida, a qual normalmente indica o aparecimento de um indivíduo com um valor de *fitness* que represente uma solução aceitável para o problema, ou que o número pré-definido de gerações foi atingido.

Os paradigmas evolutivos discutidos a seguir diferem entre si em alguns aspectos estruturais, como: operadores genéticos, estruturas de dados utilizadas para codificar cada indivíduo, métodos para criar a população inicial e métodos de seleção dos indivíduos para aplicação dos operadores genéticos. Entretanto, todos compartilham o mesmo princípio: uma população de estruturas ou indivíduos que evolui através de transformações sofridas pela aplicação de operadores genéticos e pela seleção dos mais aptos.

4.3 PRINCIPAIS PARADIGMAS EVOLUTIVOS

A partir da década de sessenta, surgiram diversos paradigmas computacionais inspirados em modelos simplificados da teoria da evolução. Entre eles cabe citar os Algoritmos Genéticos (AGs) propostos por John Holland entre os anos de 1960 e 1970 e que influenciaram outros dois paradigmas conhecidos como: Sistemas Classificadores (SCs) [BELEW1988] e Programação Genética (PG) [KOZA1992b]. Ainda nos anos setenta, mas de forma independente dos AGs, foram desenvolvidos os modelos denominados Estratégias Evolutivas e Programação Evolutiva [TANOMARU1995] que são descritos nas secções seguintes.

4.4 ALGORITMOS GENÉTICOS

Os algoritmos genéticos foram primeiramente propostos por John Holland nos anos 60 e desenvolvidos por ele e seus estudantes durante os anos 60 e 70 na Universidade de Michigan [HOLLAND1975]. Holland apresentou os AGs como uma abstração de modelos de evolução presentes na natureza. Assim, os AGs são técnicas não-determinísticas de busca e otimização, que operam sobre uma população de indivíduos (estruturas de dados que representam candidatos a solução

de um problema) através da aplicação de mecanismos de seleção, cruzamento (*crossover*) e mutação gerando novos indivíduos que, a cada geração, tornam-se mais aptos e portanto, mais próximos da solução ótima para o problema. Os AGs são métodos robustos e eficientes de busca em espaços de procura irregulares, multidimensionais e complexos. Um algoritmo genético em sua forma mais simples trabalha segundo o procedimento descrito na figura 4.1. Observe que este procedimento é executado um número n de vezes até que uma das condições de parada seja atingida. A população gerada após cada interação é chamada de geração. Após um número grande de gerações obtém-se um ou mais indivíduos com alto grau de adaptabilidade sendo esses os melhores candidatos à solução do problema proposto. A seguir, discute-se algumas características gerais e dos métodos de busca e otimização e apresentam-se os princípios que norteiam a implantação de um algoritmo genético.

Figura 4.1 – Diagrama de fluxo de um algoritmo genético simples.

4.4.1 Métodos de Busca e Otimização

Muitos dos problemas científicos podem ser formulados como problemas de busca e otimização. Os problemas de otimização caracterizam-se por possuírem um conjunto de restrições e uma função objetivo, ambos associados a variáveis de decisão. Eles podem ser problemas de maximização ou minimização e a solução

consiste em encontrar a combinação de valores para as variáveis que proporcione o melhor desempenho para o sistema. O conjunto de todas as combinações possíveis para as variáveis é denominado de espaço de busca ou espaço de estados. A combinação procurada será o conjunto de valores que maximiza ou minimiza a função objetivo sem violar as restrições do sistema. Os métodos de otimização de problemas podem ser classificados basicamente em três categorias: exaustivos, numéricos e probabilísticos.

Os métodos exaustivos executam uma busca sistemática em todos os pontos do espaço de busca até encontrar a solução do problema. Embora simples este método possui um custo computacional proibitivo quando o problema possui um grande espaço de busca.

Já os métodos numéricos são divididos em analíticos e baseados em cálculo numérico. Os analíticos são aplicados a problemas cuja função a ser otimizada é explicitamente conhecida e derivável ou pode ser aproximada por uma função derivável qualquer. O procedimento para encontrar o máximo ou mínimo da função é simples, basta encontrar os valores para os quais a derivada da função é nula dentro do espaço de busca desejado. Os métodos baseados em cálculo numéricos são capazes de encontrar somente pontos ótimos locais, não sendo eficientes para otimização de funções multimodais e não-diferenciáveis.

Os métodos probabilísticos procuram guiar a busca para regiões do espaço de busca onde é maior a probabilidade de existir uma solução para o problema. Os algoritmos genéticos discutidos na seção 4.5 são um exemplo de busca probabilística, sendo inspirados nos mecanismos de evolução presentes na natureza. Os algoritmos genéticos utilizam conceitos como adaptação, seleção natural e recombinação para selecionar conjunto de pontos com maior probabilidade de ser uma solução do problema e caracterizam-se por operarem em uma população de pontos, não necessitarem de derivadas e realizarem transições probabilísticas e não regras determinísticas [GOLDBERG1994].

4.4.2 AG – Implementação de Algoritmos Genéticos

Existem inúmeras implementações possíveis para um algoritmo genético e, em geral, elas obedecem à estrutura básica de funcionamento de qualquer algoritmo evolutivo (tabela 4.1) e diferem entre si nos detalhes estruturais. A seguir são descritos os principais aspectos da estrutura de um algoritmo genético.

4.4.2.1 Codificação dos Indivíduos

Um aspecto importante da estrutura de um algoritmo genético é a representação de cada um dos candidatos à solução do problema. Em um algoritmo genético, cada possível solução x para o problema deve ser expressa com uma seqüência de símbolos s extraídos de um alfabeto finito A . Cada seqüência s é chamada de indivíduo ou cromossomo, numa alusão aos sistemas biológicos em que os indivíduos são formados por um conjunto de cromossomos, e cada elemento de s é denominado de gene. Existem várias formas de representação dos indivíduos em um algoritmo genético, sendo duas delas mais comumente usadas: a representação binária, em que cada indivíduo é representado por uma cadeia binária de tamanho fixo de modo que cada cadeia binária represente um conjunto de valores para os parâmetros do problema e a representação através de uma cadeia de números inteiros, ou mesmo reais. A representação mais adequada irá depender do elemento que está sendo representado através do código genético. As figuras 4.2a e 4.2b representam o mesmo conjunto de quatro valores possíveis para as variáveis de um problema hipotético. Este conjunto de valores é um candidato à solução extraído do espaço de busca do problema e representa um indivíduo na população do algoritmo genético. Os genes do indivíduo são representados utilizando uma cadeia de inteiro (figura 4.2a) e com uma cadeia binária (figura 4.2b).

	var 1	var 2	var 3	var 4
a	4	6	3	5
b	1 0 0	1 1 0	0 1 1	1 0 1

Figura 4.2 – Representação inteira (a) e binária (b) de um indivíduo.

4.4.2.2 População Inicial

O primeiro passo na execução de um algoritmo genético é a geração de uma população inicial. Essa população é geralmente composta de um número N de indivíduos (candidatos à solução) selecionados aleatoriamente no espaço de busca do problema. É de vital importância para o bom desempenho do algoritmo genético que a população inicial possua a maior diversidade possível, cobrindo a maior área possível do espaço de busca. Isto porque, como no caso biológico, não há evolução sem variedade. Em outras palavras, se os indivíduos selecionados para a população inicial estiverem concentrados em uma região do espaço de busca, será mais difícil o algoritmo genético convergir para uma solução que se encontre fora dessa região.

4.4.2.3 Cálculo da Função de Avaliação (*fitness*)

A função objetivo ou função *fitness* mede a qualidade de um indivíduo da população como solução de um problema. Em um sistema biológico, a função objetivo é determinada pelo grau de adaptação do indivíduo ao meio ambiente – quanto mais adaptado for um indivíduo mais oportunidades de sobreviver e se reproduzir ele terá. Em uma população artificial, a avaliação de cada indivíduo, dada pelo valor obtido com a função objetivo, permite decidir quais indivíduos serão eliminados na próxima geração e quais sobreviverão passando parte do seu material genético para novos indivíduos [GOLDBERG1989].

4.4.2.4 Métodos de Seleção

Os mecanismos de seleção presentes nos AGs têm como função simular os processos biológicos de reprodução e seleção natural. A cada geração um grupo de indivíduos deve ser selecionado para participar do processo de reprodução que dará origem a uma nova geração. O algoritmo genético clássico utiliza um método de seleção de indivíduos denominado de Roleta (*roulette wheel*). Nesse método, atribui-se a cada indivíduo uma probabilidade proporcional ao *fitness* relativo de cada indivíduo na população.

$$p_{selec} = \frac{f(s)}{\sum_{i=1}^N f(s_i)} \quad (4.1)$$

sendo:

p_{selec} = probabilidade do indivíduo s de ser selecionado

$f(s)$ = o *fitness* do indivíduo s , dado pela função objetivo $f()$ aplicada ao indivíduo s

$\sum_{i=1}^N f(s_i)$ = o somatório do *fitness* de todos os indivíduos da população

Gera-se então, uma população temporária de N indivíduos selecionados a partir de um sorteio em que cada indivíduo tem probabilidade p dada pela equação 4.1 de ser escolhido. A figura 4.3 ilustra o processo, em que indivíduos com alto valor de *fitness* relativo possuem maior probabilidade de serem selecionados, podendo inclusive ser selecionados mais de uma vez, já indivíduos com baixo *fitness* têm baixa probabilidade de serem selecionados, podendo até ocorrer a sua extinção [MITCHELL1999b].

Figura 4.3 – Processo de seleção pelo método da roleta para uma população de 4 indivíduos.

4.4.2.5 Elitismo

A aplicação do método da roleta para a seleção de indivíduos pode causar a perda de alguns dos melhores indivíduos de uma população, visto que, existe a probabilidade deles não serem selecionados para a próxima geração [KOZA1992a]. Existe também a possibilidade do desaparecimento da elite da população em decorrência de operações genéticas. O conceito de Elitismo tem como objetivo garantir que os melhores indivíduos de uma geração permaneçam presentes na geração seguinte. Desta forma, os melhores indivíduos são mantidos na nova geração substituindo os piores indivíduos da nova geração [GOLDBERG1985; HOLLAND1975].

4.4.2.6 Recombinação (Crossover)

O operador genético *crossover* ou recombinação cria novos indivíduos a partir da combinação de dois ou mais indivíduos. Simulando o modelo biológico, o operador *crossover* age sobre os indivíduos da população (pais) gerando novos indivíduos (filhos). Os filhos são indivíduos formados exclusivamente pela combinação do material genético oriundo de seus pais. O objetivo da recombinação é a criação de novos indivíduos através da combinação de informação proveniente de diferentes soluções candidatas. Nos AGs clássicos costuma-se utilizar o operador *crossover* de um ponto com uma taxa fixa de probabilidade de ocorrência entre os indivíduos de uma população. Para este operador, são selecionados dois indivíduos (pais) que dão origem após a recombinação genética a dois novos indivíduos (filhos). A figura 4.4 ilustra o processo de *crossover* de um ponto aplicado a um par de indivíduos [HOLLAND1975].

Figura 4.4 – (A) mostra os dois pais selecionados para passarem pelo processo de recombinação e o ponto de corte, escolhido ao acaso, entre os genes 3 e 4. (B) Filhos provenientes dessa recombinação.

4.4.2.7 *Mutação*

A mutação modifica aleatoriamente um ou mais genes do cromossomo de um indivíduo. Define-se como taxa de mutação (TM) a porcentagem de genes da população que serão escolhidos para terem seus valores alterados. O objetivo do operador mutação é inserir diversidade na população de indivíduos sem desestabilizá-la. Por isso, a taxa de mutação usualmente utilizada é pequena, girando entre 0,01% e 1% do total de genes da população. Para uma população de indivíduos com codificação binária, o operador mutação descrito seleciona os genes aleatoriamente e simplesmente inverte o valor de cada gene (bit) selecionado [HOLLAND1975].

4.4.2.8 *CrITÉrios de Parada do Algoritmo Genético*

Nos AGs clássicos, costuma-se utilizar três critérios para a parada do sistema:

1. quando um número de gerações previamente definido é alcançado;
2. quando não há mais diversidade entre os indivíduos da população, fazendo com que o sistema não possa mais evoluir;
3. quando se atinge uma solução aceitável para o problema.

4.4.3 O Uso de AGs na Solução de Problemas

Nos últimos anos os AGs têm sido usados na solução de problemas nos mais diferentes campos da ciência e da engenharia. São muitos os exemplos: programação autônoma para desenvolver automaticamente estruturas computacionais, tais como, autômatos celulares [MITCHELL1992; MITCHELL1994] e arquiteturas para redes neurais artificiais [AZEVEDO1999]; problemas de otimização, incluindo otimização numérica e combinatória, projeto de circuitos e planejamento de produção [GRUAU1993]; modelos econômicos para descrição de processos, estratégias e análises de mercado [BRIAN1993; HAYWARD2004]; modelagem de sistemas imunológicos [BERSINI2002]; aprendizagem de máquina com inúmeras aplicações envolvendo sistemas de classificação e predição de tarefas em sistemas dinâmicos [MEYER1991]; aprendizagem em redes neurais artificiais [ATMAR1992; ATMAR1994]; modelagem de fenômenos ecológicos (simbiose, co-evolução, etc.) [KUBOTA1996]; modelos sociais [KURAHASHI1999; REGGIA2001]; desenvolvimento de jogos [MILES2004; RIBEIRO-FILHO1994; SMITH1980]; etc.

4.5 SISTEMAS CLASSIFICADORES

A escolha da representação das estruturas de dados é a chave nos AGs, uma vez que os AGs manipulam diretamente o código que representa os indivíduos candidatos a solução de um problema e, porque o comprimento fixo das cadeias que representam o indivíduo pode limitar e trazer dificuldades para a solução de alguns problemas, particularmente para aqueles cujo tamanho e a forma da solução desejada são desconhecidos. Assim, a necessidade de uma representação mais poderosa levou ao desenvolvimento de novos e interessantes paradigmas evolutivos como os Sistemas Classificadores (SCs) e a Programação Genética (PG) [TANOMARU1995]. Basicamente, os SCs são sistemas de produção (regras) adaptativos, em que as regras denominadas de classificadores são representadas por uma seqüência extraída de um alfabeto $A = \{a, b, c, \#\}$ e evoluem através da aplicação de um algoritmo genético [KOZA1992b; MCAULAY1994].

O capítulo 5 apresenta detalhes do desenvolvimento de um algoritmo baseado no paradigma do SC para identificação de procedimentos para a solução automática do problema proposto.

4.6 PROGRAMAÇÃO GENÉTICA

Uma extensão dos algoritmos genéticos, denominada programação genética, foi introduzida por John Koza [KOZA2003], e aplica os conceitos dos AGs em estruturas muito mais complexas. Na PG os indivíduos da população são estruturas (árvores computacionais de tamanhos variados) que representam partes de um programa. As estruturas ou árvores são compostas por vários nodos que podem conter operadores aritméticos, funções matemáticas, operadores lógicos e funções específicas para o domínio do problema estudado. A população de indivíduos é submetida a um algoritmo genético que opera basicamente com os operadores *crossover* e *mutação*, fazendo com que a população de programas evolua para a solução procurada. As figuras 4.5 e 4.6 apresentam respectivamente uma operação de *crossover* e *mutação* em um indivíduo de uma população.

Figura 4.5 – (A) mostra duas árvores representando os indivíduos de uma população e seus respectivos pontos de corte. (B) Mostra o resultado da operação de *crossover*.

Figura 4.6 – (A) um indivíduo da população e o nodo escolhido para mutação. (B) Mostra o resultado da mutação, em que o operador multiplicação foi substituído pelo operador adição.

Atualmente a programação genética tem sido aplicada a uma grande variedade de problemas, como na síntese de circuitos elétricos analógicos [KOZA2003] e na definição de arquiteturas de redes neurais artificiais [BANZHAF2000].

4.7 ESTRATÉGIAS EVOLUTIVAS

Estratégias Evolutivas (EE) surgiram na Alemanha independentemente dos estudos de Holland. No modelo mais simples um pai gera um filho através de aplicação de mutações cuja ocorrência é dada por uma distribuição Gaussiana. Deste modo, as pequenas mutações ocorrem mais freqüentemente que as mutações mais radicais. O processo consiste em substituir o pai pelo filho sempre que a adaptabilidade deste foi melhor do que a de seu pai. Este modelo levou ao resultado conhecido como regra do 1/5 de sucesso, já que aproximadamente 20% das mutações produzem filhos melhores que os pais [BÄCK1997].

4.8 PROGRAMAÇÃO EVOLUTIVA

Os métodos de Programação Evolutiva (PE) desenvolvidos inicialmente por Fogel [FOGEL1966], também surgiram independentemente dos trabalhos de Holland. A PE consiste em uma população de N indivíduos que são copiados para uma população temporária e sofrem mutações variáveis, não havendo mecanismos

de recombinação. Através de um torneiro estocástico são selecionados da população de $2N$ indivíduos os N indivíduos que formarão a geração seguinte [TANOMARU1995].

O próximo capítulo detalha os objetivos e a metodologia utilizada nesse trabalho e descreve o sistema implementado.

CAPÍTULO 5

MÉTODOS E ENSAIOS

Neste capítulo, descrevem-se a metodologia e os ensaios utilizados na abordagem do problema proposto.

5.1 INTRODUÇÃO

A investigação realizar-se-á através de diversos ensaios que envolvem jogadores humanos e virtuais. Os jogadores virtuais são representados por indivíduos oriundos dos diferentes algoritmos evolutivos avaliados nesse trabalho. Na primeira fase, são realizados dois ensaios: o ensaio I avalia o desempenho dos jogadores humanos e o ensaio II investiga a utilização de algoritmos puramente combinatórios. A partir dos resultados e conclusões extraídas dos ensaios I e II, são propostos novos modelos evolutivos que utilizam a combinação de operadores sem memória (ensaio III-a), operadores com memória (ensaio III.b) e operadores instanciados (ensaio IV) na solução do problema proposto (figura 5.1).

Figura 5.1 – Resumo dos ensaios realizados.

A figura 5.2 resume as principais abordagens utilizadas para jogadores humanos e virtuais durante os ensaios e as relações existentes entre elas:

- *jogador humano*, estratégias que podem variar de simples lances combinatórios (jogador iniciante) a utilização de um conjunto de operadores bem definido (jogador especialista);
- *jogador virtual*, estratégias que variam de lances combinatórios, implementados através de um algoritmo genético, a um conjunto de operadores selecionadas e combinadas através de um classificador genético.

A figura 5.2 mostra ainda um terceiro eixo, o qual está relacionado à presença e à quantidade de memória que os jogadores humanos ou virtuais possuem. Os resultados indicam que procedimentos gerais para solução do problema estão relacionados à capacidade de memória e de raciocínio dedutivo dos jogadores.

Figura 5.2 - Principais abordagens utilizadas para neste trabalho.

A seguir, são detalhados os ensaios, o sistema desenvolvido para suportar as simulações e os algoritmos evolutivos propostos.

5.2 JOGADOR HUMANO

Ensaio I (H-C e H-R) – Na primeira fase do projeto, os ensaios foram realizados com jogadores humanos. O propósito deste ensaio é analisar o desempenho de diferentes jogadores, identificar procedimentos e extrair uma lista de operadores básicos para a solução do problema. Além disso, os resultados apresentados pelos jogadores serão utilizados como base de comparação com os resultados obtidos através dos algoritmos evolutivos descritos nos ensaios III e IV. Os procedimentos utilizados por jogadores humanos podem variar de simples combinações de cores (H-C) a procedimentos complexos e gerais baseados em operadores explorando a avaliação de lances anteriores (H-R). O ensaio foi realizado com três jogadores com níveis diferentes de habilidades e cada um disputou sequencialmente 20 partidas no modo 4x6. Os resultados são apresentados no próximo capítulo (figura 6.2). A tabela 5.1 resume as características dos três jogadores:

Tabela 5.1 – Descrição dos três jogadores avaliados em 20 partidas (ensaio I).

Jogador	Nível
1	iniciante
2	intermediário
3	avançado ou especialista

5.3 JOGADOR VIRTUAL

O jogador virtual é a entidade que representa a melhor solução encontrada em cada uma das diferentes abordagens evolutivas apresentadas. Ele é, em todos os casos, o melhor indivíduo da população após o algoritmo evoluir por várias gerações.

Ensaio II (V-C) – O primeiro algoritmo evolutivo investigado consiste em um algoritmo genético [MITCHELL1999b]. O algoritmo procura a solução para o problema, recombina indivíduos de uma população de potenciais soluções. Cada indivíduo representa um único lance (seqüência de cores) e a cada geração o melhor indivíduo da população é apresentado e avaliado, atribuindo-se a ele um número correspondente de pinos pretos e brancos. O objetivo é encontrar uma solução através da simples recombinação de cores, o que pode ser comparado à estratégia utilizada por um jogador iniciante. O algoritmo genético é apresentado em três versões que diferem quanto ao objetivo e são descritas na tabela 5.2. Cada uma delas é relacionada a uma diferente abordagem para conduzir o processo de geração e seleção dos indivíduos e tem por objetivo acompanhar de uma ou outra forma o jogador humano.

Tabela 5.2 – Descrição do ensaio II - Algoritmo Genético.

ALGORITMO	DESCRIÇÃO
AG1	Procura solução - busca combinatória, não há relação com a estratégia do jogador humano.
AG2	Segue passos do jogador humano – busca combinatória que procura <u>reproduzir as mesmas cores</u> utilizadas em cada um dos lances realizados pelo jogador humano.
AG3	Segue passos do jogador humano – busca combinatória que procura <u>reproduzir os acertos e erros</u> (pinos pretos e brancos) obtidos em cada um dos lances realizados pelo jogador humano.

O propósito desse ensaio é explorar abordagens puramente evolutivas que possam auxiliar na especificação de um modelo evolutivo mais amplo, capaz de extrair, por comparação entre a ação dos jogadores humano e virtual, o procedimento utilizado pelo jogador humano para a solução do problema. As três versões do algoritmo genético descrito na tabela 5.2 diferem quanto ao objetivo e procuram de forma gradativa aproximar-se do procedimento apresentado pelo jogador humano, ainda que através de um método combinatório. A primeira versão (AG1) busca apenas encontrar a solução do problema e não possui vínculo com a solução apresentada pelo jogador humano. A segunda versão (AG2) procura

reproduzir exatamente cada lance do jogador humano (cor e posição). A terceira versão (AG3) busca através da combinação genética de lances reproduzir a mesma pontuação obtida em cada lance pelo jogador humano (pinos pretos e brancos). É importante ressaltar que este ensaio não apresenta nenhum processo de extração de procedimentos de raciocínio, pois as soluções não apresentam um conjunto de procedimentos que possa ser repetido para solucionar o problema. Entretanto, os resultados desse ensaio indicam um possível caminho de investigação, descrito pelas três versões, para se extrair procedimentos gerais de solução do problema. Além disso, o algoritmo genético utilizado nesse ensaio será implementado, com algumas adaptações, nos ensaios III e IV, dando origem ao classificador genético descrito na próxima seção.

Ensaio III-a (V-R) – O ensaio consiste na tentativa de gerar uma solução geral para o jogo *Mastermind*. Para isso, as soluções combinatórias serão guiadas por uma versão modificada do algoritmo genético denominada Classificador Genético (CG). No classificador genético o conjunto de candidatos à solução do problema, que formam a população inicial do algoritmo genético utilizado no ensaio II é substituído por um conjunto de operadores. Cada indivíduo da população passa a ser formado por um conjunto de operadores simples (operadores sem memória) que quando combinados e executados sequencialmente devem ser capazes de solucionar o problema. Inicialmente, cada conjunto de procedimentos ou indivíduo consiste de uma sequência de operadores selecionados aleatoriamente de uma lista previamente extraída dos ensaios realizados com jogadores humanos e descritos na seção 5.1 (ensaio I). Nesse ensaio foram repetidas as três abordagens algorítmicas descritas no ensaio II (tabela 5.2).

Nesse ensaio avalia-se um algoritmo evolutivo que manipula operadores, diferentemente do ensaio II, que manipulava diretamente os lances candidatos a soluções do problema (conjunto de cores). O algoritmo utilizado recombina, através de um algoritmo genético, um conjunto de operadores que descreve os procedimentos adotados pelos jogadores humanos para a realização de cada um dos lances de uma partida. Assim como no ensaio II, as três versões do algoritmo utilizado caminham no sentido de aproximar a solução encontrada da solução apresentada pelo jogador humano. A versão CG1S procura um conjunto de

operadores que soluciona o problema, mas não tendo relação com o procedimento realizado pelo jogador humano. A versão CG2S busca um conjunto de procedimento capaz de reproduzir os mesmos lances do jogador humano (cores e posições). A terceira versão (CG3S) procura um conjunto de operadores que repita apenas os resultados obtidos pelo jogador humano (pinos pretos e brancos). O objetivo é verificar se o indivíduo vencedor após várias gerações poderá ser um bom representante do processo de raciocínio humano que ele tentou reproduzir.

Ensaio III-b (V-R) – A partir dos resultados obtidos nos ensaios II e III-a e das conclusões extraídas do ensaio I, discutidas no capítulo 7, observou-se que os procedimentos gerais para a solução do problema não podem ser construídos automaticamente sem que o sistema proposto possua um mecanismo de memória que considere os lances anteriores. Isso se deve ao fato da existência de um grau de aleatoriedade entre os lances que impede que o algoritmo convirja para uma solução geral. Com base nessas evidências foi proposta uma nova versão para o classificador genético, sendo o conjunto de operadores simples (sem memória), que constituíam a população inicial na versão III-a, substituído por um conjunto de operadores com memória. Os operadores com memória verificam se o lance anterior obteve ganhos quando comparado com o seu antecessor, caso contrário, utiliza o antecessor para gerar um novo lance. Novamente, serão repetidas as três avaliações algorítmicas realizadas no ensaio II e III-a (tabela 5.2).

Ensaio IV (V-R) – A análise dos resultados do ensaio III (discutidos no capítulo 7) mostrou uma dificuldade na convergência do algoritmo ainda decorrente do grau de aleatoriedade existente no conjunto de operadores extraído das partidas com jogadores humanos. Além disso, a inclusão de memória nos operadores - proposta na versão III-b - não se mostrou suficiente para reduzir o grau de incerteza e permitir que se avalie o grau de generalização das soluções encontradas. Desta forma, esse novo ensaio apresenta um conjunto de operadores instanciados, cada operador foi instanciado de forma a representar um único lance possível (lances ótimos). O propósito desse ensaio é contornar a aleatoriedade presente nos operadores gerais do ensaio III e mostrar a possibilidade de convergir o novo algoritmo para uma solução que se aproxime da solução utilizada pelo jogador

humano. O ensaio IV também é apresentado em três versões conforme descrito na tabela 5.2.

Nas seções seguintes, são apresentados a interface do sistema e os detalhes de implementação do Algoritmo Genético e do Classificador Genético utilizados nos ensaios II, III e IV.

5.4 DESCRIÇÃO DO SISTEMA PROPOSTO

A figura 5.3 mostra a interface desenvolvida para execução dos ensaios descritos neste trabalho. A interface consiste de dois tabuleiros, um para o jogador humano e outro para o jogador virtual. Cada jogador pode realizar até 24 lances por partida na tentativa encontrar o código secreto. Cada lance realizado é avaliado pelo sistema e seu resultado é apresentado no formato de marcadores pretos e brancos (indicando o número de cores corretas na posição correta e o número de cores corretas na posição errada respectivamente). Ao lado dos tabuleiros vê-se um painel de ferramentas com cinco grupos:

Preferências Gerais - permite alterar parâmetros gerais de execução do sistema: o número de cores que constará da lista de sorteio para criação do código secreto, o número de cores presentes no código secreto, se existirão cores repetidas no código secreto e número de indivíduos por geração.

Ferramentas – contém alguns utilitários do sistema como gerar um novo código secreto, visualizar o código secreto e registrar um novo jogador.

Algoritmos – contém as cinco modalidades de execução do jogo:

1. jogador humano;
2. algoritmo genético;
3. classificador genético (operadores sem memória);
4. classificador genético (operadores com memória).

5. classificador genético (conjunto de operadores instanciados).

Propriedades – permite configurar propriedades específicas de cada modalidade de execução, como por exemplo, a função *fitness*.

Figura 5.3 – Interface do sistema desenvolvido para realização dos ensaios.

O sistema possui quatro módulos que se relacionam conforme descrito pela figura 5.4.

Figura 5.4 – Módulos do sistema e suas interconexões.

A funcionalidade de cada módulo é descrita a seguir:

5.4.1 Módulo Especialista

É responsável por gerar aleatoriamente o código secreto de uma partida; avaliar cada lance segundo as regras do jogo e enviar o resultado da avaliação ao módulo de controle.

5.4.2 Módulo de Controle

Responsável pela apresentação gráfica do jogo, interação com o jogador humano e encaminhamento dos lances executados pelo jogador humano ou virtual ao Módulo Especialista. O módulo de controle possui ainda as funções de inicialização do jogo e inscrição de um novo jogador, sendo também responsável pelo armazenamento e recuperação dos lances de uma partida. A tabela 5.3 mostra a estrutura de um desses arquivos de armazenamento. Os quatro algarismos que aparecem nos campos *Código Secreto* e *Lance* referem-se às cores existentes no alfabeto utilizado para gerar o código secreto (tabela 5.4).

Tabela 5.3 – Estrutura de um arquivo texto contendo os dados referentes a duas partidas de um mesmo jogador.

Número da Partida	Número do lance	Código Secreto	Lance	Marcadores Pretos	Marcadores Brancos
1	1	6,2,1,3	1,2,2,1	1	1
1	2	6,2,1,3	1,2,3,4	1	2
1	3	6,2,1,3	1,2,5,3	2	1
1	4	6,2,1,3	1,6,2,3	1	3
1	5	6,2,1,3	1,3,6,2	0	4
1	6	6,2,1,3	6,2,1,1	3	1
1	7	6,2,1,3	6,2,1,3	4	0
2	1	6,3,2,5	1,1,2,2	1	0
2	2	6,3,2,5	6,1,3,4	1	1
2	3	6,3,2,5	6,5,3,2	1	3
2	4	6,3,2,5	3,5,2,6	1	3
2	5	6,3,2,5	6,3,2,5	4	0

Tabela 5.4 – Tabela de codificação de cores para o *Mastermind*.

TABELA DE CORES	
código	cor
1	amarela
2	vermelha
3	azul escura
4	rosa
5	laranja
6	azul clara
7	verde
8	branca

5.4.3 Módulo Algoritmo Genético

Contém os códigos de execução do algoritmo genético (AG). O algoritmo genético implementado é utilizado no ensaio II algoritmos (AG1, AG2 e AG3) e nos ensaios III e IV compartilhando a estrutura de execução e algumas funções básicas (*crossover*, *mutação* e *seleção*) com o classificador genético descrito na próxima seção. Detalhes da implementação do algoritmo genético são discutidos na seção 5.5.

5.4.4 Módulo Classificador Genético

O módulo classificador genético contém as funções de execução do Classificador Genético (CG). O classificador genético consiste de um algoritmo genético modificado. No classificador genético os candidatos à solução do problema que formam a população inicial do algoritmo genético utilizado no ensaio II são substituídos por um conjunto de operadores ou procedimentos. Cada indivíduo da população é formado por um conjunto de operadores que executados seqüencialmente devem ser capazes de solucionar o problema. Os detalhes de implementação do classificador genético nas diferentes modalidades são apresentados na seção 5.6.

5.5 DETALHES DO ALGORITMO GENÉTICO

No algoritmo genético implementado, a população inicial é fixa e gerada aleatoriamente. Cada indivíduo representa um lance do jogo – uma combinação de cores que pode ser a combinação procurada – e é avaliado através de uma função que mede o quanto cada lance respeita os ganhos já obtidos nos lances anteriores [BENTO1999]. A cada geração o indivíduo com a melhor avaliação (*fitness*) é escolhido como próximo lance a ser executado. No final de uma partida tem-se a apresentação de uma seqüência com os melhores indivíduos que vão da primeira geração (lance inicial) até a última geração (solução do problema). A figura 5.5 mostra uma partida resolvida automaticamente após o algoritmo genético ter evoluído por seis gerações.

Figura 5.5 – Lances de uma partida executada pelo algoritmo genético.

O estudo da solução do problema proposto através da utilização de um algoritmo genético (ensaio II) é utilizado como primeira aproximação para se chegar a uma solução geral e que guarde semelhanças com os procedimentos apresentados por um jogador humano. Além disso, as funções desenvolvidas para o algoritmo genético são parametrizadas para serem utilizadas pelo classificador genético (ensaios III e IV). A seguir, apresentam-se os detalhes do algoritmo genético implementado.

5.5.1 Codificação

Na implementação do algoritmo, cada pino colorido é considerado um gene e a seqüência de N pinos coloridos é um cromossomo que representa um indivíduo da população de candidatos à solução do problema. A população inicial contém um número n de indivíduos, cada um representando um possível lance para o jogo. Cada indivíduo ou lance é codificado na forma de um vetor de números inteiros e gerado aleatoriamente a partir de um alfabeto de cores com cardinalidade L (L , N e n são configuráveis através do *Painel de Controle*). A figura 5.6 ilustra o processo de geração da população inicial.

Figura 5.6 – Geração da população inicial contendo n indivíduos.

5.5.2 Seleção

A seleção dos indivíduos que participarão do processo de *crossover* e mutação consiste do sorteio de $n/2$ pares através da técnica da roleta descrita na seção 5.5.2.4.

5.5.3 Elitismo

Após o processo de geração dos n filhos, aplica-se a todos os indivíduos o conceito de elitismo, ordenando-se os $2n$ indivíduos (n pais + n filhos) e selecionando os n melhores indivíduos que formarão a nova geração. Isto garante a sobrevivência dos melhores indivíduos nas gerações seguintes.

5.5.4 Reprodução

A cada indivíduo da população é associada uma probabilidade de reprodução dada pela equação 5.1:

$$PR_i = \frac{f_i}{\sum_{i=1}^N f_i} \quad \text{sendo } f \text{ o fitness do indivíduo } i. \quad (5.1).$$

5.5.5 Função *Fitness*

O cálculo do *fitness* permite selecionar o melhor indivíduo de cada geração, o qual será escolhido como o próximo lance em uma partida do *Mastermind*. Como descrito na seção 5.2, o algoritmo genético é apresentado em três versões e a definição de melhor indivíduo de cada geração varia de acordo com a versão do algoritmo:

AG1 – define-se como melhor indivíduo aquele cuja seqüência de cores mantém os ganhos obtidos nos lances anteriores.

AG2 – o melhor indivíduo é aquele cuja seqüência de cores é a mais próxima da seqüência de cores utilizada pelo jogador humano naquele lance.

AG3 – o melhor indivíduo é aquele cuja avaliação (pinos pretos e brancos) é a mais próxima da avaliação obtida pelo jogador humano naquele lance.

Para o AG1 o objetivo é medir o quanto cada indivíduo da população respeita os ganhos já obtidos em todos os lances anteriores, para isso, realiza-se o seguinte procedimento para cada indivíduo da população:

- Primeiro, mede-se a similaridade entre o indivíduo i e cada um dos m lances já executados pelo algoritmo genético. A medida da similaridade é realizada confrontando-se o indivíduo i com o lance m e anotando-se a pontuação obtida (pinos pretos e brancos (XP_i^m, XB_i^m)).
- Em seguida, calcula-se a distância euclidiana entre os resultados obtidos no item anterior (XP_i^m, XB_i^m) e o resultados obtidos pelos m lances quando confrontado com o código secreto (NP_m, NB_m) equação 5.2.
- O *fitness* do indivíduo i é então calculado somando-se as m distâncias conforme equação 5.3.

$$d_i^m = \sqrt{(NP_m - XP_i^m)^2 + (NB_m - XB_i^m)^2} \quad (5.2)$$

$$f(i) = \sum_{m=1}^n d_i^m \quad (5.3)$$

A figura 5.7 e a tabela 5.5 exemplificam o cálculo do *fitness* para um indivíduo da população.

Figura 5.7 – Exemplo de cálculo do *fitness* para um indivíduo algoritmo genético.

A figura 5.7 (lado esquerdo) apresenta os quatro primeiros lances executados pelo algoritmo genético e alguns indivíduos da próxima geração (lado direito). Os dados da tabela 5.5 exemplificam o cálculo do *fitness* para o primeiro indivíduo da 4ª geração ($i=1 \Rightarrow 2456$) utilizando-se a equação 5.3.

Tabela 5.5 – Tabela auxiliar para o cálculo do *fitness*.

A	B	C	D	E	F	G	H
m	(NP_m, NB_m)	Lance	Indivíduo $i=1$	(XP_i^m, XB_i^m)	$ NP_m - XP_i^m $	$ NB_m - XB_i^m $	$f(1) = \sum_{m=1}^n d_i^m$
1	1,0	1225	2456	0,2	1	2	5,47
2	2,1	3164		0,2	2	1	
3	0,3	4563		0,3	0	0	
4	1,3	2643		1,2	0	1	

Coluna A – indica o índice dos lances executados pelo algoritmo genético.

Coluna B – apresenta a pontuação obtida pelos lances do algoritmo genético quando comparado ao código secreto (figura 5.7).

Coluna C – Lances executados pelo algoritmo genético. Os lances representam o melhor indivíduo de cada geração executada.

Coluna D – Indivíduo $i=1$ da população a ser avaliada.

Coluna E – Apresenta a pontuação obtida pelo indivíduo i quando comparado ao lance m executado pelo algoritmo genético. Exemplo: comparando o indivíduo $i=1$ (2456) com o lance $m=1$ (1225) observa-se que há duas cores corretas em posições erradas ($XB_1^1 = 2$) e nenhuma cor correta na posição correta ($XP_1^1 = 0$), portanto a avaliação do indivíduo $i=1$ quando comparado ao lance $m=1$ é $(XP_1^1, XB_1^1) = (0, 2)$.

Coluna F – Apresenta o módulo da diferença entre o número de cores corretas na posição correta do lance m quando comparado ao código secreto (NP), e o número de cores corretas na posição correta do indivíduo i , quando comparado ao lance m (XP).

Coluna G – Apresenta o módulo da diferença entre o número de cores corretas na posição incorreta do lance m quando comparado ao código secreto (NB), e o número de cores corretas na posição incorreta do indivíduo i , quando comparado ao lance m (XB).

Coluna H – Apresenta o cálculo final do fitness do indivíduo i obtido pela equação 5.2.

O procedimento descrito refere-se ao cálculo do *fitness* para a versão AG1, as versões AG2 e AG3 têm cálculo simplificado, pois nesses casos, busca-se o indivíduo com melhor classificação quando comparado, respectivamente, com o lance m (mesmas cores) e com o resultado do lance m (mesmo número de pinos pretos e brancos).

O apêndice A apresenta um comparativo entre o *fitness* euclidiano e um segundo método disponível na literatura, o objetivo dessa comparação é validar a

utilização do método proposto o qual também será utilizado para o Classificador Genético descrito no item 5.6.

5.5.6 Crossover Condicional

Nesta fase do trabalho, optou-se por implementar um modelo de *crossover* presente na literatura e denominado *Crossover Condicional*. O operador consiste na introdução de uma heurística para recombinação dos pares que, baseado no *fitness* dos indivíduos selecionados, altera o comportamento do operador *crossover*.

O processo de *crossover* dos pares selecionados consiste das seguintes etapas:

1. Define-se XP_i e XB_i como sendo o número de pinos pretos e brancos, respectivamente, atribuído ao indivíduo i da população, depois da seleção, por comparação como o último lance executado pelo algoritmo genético.

2. Calcula-se para cada par (i,j) de indivíduos selecionados para o *crossover* os parâmetros HP e HB conforme equações 5.4 e 5.5.

$$HP_i = \min(XP_i, NP) + \min(XB_i, NB); \quad HP_j = \min(XP_j, NP) + \min(XB_j, NB) \quad (5.4)$$

$$HB_i = |NB - XB_i|; \quad HB_j = |NB - XB_j| \quad (5.5)$$

Sendo HP_i e HP_j o número de pinos que podem estar com cor e posição corretas (marcador preto) nos indivíduos i e j e HB_i e HB_j os pinos que podem estar com a cor correta mais na posição errada.

3. Seleciona-se aleatoriamente HP_i posições do indivíduo i que são copiadas (cor e posição) para o novo indivíduo.

4. Seleciona-se aleatoriamente HB_i posições do indivíduo i que são transferidas para o novo indivíduo em posições selecionadas aleatoriamente entre as posições ainda não ocupadas. Isso significa que a cor estará presente no novo indivíduo, mas em uma nova posição.

5. Sorteia-se $(P - HP_i - HB_i)$ posições do indivíduo j que são copiadas para as posições ainda não ocupadas no novo indivíduo.

O segundo indivíduo é gerado pelo mesmo método, trocando-se as funções dos indivíduos i e j . A figura 5.8 exemplifica a criação de um novo indivíduo.

Figura 5.8 – Exemplo de aplicação do *crossover* condicional

O operador *crossover* opera diferentemente no caso de todas as cores do código secreto já terem sido descobertas ($P - HP_i - HB_i = 0$), neste caso, somente o último lance é utilizado e os novos indivíduos gerados são clones desse indivíduo (figura 5.9) [BENTO1999].

Figura 5.9 – Exemplo de aplicação do *crossover* condicional para o caso em que todas as cores do código secreto já foram descobertas.

5.5.7 Mutação

O algoritmo genético possui uma taxa de mutação de 0,5%. O processo de mutação consiste em selecionar ao acaso 0,5% dos genes da população e substituí-lo por outro gene também escolhido ao acaso da tabela 5.4.

5.5.8 Heurísticas

Segundo Puchkin, heurística é um conjunto de técnicas destinadas a solucionar o problema de resolver problemas [PUCHKIN1969]. As heurísticas também podem ser entendidas como funções que funcionam como conhecimento à priori do problema e aumentam a eficiência de um sistema de busca. No algoritmo genético descrito, optou-se por implementar apenas uma heurística para o cálculo do operador *crossover* com o objetivo de aumentar em algum grau a sua eficiência e permitir comparações com outros trabalhos que, assim como no ensaio II, procuram a solução do problema através da manipulação direta de indivíduos candidatos à solução [BENTO1999; MERELLO1996; MERELLO1999]. A heurística descrita é utilizada apenas no algoritmo genético implementado no ensaio II e não está presente nos ensaios III, IV. A opção pela não utilização de heurísticas nos ensaios III e IV deve-se ao compromisso de manter os modelos estudados o mais próximo das características originais de um algoritmo evolutivo.

5.6 DETALHES DO CLASSIFICADOR GENÉTICO

O classificador genético foi desenvolvido utilizando-se as mesmas técnicas de seleção, elitismo, reprodução, *fitness* e mutação, implementadas no *Módulo AG* e descritas nas seções 5.5.2, 5.5.3, 5.5.4, 5.5.5 e 5.5.6 respectivamente. Seu diferencial encontra-se na população de indivíduos que formam a população inicial. No classificador genético cada indivíduo representa uma partida completa e é formado por um conjunto de operadores (genes) que, se executados seqüencialmente, podem levar a solução do problema. Assim como no algoritmo genético o classificador genético utiliza uma função de mérito (*fitness*) que avalia cada um dos indivíduos da população e atribui uma nota, permitindo a seleção dos melhores indivíduos os quais serão submetidos ao processo de seleção, recombinação e mutação. Os detalhes classificador genético descritos nas seções seguintes são válidos para os ensaios III-a (CG sem memória), III-b (CG com memória) e IV (CG com operadores instanciados).

5.6.1 Algoritmo do Classificador Genético

O *Módulo Classificador Genético* ao ser acionado executa os seguintes procedimentos:

1. Gera uma população inicial de n indivíduos (conjunto de operadores).
2. Envia cada um dos indivíduos da população (conjunto de operadores que descreve uma partida completa) ao módulo Especialista que os executa e grava a pontuação obtida em cada um dos lances da partida.
3. Submete o resultado de cada indivíduo da população à função de avaliação.
4. Ordena os indivíduos, do maior para o de menor *fitness* e seleciona os pares através do método da roleta.
5. Aplica os operadores *crossover* e mutação gerando n filhos.
6. Avalia cada um dos n filhos (passos 2 e 3).
7. Reordena os $2n$ indivíduos (n pais + n filhos) e seleciona os n melhores para formar a próxima geração (elitismo).
8. Verifica se o critério de parada (um número k de gerações ou valor da similaridade). Se o critério for atingido apresenta a melhor estratégia e finaliza.

A tabela 5.6 apresenta o algoritmo simplificado dos procedimentos descritos:

Tabela 5.6 – Algoritmo executado pelo módulo classificador genético.

```

classificador_genético {
    gerar população inicial, composta por  $n$  indivíduos;
    calcular fitness dos indivíduos da população;
    ordenar os indivíduos, do menor para o de maior fitness;
    enquanto (critério de parada não for atingido){
        selecionar os indivíduos pelo método da roleta;
        gerar os filhos;
        calcular fitness dos filhos;
        avaliar os  $n$  filhos;
        ordenar os  $2n$  indivíduos ( $n$  pais +  $n$  filhos);
        selecionar os  $n$  melhores para a próxima geração;
        Reordenar e selecionar os  $n$  melhores indivíduos;
    } //fim enquanto
} //fim

```

5.6.2 População inicial

A população inicial é formada por um conjunto de indivíduos candidatos à solução do problema. Cada indivíduo é formado por uma seqüência de operadores que descreve uma partida completa. Os operadores são extraídos aleatoriamente do conjunto de procedimentos básicos descritos nas tabelas 5.7 (ensaio III-a e III-b) e 6.14 (ensaio IV). É importante ressaltar que no ensaio III-a os operadores não possuem nenhum mecanismo de memória e são baseados em procedimentos gerais extraídos a partir da observação dos lances realizados durante as partidas disputadas pelos jogadores humanos no ensaio I.

Tabela 5.7 – Lista de operadores para solução do *Mastermind*.

Lista de Operadores	
a	Gera um lance com n cores escolhidas ao acaso.
b	Gera um lance com uma única cor escolhida ao acaso.
c	Gera um lance com uma cor qualquer repetida k vezes (k é um número aleatório).
d	Gera um lance com uma cor qualquer repetida $(n - p - b)^1$ vezes.
e	Repete o último lance trocando k cores (k é um número aleatório).
f	Repete o último lance trocando k cores de posição (k é um número aleatório).
g	Repete o último lance trocando b cores de posição.
h	Repete o último lance trocando $(n - p - b)$ cores escolhida ao acaso.
i	Repete o ultimo lance trocando b cores de posição e substituindo $(n - p - b)$ cores por uma única cor escolhida ao acaso.
j	Repete o ultimo lance trocando b cores de posição e substituindo $(n - p - b)$ cores por outras cores escolhidas aleatoriamente.

O tamanho de cada indivíduo da população, isto é, o número de lances que cada indivíduo possui depende da versão do classificador genético que está sendo executada. Na versão 1 (CG1S, CG1M e CG1E), o número de lances máximo é parametrizável e nas versões 2 e 3 (CG2S, CG2M, CG2E, CG3S, CG3M e CG3E) o número de lances é igual ao número de lances utilizados pelo jogador para encontrar o código secreto. Exemplo: se um jogador solucionar o jogo em 7 lances,

¹ b e p representam respectivamente o número de pinos pretos e brancos no último lance.

a população inicial será gerada com indivíduos contendo 7 genes, cada gene representando um lance ou operador extraída da tabela 5.7. Portanto neste caso o indivíduo representa toda uma partida e os operadores descritos são válidas para os ensaios III-a, III-b e IV.

5.6.3 Função *Fitness*

As simulações realizadas como o classificador genético utilizam a função *fitness* descrita no item 5.5.5. O classificador genético é apresentado em três versões e a definição de melhor indivíduo de cada geração varia de acordo com a versão do algoritmo:

1 - Para a versão 1 (CG1S, CG1M e CG1E), define-se como melhor indivíduo aquele cuja seqüência de operadores leva a lances que mantenham os ganhos obtidos nos lances anteriores.

2 – Na versão 2 (CG2S, CG2M e CG2E) o melhor indivíduo é aquele cuja seqüência de operadores leva a lances que reproduzam as mesmas cores dos lances do jogador humano.

3 – o melhor indivíduo na versão 3 (CG3S, CG3M e CG3E) é aquele cuja seqüência de regas conduz a lances que reproduzam os ganhos obtidos em cada lance do jogador humano.

A seção seguinte descreve-se os detalhes para o cálculo do *fitness* por medida de distância tendo como base no ensaio I, versão 3 (CG3S)² do classificador genético (sem memória). Nessa versão, um indivíduo representa a estratégia completa de uma partida e o cálculo do *fitness* mede a similaridade dessa estratégia com a estratégia do jogador humano.

5.6.3.1 CG3S – Cálculo da Função *Fitness*

A função de avaliação tem por objetivo calcular a similaridade entre a estratégia do jogador humano e a de um indivíduo qualquer da população de

---

² Os detalhes do cálculo do *fitness* para os algoritmos do classificador genético versão 1 e versão 2 não são apresentados. Optou-se por ilustrar o processo com a versão CG3S do algoritmo.

soluções. A similaridade é avaliada medindo-se à distância euclidiana entre os lances efetuados pelo jogador e os respectivos lances efetuados por cada um dos indivíduos da população da seguinte forma: primeiro, cada lance m do jogador h (jogador humano) é representado por um par ordenado $(P, B)_m^h$ que corresponde ao número de marcadores pretos e brancos obtidos quando o lance é confrontado com a senha secreta. Em seguida, cada indivíduo v (jogador virtual) da população é executado e o resultado de cada lance m é representado por um par ordenado $(P, B)_m^v$. O par ordenado $(P, B)_m^v$ representando o número de marcadores brancos e pretos que o lance m do indivíduo v obteve quando comparado com a senha secreta. O *fitness* do indivíduo v é atribuído somando-se às distâncias entre os respectivos lances do jogador h e do indivíduo v (equação 5.11). Quanto menor o *fitness* obtido por um indivíduo, mas próxima à estratégia representada por ele estará da estratégia utilizada pelo jogador. A figura 5.10 mostra os lances de uma partida executada por um jogador humano e a representação gráfica dos lances de um indivíduo qualquer da população, a figura também exemplifica o cálculo do *fitness* para este indivíduo.

Figura 5.10 – Cálculo do *fitness* para um indivíduo da população de soluções.

$$f(v, h) = \sum_{m=1}^m d_m^{vh} = \sum_{m=1}^m \sqrt{(P_m^v - P_m^h)^2 + (B_m^v - B_m^h)^2} \quad (5.5)$$

sendo:

$f(v, h)$ = valor do *fitness* do indivíduo v da população em relação ao jogador h .

d_m^{vh} = distância entre o resultado obtido pelo jogador h no lance m , representado pelo par ordenado $(P, B)_m^h$ e o resultado obtido pelo indivíduo v da população no lance m e representado pelo par ordenado $(P, B)_m^v$.

A figura 5.11 apresenta graficamente a medida de similaridade entre 5 lances de um indivíduo qualquer da população de soluções e os respectivos lances da estratégia do jogador humano. Observe que o jogador virtual inicia a partida reproduzindo o mesmo lance do jogador humano, este procedimento foi adotado para permitir uma melhor comparação entre as estratégias uma vez que o lance inicial é um lance aleatório não possuindo parâmetro de comparação.

Figura 5.11 – Representação gráfica da medida da similaridade entre os lances de um indivíduo qualquer da população de soluções e os lances do jogador humano.

5.6.4 Crossover

O Classificador Genético utiliza o *crossover* de um ponto descrito na seção 5.4.2.6. São selecionados dois indivíduos (pais) que dão origem após a recombinação genética a dois novos indivíduos (filhos). Ressalta-se que diferentemente do operador utilizado no ensaio II e descrito na seção 5.5.6, o

crossover utilizado pelo classificador genético não faz uso de nenhuma heurística durante o processo de reprodução. A figura 5.12 ilustra o processo de *crossover* de um ponto aplicado a um par de indivíduos, nesse exemplo, cada indivíduo é formado por 8 genes, cada gene representa um operador extraído da tabela 5.7.

Figura 5.12 – (A) mostra o processo de recombinação e o ponto de corte, escolhido ao acaso, entre os genes 3 e 4. (B) Filhos provenientes dessa recombinação.

5.6.5 Mutação

O Classificador Genético possui uma taxa de mutação de 0,5%. O processo de mutação consiste em selecionar ao acaso 0,5% dos genes (operadores) da população e substituí-lo por outro gene também escolhido ao acaso das tabelas 5.7 (ensaio III-a e III-b) e 6.14 (ensaio IV).

5.6.6 Critérios de Parada

São dois os critérios de parada adotados para o classificador genético: 1. número de gerações, definido através da opção *Propriedades* do *Painel de Controles* do sistema. 2. quando o SC encontra uma solução para o problema com similaridade 1 (100%) quando comparada a solução apresentada pelo jogador humano.

No próximo capítulo são apresentados e analisados os resultados dos ensaios descritos.

CAPÍTULO 6

RESULTADOS E AVALIAÇÕES DOS ENSAIOS

Neste capítulo são apresentados os resultados obtidos nos diferentes ensaios realizados nessa pesquisa. Note que a proposta desse trabalho é investigar a utilização de modelos evolutivos para identificação de procedimentos utilizados por humanos para a solução de problemas de raciocínio dedutivo. Os diferentes ensaios foram realizados com o objetivo de permitir a análise e proposição gradativa de uma solução para o problema, a tabela 6.1 apresenta um resumo das características e algoritmos dos diferentes ensaios.

Tabela 6.1 – Resumo dos ensaios e algoritmos.

ENSAIOS				
Jogador Humano	Jogador Virtual			
I	II	III-a	III-b	IV
Comb./Oper. H-C / H-R	Combinatório V-C	Operadores V-R (S/M/E)		
Algoritmo Humano	Algoritmo Genético	Classificador Genético operadores básicos sem memória	Classificador Genético operadores básicos com memória	Classificador Genético operadores instanciados sem memória
Ensaio com 3 jogadores:	Algoritmo combinatório em 3 versões:	Algoritmo que recombina por meio de um AG operadores sem memória em 3 versões:	Algoritmo que recombina por meio de um AG operadores com memória em 3 versões:	Algoritmo que recombina por meio de um AG um conjunto instanciados de operadores sem memória em 3 versões:
Iniciante Intermediário Especialista	AG1 AG2 AG3	CG1S CG2S CG3S	CG1M CG2M CG3M	CG1E CG2E CG3E

6.1 ENSAIO I - ANÁLISE DO DESEMPENHO DE JOGADORES HUMANOS

A seguir, apresentam-se os resultados obtidos na disputa de vinte partidas sucessivas realizadas por três jogadores com níveis de habilidades diferentes. As partidas foram disputadas utilizando-se códigos secretos com quatro cores extraídas de um alfabeto de seis cores (4x6), não havendo repetição de cores no código secreto. Para garantir condições iguais aos jogadores e permitir uma melhor comparação entre os resultados foram utilizados os mesmos códigos e seqüência de apresentação para os três jogadores. Os histogramas das figuras 6.4, 6.6 e 6.8 descrevem a distribuição de lances utilizados pelos jogadores até a descoberta do código secreto e as figuras 6.5, 6.7 e 6.9 apresentam a variação do desempenho do jogador ao longo das vinte partidas.

Figura 6.1 – Distribuição do número de lances necessários para solução do jogo em 20 partidas realizadas pelo jogador 1 (iniciante).

Figura 6.2 – Variação do desempenho do jogador 1 (iniciante) em 20 partidas.

Figura 6.3 – Distribuição do número de lances necessários para solução do jogo em 20 partidas realizadas pelo jogador 2 (intermediário).

Figura 6.4 – Variação do desempenho do jogador 2 (intermediário) em 20 partidas.

Figura 6.5 – Distribuição do número de lances necessários para solução do jogo em 20 partidas realizadas pelo jogador 3 (avançado).

Figura 6.6 – Variação do desempenho do jogador 3 (avançado) em 20 partidas.

Analisando-se os histogramas das figuras 6.1, 6.3 e 6.5, observa-se que à medida que os jogadores utilizam-se de alguma estratégia ou procedimento para encontrar a solução, menor é a variação do número de lances utilizados em cada partida. Em outras palavras, quanto melhor ou mais sistemática é a estratégia de solução do jogador, menor é o desvio padrão da sua distribuição de lances (tabela 6.2). Esse resultado permite classificar os três jogadores segundo seu nível de habilidade:

Jogador 1 – a observação das partidas do jogador 1, mostrou que o mesmo não utilizou nenhuma estratégia sistemática para a solução do problema. Além disso, o acompanhamento visual das partidas mostrou uma dificuldade em recuperar, relacionar e utilizar os resultados dos lances anteriores para construir o lance seguinte. Por essa razão, o jogador 1 é apresentado na tabela 6.2 como um jogador iniciante.

Jogador 2 - a observação das partidas do jogador 2, mostrou que o mesmo não utilizou nenhuma estratégia sistemática na solução do problema, iniciando sempre com lances aleatórios. Entretanto, o jogador 2 fez uso intenso da sua capacidade de raciocínio dedutivo para a tomada de decisão, no que diz respeito ao próximo lance a ser executado. Desta forma, o jogador 2 foi classificado como um

jogador de nível intermediário por não usar nenhuma estratégia sistemática de solução.

Jogador 3 – o jogador 3 mostrou um desempenho bastante regular durante todas as 20 partidas, fruto da utilização de uma estratégia de busca sistemática denominada de Estratégia Ótima (seção 4.3) que garantiu a descoberta das quatro cores pertencentes ao código secreto em no máximo 6 lances. Desta forma foi classificado como especialista.

A tabela 6.2 apresenta um resumo das vinte partidas disputadas pelos três jogadores. Durante as partidas os lances dos jogadores foram avaliados e, em algumas situações, os jogadores foram questionados sobre o raciocínio utilizado na definição do lance. Essas informações serviram como base para a construção do conjunto de procedimentos descritos tabela 5.7 e utilizados nos ensaios III-a e III-b.

É importante observar, que a despeito da classificação atribuída aos jogadores, os resultados apresentados na tabela 6.2, mostram que o melhor desempenho médio é obtido pelo jogador 2. Além disso, em um torneio entre os três jogadores o jogador 2 seria declarado campeão por haver utilizado o menor número de lances nas 20 partidas, 116 lances contra 129 do jogador 3. Esse resultado evidencia o caráter combinatório do problema proposto e mostra que a estratégia de realizar um lance inicial ao acaso, utilizada pelo jogador 2, embora não seja sempre eficiente e exija maior atenção e raciocínio dedutivo na realização dos lances, pode ser adequada em alguns casos. A questão é a da sorte na escolha da primeira combinação, o que impacta no desempenho médio do jogador. Há que se ressaltar, porém, que o número de partidas foi pequeno e comprometendo análises estatísticas. Acredita-se que com um número muito maior de partidas as medidas teriam sido outras, evidenciando de fato a qualidade de cada um dos três jogadores.

As partidas foram disputadas no modo 4x6, não havendo repetição de cores no código secreto. Essa simplificação teve por objetivo facilitar a resolução do problema pelos três jogadores, permitindo uma melhor comparação dos resultados. Entretanto, em um torneio disputado na modalidade 5x8 e com repetição de cores, é possível especular que a estratégia sistemática de solução do jogador 3 levaria vantagem. Observa-se ainda, que seria possível melhorar a base de comparação entre o desempenho dos três jogadores eliminando a influência do fator sorte no primeiro lance, o que poderia ser feito exigindo que todos os jogadores iniciassem as

partidas executando o mesmo lance. Entretanto, essa atitude teria um efeito indesejado de tolher o surgimento de novas estratégias, por isso, esse procedimento não foi adotado.

Tabela 6.2 – Desempenho dos jogadores em 20 partidas sucessivas.

	Jogador 1	Jogador 2	Jogador 3
	Iniciante	Intermediário	Especialista
Número Médio de Lances	7,50	5,80	6,15
Desvio Padrão	3,72	2,63	0,74
Máximo	18	12	7
Mínimo	3	2	5
Total de lances em 20 partidas	151	116	129

6.2 ENSAIO II – ALGORITMO GENÉTICO

O ensaio II consiste do estudo da solução do problema proposto a partir de uma busca combinatória guiada por um algoritmo genético. Conforme descrito na tabela 5.2, são propostas três versões para o algoritmo genético (AG1, AG2 e AG3) que procuram gradativamente aproximar-se da solução apresentada pelo jogador humano. Os resultados obtidos no ensaio II foram importantes no desenvolvimento do modelo conceitual utilizado nos ensaios III-a, III-b e IV.

6.2.1 AG1 (algoritmo genético – busca solução) – ANÁLISE DOS RESULTADOS

A primeira versão do algoritmo genético denominada de AG1 procura a solução para o jogo *Mastermind*, recombina indivíduos de uma população de potenciais soluções. Cada indivíduo representa um único lance (seqüência de cores) e candidato a solução do problema. O objetivo é encontrar uma solução através da simples recombinação de cores. Nesta versão não há interação com os resultados do jogador humano e o AG1 procura a solução final sem se importar com os passos executados entre o estado inicial e o estado final.

Quanto ao desempenho apresentado pelo AG1, pode-se concluir:

- Nas 100 simulações executadas o desempenho médio do AG1 ficou em 5,03 lances por partida. Este desempenho não permite comparações diretas com outros trabalhos similares uma vez que os parâmetros utilizados nas simulações não são compatíveis. Além disso, conforme discutido no item 5.5.8, algumas heurísticas importantes implementadas em outros trabalhos com o objetivo de aumentar a eficiência do algoritmo genético não foram aqui contempladas [MERELLO1999; BENTO1999]. As tabelas 6.3 e 6.4 mostram a título de informação os parâmetros de simulação e os resultados dos trabalhos citados.

Tabela 6.3 – Parâmetros e resultados de simulação para o *Mastermind* (4x6).

Mastermind versão Clássica (4x6)			
Autor	Lances Médios	Número de Partidas	Tamanho da População
Canto e Netto	5,03	100	300
Merelo [MERELO1999]	4,30	1000	300

Tabela 6.4 – Parâmetros e resultados de simulação para o *Mastermind* (5x8).

Mastermind versão Super (5x8)			
Autor	Lances Médios	Número de Partidas	Tamanho da População
Canto e Netto	13,1	30	200
Bento [BENTO1999]	6,8	500	300
Merelo [MERELO1999]	5,9	500	400

Note que o algoritmo proposto tem como objetivo encontrar um procedimento que seja próximo àquele desenvolvido por um jogador humano, e não simplesmente encontrar a melhor estratégia para resolver o jogo a partir de modelos de inteligência artificial e computação evolutiva. Desta forma, como dito, os resultados desta seção servem mais para ilustrar e como embasamento para a discussão das duas outras variantes (AG2 e AG3).

6.2.2 AG2 (algoritmo genético – busca reproduzir lances do humano) – ANÁLISE DOS RESULTADOS

A segunda versão do algoritmo genético, denominada de AG2, apresenta uma primeira tentativa de construir um algoritmo que reproduza os lances executados pelo jogador humano. Neste modelo, procura-se apenas reproduzir cada um dos lances do jogador humano através de uma busca combinatória guiada por um algoritmo genético. O AG2 é executado para cada um dos lances do jogador humano até que um dos critérios de parada seja atingido (encontrar a seqüência de cores usada pelo jogador no mesmo lance, ou atingir o número máximo de 50 gerações). A figura 6.7 apresenta o resultado da execução de uma partida. Foram executadas 100 partidas, sendo necessárias em média de 5,4 gerações por lance para que o algoritmo encontre uma seqüência de cores igual à escolhida pelo jogador humano a cada lance.

Figura 6.7 – Resultado da execução do algoritmo genético versão AG2.

6.2.3 AG3 (algoritmo genético – busca obter mesma avaliação do humano) – ANÁLISE DOS RESULTADOS

A terceira versão do algoritmo genético (AG3) é uma variação do algoritmo proposto na seção anterior. Nesta versão, o AG3 busca conseguir obter o resultado de cada um dos lances do jogador humano, ou seja, o número de pinos brancos e pretos obtidos pelo jogador humano em cada lance da partida. A figura 6.8 mostra que os lances gerados pelo AG3 não são necessariamente iguais (mesma sequência de cores) aos propostos pelo jogador humano, isto porque, existem diferentes combinações de cores que podem obter o mesmo resultado. Para o algoritmo AG3, foram executadas 100 partidas, sendo necessárias em média 10,6 gerações para que se encontrar um lance com pontuação equivalente ao do jogador humano. Na próxima seção (ensaio III-a), é proposta uma nova abordagem para o algoritmo genético denominada de Classificador Genético (CG), nesta abordagem os indivíduos da população são substituídos por um conjunto de operadores baseados em procedimentos gerais extraídos a partir da observação dos lances realizados durante as partidas disputadas pelos jogadores humanos no ensaio I (tabela 5.7).

Figura 6.8 – Resultado da execução do algoritmo genético versão AG3.

6.3 ENSAIO III-a – CLASSIFICADOR GENÉTICO (operadores sem memória)

O Classificador Genético (CG) descrito nas seções seguintes tem como base o algoritmo genético descrito no ensaio II. Nessa versão do ensaio, o conjunto de candidatos à solução do problema foi substituído por um conjunto de operadores simples (sem memória), isto é, os operadores não avaliam lances passados antes de executar o seu procedimento. O ensaio consiste na tentativa de gerar, através da combinação genética, um conjunto de operadores que, quando combinados e executados sequencialmente, devem ser capazes de encontrar a solução do problema. Neste caso o indivíduo representa toda uma sequência de lances para concluir a partida. A seguir, são discutidas as três versões de algoritmo para o classificador genético (CG1S, CG2S e CG3S).

6.3.1 CG1S (operadores sem memória – busca solução) – ANÁLISE DOS RESULTADOS

A primeira versão do classificador genético denominado CG1S, cujos detalhes foram descritos na seção 5.3 procura, orientada por um AG, uma combinação de operadores sem memória (extraídos da tabela 5.7) capaz de encontrar a solução para o problema em questão. Com relação às simulações executadas com o CG1 seguem algumas considerações e conclusões:

- Durante os ensaios realizados observou-se que o resultado de uma partida depende fortemente do lance inicial. Por essa razão, para permitir condições iguais de comparação entre diferentes procedimentos gerados pelo CG1S, fixou-se que todos os indivíduos da população realizam sempre o mesmo lance inicial. Observação: essa regra é válida também para os algoritmos CG2S e CG3S, mas nesses casos, o lance inicial é sempre igual ao lance inicial do jogador humano.
- Todos os indivíduos são formados por 12 genes (12 operadores simples). Esse número é baseado no número máximo de lances utilizado pelo jogador 2 (tabela 6.2). Acima de 12 lances acredita-se que o procedimento gerado pelo classificador genético tenha pouco significado e a solução, caso seja encontrada, estará mais fortemente associada ao acaso. Observação: no

caso dos algoritmos CG2S e CG3S o número de genes dos indivíduos é igual ao número de lances utilizado pelo jogador humano para solucionar o problema na referida partida.

- Foram realizados cinco conjuntos de simulações e os resultados mostraram que a combinação de operadores simples, na maioria das vezes, não convergiu para uma seqüência de operadores capaz de solucionar o problema nas diferentes partidas. Outro resultado observado é que mesmo quando houve a convergência do algoritmo para uma solução do problema, a seqüência de operadores encontrada não foi capaz de solucionar o mesmo problema em uma segunda tentativa. Ou seja, não se encontrou um conjunto de operadores considerado suficientemente genérico para resolver qualquer partida, mas sim um conjunto específico para cada caso.
- A tabela 6.5 descreve os parâmetros e resultados dos cinco conjuntos de simulações executadas com o CG1S e a tabela 6.6 os detalhes do procedimento obtido na simulação número 3. As simulações foram executadas com os mesmos parâmetros de configuração, entretanto, a simulação número 3 convergiu para uma solução na 11ª geração enquanto as outras quatro tentativas não convergiram durante as cem primeiras gerações (coluna 3 – tabela 6.5). Outro dado importante é que a seqüência de operadores extraída da simulação número 3 foi avaliada com o mesmo código em cem partidas sendo capaz de solucionar o problema em apenas seis casos. A tabela 6.5 também mostra o número de vezes em cem partidas que a melhor seqüência de operadores em cada simulação foi capaz de descobrir as cores presentes no código, mas não foi capaz de encontrar a ordem correta (coluna 5 – tabela 6.8).

Tabela 6.5 – Parâmetros e resultados de simulação com o CG1.

1	2	3	4	5	6
nº	Nº de operadores	O algoritmo CG2S encontrou algum conjunto de operadores que <u>soluciona o problema</u> ?	Seqüência de operadores que representa o melhor indivíduo após 100 gerações	Porcentagem de soluções com cores corretas em 100 partidas com o mesmo código secreto	Porcentagem de soluções com cor e posição corretas em 100 partidas com o mesmo código secreto
1	12	não	1,c,b,j,a,i,j,c,b,j,a,i	4	0
2	12	não	1,i,b,i,j,c,e,j,i,b,g,g	6	0
3	12	sim	1,b,g,c,c,d,a,a,c,g,d,g	5	6
4	12	não	1,d,b,i,j,c,e,j,i,c,g,g	6	0
5	12	não	1,h,j,c,c,j,c,e,i,h,g,j	2	0

Tabela 6.6 – Descrição da seqüência de operadores gerados pelo CG1S na simulação número 1.

Número do lance	operadores	Descrição dos Operadores
1	1, 2, 3, 4	Lance inicial (igual para todas as simulações)
2	<i>b</i>	Gera um lance com uma única cor escolhida ao acaso.
3	<i>g</i>	Repete o último lance trocando <i>b</i> cores de posição.
4	<i>c</i>	Gera um lance com uma cor qualquer repetida <i>k</i> vezes (<i>k</i> é um número aleatório).
5	<i>c</i>	Gera um lance com uma cor qualquer repetida <i>k</i> vezes (<i>k</i> é um número aleatório).
6	<i>d</i>	Gera um lance com uma cor qualquer repetida $(n - p - b)$ vezes.
7	<i>a</i>	Gera um lance com <i>n</i> escolhidas ao acaso.
8	<i>a</i>	Gera um lance com <i>n</i> escolhidas ao acaso.
9	<i>c</i>	Gera um lance com uma cor qualquer repetida <i>k</i> vezes (<i>k</i> é um número aleatório).
10	<i>g</i>	Repete o último lance trocando <i>b</i> cores de posição.
11	<i>d</i>	Gera um lance com uma cor qualquer repetida $(n - p - b)$ vezes.
12	<i>g</i>	Repete o último lance trocando <i>b</i> cores de posição.

A análise do único conjunto de operadores que emergiu do algoritmo CG1S e que foi capaz de solucionar o problema (simulação número 3 tabela 6.6), não permite identificar nenhuma seqüência de procedimentos que caracterize uma estratégia de solução. Este fato corrobora com o resultado já descrito de que este conjunto de operadores, quando testado com o mesmo código secreto, obteve

desempenho pouco expressivo e sugere que a descoberta da solução ocorreu em decorrência do acaso.

Por outro lado, sugere que como decorrência de um procedimento evolutivo, tal indivíduo teve certo sucesso (na medida em que seu desempenho está muito além do desempenho de um indivíduo que tenha contado apenas com a sorte) tendo assim um destaque que o diferencia estatisticamente de casos simplesmente aleatórios. Mas este indivíduo representa apenas o surgimento de um indivíduo capaz de resolver o problema (numa proporção maior que a da simples aleatoriedade), ainda que não represente (por aproximação) um jogador humano. Estes casos serão tratados nas próximas sub-seções.

6.3.2 CG2S (operadores sem memória – busca reproduzir lances do humano) – ANÁLISE DOS RESULTADOS

A segunda versão do classificador genético, denominada de CG2S, é uma primeira tentativa de encontrar, através da recombinação genética, uma sequência de operadores que reproduza os lances executados pelo jogador humano. O CG2S procura por um conjunto de operadores capaz de reproduzir cada um dos lances do jogador humano (mesma sequência de cores). O CG2S foi utilizado como primeira aproximação para se tentar chegar a um conjunto de operadores que reproduza o raciocínio desenvolvido pelo jogador humano. Nas simulações apresentadas foram utilizadas populações de 200 indivíduos e o jogador humano resolveu a partida em 5 lances (tabela 6.7).

Os resultados descritos na tabela 6.7 mostram que, assim como no ensaio anterior, não foi possível identificar nenhuma sequência de operadores que configure algum tipo de estratégia para a solução do problema. Cabe ressaltar que embora não tenham sido encontradas soluções idênticas às do jogador humano (propósito deste ensaio), alguns dos melhores indivíduos após 100 gerações foram capazes de resolver algumas partidas em 5 lances.

Tabela 6.7 – Parâmetros e resultados de simulação com o CG2S.

1	2	3	4	5	6
nº	Nº de operadores	O algoritmo CG2S encontrou algum conjunto de operadores que <u>reproduz os lances</u> do Jogador humano?	Seqüência de operadores que representa o melhor indivíduo após 100 gerações	Porcentagem de soluções com cores corretas em 100 partidas com o mesmo código secreto	Porcentagem de soluções com cor e posição corretas em 100 partidas com o mesmo código secreto
1	5	não	1,c,g,h,g	4	1
2	5	não	1,g,b,e,j	6	0
3	5	não	1,c,f,c,a	8	2
4	5	não	1,c,b,a,e	10	3
5	5	não	1,i,b,a,j	5	0

6.3.3 CG3S (operadores sem memória – busca obter mesma avaliação do humano) – ANÁLISE DOS RESULTADOS

O algoritmo CG3S tem como objetivo encontrar uma seqüência de operadores que reproduza os resultados obtidos em cada um dos lances executados pelo jogador humano (mesmo número de pinos pretos e brancos). Foram realizados cinco conjuntos de simulações com o jogador utilizando duas estratégias diferentes para resolver o problema. Nas duas primeiras simulações o jogador humano utilizou a estratégia denominada ótima (seção 4.4) e resolveu as partidas respectivamente em seis e sete lances. Nessa estratégia os lances que são sabidamente incorretos são jogados, com o objetivo de extrair informação sobre o código escondido e reduzir o espaço de busca. Nas outras três simulações o jogador humano não utilizou nenhuma estratégia ótima, apenas iniciou a partida com lances aleatórios e procurou realizar os lances seguintes com base na análise do número de pinos pretos e brancos recebidos como *feedback*. Nesse caso o jogador utilizou respectivamente cinco, seis e cinco lances para resolver as três últimas partidas. A tabela 6.8 resume os resultados e parâmetros das cinco simulações.

Tabela 6.8 – Parâmetros e resultados de simulação com o CG3S.

1	2	3	4	5	6
nº	Nº de operadores	O algoritmo CG3S encontrou algum conjunto de operadores que <u>reproduz os ganhos</u> do Jogador humano?	Seqüência de operadores que representa o melhor indivíduo após 100 gerações	Soluções com cores corretas em 100 partidas com o mesmo código secreto	Soluções com cor e posição corretas em 100 partidas com o mesmo código secreto
1	6	não	1, j, d, e, a, f	3	0
2	7	não	1, b, i, e, e, i, j, e	8	2
3	5	não	1, j, g, f, e	6	0
4	6	não	1, j, g, e, a, c	4	1
5	5	não	1, d, j, h, f	2	0

As simulações com o CG3S mostraram que, após 100 gerações, o algoritmo não converge para nenhum conjunto de operadores capaz de reproduzir os ganhos do jogador humano na sua totalidade (coluna 3 – tabela 6.8). Os conjuntos de operadores obtidos após a execução do algoritmo foram capazes apenas de solucionar o problema em poucas partidas (coluna 6 – tabela 6.8). Entretanto, uma análise individualizada dos lances realizados durante uma partida, permite identificar algum grau de similaridade com o procedimento do jogador humano como, por exemplo, realizar lances com cores repetidas (operador *b*, tabela 5.7) e manter as cores corretas no próximo lance (operador *i* e *j*). Esses resultados são destacados na figura 6.9 que ilustra a solução em sete lances apresentada por um jogador humano (lado direito da figura 6.9) e o procedimento encontrado pelo algoritmo CG3S após 100 gerações com uma população de 200 indivíduos (lado esquerdo da figura 6.9).

Desta forma pode-se dizer que ainda que de forma parcial o procedimento conseguiu aproximar-se de seu propósito, ou seja de obter as mesmas notas obtidas pelo jogador humano. Mas a proporção com que isto ocorreu não nos permite validar estatisticamente este procedimento, demonstrando que novas considerações devam ser feitas a respeito do procedimento para melhor adequá-lo ao seu propósito, ou seja copiar o procedimento usado pelo jogador humano.

Figura 6.9 - Procedimento obtida depois de 100 gerações pelo CG3S (jogador virtual número 2 da tabela 6.8).

A tabela 6.9 apresenta um resumo da análise dos lances do jogador humano e virtual a partir dos operadores descritos na tabela 5.7. A tabela mostra o resultado da simulação contendo a seqüência de operadores utilizada e a pontuação alcançada pelos jogadores humano e virtual. Com a análise dos lances e a descrição verbal do raciocínio utilizado pelo jogador humano é possível observar que o mesmo faz uso de sua memória para avaliar pelo menos o último lance antes de executar o próximo, enquanto o jogador virtual não o faz (na versão implementada o conjunto de operadores que compõe o jogador virtual não possui nenhum sistema de memória que permita avaliar se o último lance trouxe ou não ganhos quando comparado ao seu antecessor).

Tabela 6.9 - Análise da similaridade entre os lances do jogador humano e virtual.

Lance	Jogador Virtual (CG3S)		Jogador Humano	
	Pontuação (P, B)	operador	Pontuação (P, B)	operador
0	(1,0)	<i>b</i>	(1,0)	<i>b</i>
1	(2,0)	<i>i</i>	(2,0)	<i>i</i>
2	(2,0)	<i>e</i>	(1,2)	<i>i</i>
3	(2,0)	<i>e</i>	(0,3)	<i>i ou j</i>
4	(0,2)	<i>i</i>	(1,3)	<i>i ou j</i>
5	(1,2)	<i>j</i>	(2,2)	<i>g</i>
6	(2,2)	<i>e</i>	(4,0)	<i>g</i>

A partir dos resultados das três simulações descritas, pode-se elencar algumas hipóteses e conclusões:

- Embora o procedimento executado pelo conjunto de operadores, obtido através do pelo classificador genético, possua elementos encontrados no procedimento utilizado pelo jogador humano, esses elementos são desconexos e o procedimento não conduz sempre o jogador virtual à solução do jogo, mesmo quando o procedimento gerado é aplicado repetidamente ao mesmo código secreto na tentativa de solucioná-lo.
- A análise dos resultados permite levantar a hipótese que a geração de procedimentos gerais para o *Mastermind* não pode ser construída sem um mecanismo de memória que considere os lances passados na descoberta do código secreto. Em outras palavras, o sucesso do próximo lance depende fortemente da análise dos lances anteriores.
- O conjunto de operadores descrito na tabela 5.7 descreva as possibilidades que o jogador humano possui de realizar um lance, inclusive com os graus de liberdade existentes na escolhas das cores, entretanto, esse grau de liberdade, descritos nos operadores como mecanismos aleatórios de escolha de cores acaba por ser responsável pela impossibilidade de convergência dos algoritmos, uma vez que o jogador humano ao escolher uma cor entre várias disponíveis, não o faz de forma aleatória, mas sim

utilizando-se de mecanismos cognitivos complexos que combinam memória e avaliação de lances futuros.

As próximas seções apresentam dois novos ensaios, o objetivo é testar um mecanismo de memória simples e sua influência sobre o comportamento do algoritmo proposto (ensaio III-b) e o ensaio IV apresenta uma nova versão do classificador genético que procura solucionar o problema da aleatoriedade presentes nos operadores utilizados.

6.4 ENSAIO III-b – CLASSIFICADOR GENÉTICO (operadores com memória)

Com base nos resultados já discutidos no ensaio anterior, propõe-se uma nova versão para o classificador genético, sendo o conjunto de operadores simples (sem memória), que constituíam a população inicial na versão III-a, substituído por um conjunto de operadores com memória. Os operadores com memória verificam se o último lance obteve ganhos quando comparado com o seu antecessor, caso contrário, utiliza-se o antecessor para gerar um novo lance. Esse mecanismo de memória evita que um resultado ruim, ocorrido devido à aplicação de um determinado operador, se propague para o próximo lance. Com isso, simulasse o comportamento de jogadores humanos classificados como iniciantes. Um jogador iniciante ao obter em um lance uma pontuação inferior à obtida no lance anterior, descarta este lance e retorna ao lance anterior para, a partir dele, realizar uma nova análise e executar o lance seguinte. Com isso, o jogador iniciante despreza importantes informações contidas naquele lance. Já um jogador especialista utiliza-se de uma estratégia mais complexa procurando extrair do resultado ruim informações que serão combinadas com os resultados anteriores para gerar o novo lance.

Essa abordagem tem como objetivo avaliar a importância da memória no processo de decisão e verificar se apenas a introdução de um mecanismo de memória simples, como o que está sendo proposto, será suficiente para permitir uma melhoria no desempenho do algoritmo proposto. Os resultados das três versões do algoritmo (tabela 5.4) são apresentados a seguir.

6.4.1 CG1M (operadores com memória – busca solução) – Análise dos Resultados

A seguir são apresentados os resultados dos cinco conjuntos de simulações executados com a versão do classificador genético denominado CG1M. A versão em questão procura um conjunto de operadores (extraídos da lista da tabela 5.7) capaz de encontrar a solução para o *Mastermind*. O diferencial em relação à versão correspondente do ensaio III-a é a presença do mecanismo de memória já discutido. Nessa versão o algoritmo trabalha independentemente da solução encontrada pelo jogador humano. As simulações foram realizadas utilizando-se populações de duzentos indivíduos, cada indivíduo representando um conjunto de doze operadores e executando o mesmo lance inicial. Em cada simulação o algoritmo evolui até encontrar um indivíduo que solucione o problema. A tabela 6.10 apresenta a geração em que a solução foi encontrada, o conjunto de operadores e o resultado da aplicação desse conjunto de operadores em cem novas partidas com o mesmo código secreto.

Tabela 6.10 - Parâmetros e resultados da simulação do algoritmo CG1M.

1	2	3	4	5	6
nº	Nº de operadores	O algoritmo encontrou um conjunto de operadores que <u>soluciona o problema?</u>	Sequência de operadores encontrada para solução do problema	Soluções com cores corretas em 100 partidas com o mesmo código secreto	Soluções com cor e posição corretas em 100 partidas com o mesmo código secreto
1	12	sim	1,i,b,i,j,c, e,j,i,b,g,g	27	6
2	12	sim	1,i,b,h,j,e, h,e,i,h,a,i	35	8
3	12	sim	1,i,b,h,j,e, h,e,i,j,a,i	31	7
4	12	sim	1,i,h,b,j,e, a,e,i,d,f,i	14	4
5	12	sim	1,i,h,b,j,e, a,e,i,d,g,i	21	6

Os resultados demonstram que o conjunto de operadores encontrado não caracteriza uma solução geral, entretanto, pode-se observar que o uso do mecanismo de memória permitiu encontrar pelo menos uma solução em cada uma das cinco simulações (tabela 6.10 coluna 3), além disso, houve um aumento significativo do número de partidas solucionadas com o conjunto de operadores encontrado pelo algoritmo, quando comparado com o algoritmo correspondente do ensaio III-a (sem memória). A figura 6.11 apresenta a comparação entre os resultados obtidos pelo melhor conjunto de operadores em cada um dos algoritmos. Nesse caso, o algoritmo dotado de mecanismo de memória obteve um desempenho quatro vezes melhor que o seu correspondente sem memória, quando se observa o número de partidas solucionadas, e um desempenho sete vezes melhor quando se compara o número de partidas em que o algoritmo conseguiu encontrar somente as cores corretas. Há, portanto aqui uma evidência da importância do mecanismo de memória para resolver este problema, corroborando com a expectativa de que a capacidade de memória é necessária para poder resolver problemas desta natureza.

Figura 6.10 - Comparação entre os resultados dos algoritmos CG1S (ensaio III-a) e CG1M (ensaio III-b) obtidos em 100 partidas com o mesmo código secreto.

6.4.2 CG2M (operadores com memória – busca reproduzir lances do humano) –

Análise dos Resultados

A tabela 6.11 compila os resultados das simulações realizadas com o algoritmo CG2M, o objetivo é encontrar uma seqüência de operadores (com memória) que reproduza fielmente os lances do jogador humano (mesma cor). O algoritmo foi executado com uma população de duzentos indivíduos, cada indivíduo contendo sete operadores (número de lances utilizado pelo jogador humano para solucionar o problema).

Tabela 6.11 – Parâmetros e resultados da simulação CG2M.

1	2	3	4	5	6	7
nº	Nº de operadores	O algoritmo encontrou algum conjunto de operadores que <u>reproduz os ganhos</u> do Jogador humano?	O algoritmo encontrou algum conjunto de operadores que <u>soluciona o problema?</u>	Seqüência de operadores	Soluções com cores corretas em 100 partidas com o mesmo código secreto	Soluções com cor e posição corretas em 100 partidas com o mesmo código secreto
1	5	não	sim	1, f, c, a, i	9	2
2	5	não	sim	1, f, a, a, h	5	1
3	5	não	sim	1, b, a, e, a	8	2
4	5	não	sim	1, b, a, j, a	11	3
5	5	não	sim	1, g, c, j, a	8	3

Os resultados mostram que não houve convergência do algoritmo para nenhum conjunto capaz de reproduzir os lances do jogador humano (tabela 6.11 coluna 3). Mesmo não conseguindo encontrar um conjunto que reproduza os lances do jogador humano o algoritmo foi capaz de encontrar pelo menos uma seqüência que resolva o problema (tabela 6.11 colunas 4 e 5). Esse conjunto de operadores foi avaliado em cem novas partidas com um mesmo código para testar a sua generalidade, o número de partidas em que houve apenas a descoberta das cores do código e o número de partidas resolvidas (cor e posição) são apresentadas na

tabela 6.11 (colunas 6 e 7) e evidenciam o problema da incapacidade do conjunto de operadores de generalizar uma solução mesmo quando o conjunto de operadores é testado com um mesmo código.

O próximo ensaio verifica a possibilidade de encontrar um conjunto de operadores que reproduza apenas os ganhos (pinos pretos e brancos) obtidos pelo jogador humano.

6.4.3 CG3M (operadores com memória – busca obter mesma avaliação do humano) – Análise dos Resultados

A tabela 6.12 apresenta os resultados obtidos com a versão CG3M. O objetivo desse algoritmo é encontrar um conjunto de operadores com memória que reproduza os ganhos do jogador humano, isto é, reproduza a seqüência de pinos pretos e brancos obtida pelo jogador humano. O algoritmo utiliza os mesmos parâmetros de simulação utilizados na versão CG1M e CG2M.

Tabela 6.12 – Parâmetros e resultados da simulação do algoritmo CG3M.

1	2	3	4	5	6	7
nº	Nº de operadores	O algoritmo encontrou algum conjunto de operadores que <u>reproduz os ganhos</u> do Jogador humano?	O algoritmo encontrou algum conjunto de operadores que <u>soluciona o problema?</u>	Seqüência de operadores	Soluções com cores corretas em 100 partidas com o mesmo código secreto	Soluções com cor e posição corretas em 100 partidas com o mesmo código secreto
1	5	não	sim	1,a,h,h,i	12	2
2	5	não	sim	1,f,f,i,h	6	1
3	5	não	sim	1,b,b,e,a	8	3
4	5	não	sim	1,a,b,a,j	10	2
5	5	não	sim	1,g,a,j,b	7	1

A avaliação dos resultados da tabela 6.12 mostra que o algoritmo não convergiu para uma solução capaz de reproduzir os ganhos do jogador humano na sua totalidade, além disso, o conjunto de operadores encontrado não caracteriza

uma solução geral para o problema uma vez que, ao ser aplicado cem vezes a um mesmo código, foi capaz de encontrar a solução algumas poucas vezes (tabela 6.12 colunas 6 e 7). Entretanto, quando se compara os resultados apresentados nas tabelas 6.8 (CG3S sem memória) e 6.12 (CG3M com memória) observa-se uma melhora no desempenho do conjunto de operadores dotado do mecanismo de memória, o que comprova a influência da memória no processo de solução do problema em questão. A figura 6.11 apresenta a comparação entre os resultados obtidos pelo melhor conjunto de operadores em cada um dos algoritmos utilizando o mesmo número de lances (ver tabelas 6.8 e 6.12).

Figura 6.11 - Comparação entre os resultados dos algoritmos CG3S (ensaio III-a) e CG3M (ensaio III-b) obtidos em 100 partidas com o mesmo código secreto e o mesmo número de lances.

Com base nos resultados apresentados nos ensaios III-a e III-b, podem-se extrair três conclusões importantes:

- A introdução do mecanismo simples de memória auxilia para que algumas soluções específicas sejam encontradas pelo algoritmo.
- Os operadores gerais descritos por humanos (tabela 5.7) possuem um grau de aleatoriedade que impede que o algoritmo convirja para uma solução geral. Essa aleatoriedade é expressa quando se seleciona um operador, como por exemplo, o operador *i* da tabela 5.7 (*Repete o ultimo lance trocando b cores de posição e substituindo $(n - p - b)$ cores por uma única cor*

escolhida ao acaso), sendo respectivamente: n o número de cores do código, p número de pinos pretos e b número de pinos brancos. A figura 6.11 exibe as possibilidades de lances gerados a partir da aplicação do operador i a um lance n qualquer.

Figura 6.12 – Lances gerados a partir da aplicação do operador i , o desempenho de cada lance aparece ao lado dos números representado o lance.

A análise da figura 6.12 permite concluir que qualquer um dos seis lances gerados a partir do operador i são equiprováveis, mas apenas um corresponde ao código procurado. Observe que o mecanismo de memória implementado verifica se o lance n trouxe vantagens com relação ao lance $n-1$ e não faz nenhuma verificação com relação ao lance $n+1$.

- O jogador humano faz uso de funções cognitivas complexas que englobam mecanismos de memória, raciocínio dedutivo e análise de lances futuros, permitindo a redução do número de lances disponíveis para a escolha. As figuras 6.13 e 6.14 exemplificam esse raciocínio. Observe que o jogador humano realiza o mesmo lance n descrito na figura 6.12, mas de posse do lance $n-1$ e de seu resultado é capaz de concluir que ao trocar a cor

6 presente no lance $n-1$ pela cor 3 no lance n , não houve alteração no número de cores corretas, pois tanto no lance $n-1$ como no lance n há três cores corretas. Em outras palavras, a cor 3 e a cor 6 fazem parte do código secreto. Analisando-se as combinações possíveis para o lance $n+1$ o jogador humano elimina quinze dos vinte e quatro possíveis lances, pois não possuem as cores 3 e 6 (figura 6.13).

Figura 6.13 – Representação da árvore de decisão do jogador humano (1).

Agora, analisando a pontuação dos lances $n-1$ (0,3), o jogador humano conclui que o código procurado não pode repetir as posições das cores do lance $n-1$, isto restringe seu espaço de busca de nove lances possíveis para apenas cinco (figura 6.14).

Figura 6.14 – Representação da árvore de decisão do jogador humano (2).

O próximo ensaio apresenta uma nova abordagem que utiliza um conjunto de operadores instanciados e tem como objetivo eliminar a aleatoriedade dos operadores descrita pela tabela 5.7 e avaliar o desempenho dessa nova abordagem sobre o efeito do algoritmo proposto.

6.5 ENSAIO IV - CLASSIFICADOR GENÉTICO (conjunto de operadores instanciados)

A partir das conclusões dos ensaios III-a e III-b, apresenta-se uma nova versão para o classificador genético. Essa versão substitui o conjunto de operadores descrito na tabela 5.7 por um conjunto de operadores instanciados que descrevem todas as possibilidades de jogadas para um determinado lance a partir da análise da pontuação obtida. Para que o número de operadores utilizados não se tornasse demasiadamente grande, reduziu-se a dimensão do problema, nesse ensaio foram

utilizados combinações de três cores extraídas de um alfabeto com 5 cores diferentes (3x5). O objetivo é mostrar que, eliminando-se a aleatoriedade presente nos operadores utilizados nos ensaios III-a e III-b, o algoritmo evolutivo proposto será capaz de convergir para uma solução satisfatória nas três versões até aqui estudadas. Os operadores utilizados nesse ensaio foram instanciados a partir da seguinte análise:

Seja: NP é o número de pinos pretos (cor correta na posição correta)

NB é o número de pinos brancos (cor correta na posição incorreta)

As possíveis combinações para as variáveis NP e NB em uma partida (3x5) são apresentados na tabela 6.13:

Tabela 6.13 – Análise da pontuação possível de ser obtida em uma partida.

	Condição	Descrição da Condição
1	$NP=0$ e $NB=0$	Nenhuma cor correta.
2	$NP=0$ e $NB=1$	1 cor na posição incorreta.
3	$NP=0$ e $NB=2$	2 cores na posição incorreta.
4	$NP=0$ e $NB=3$	3 cores na posição incorreta.
5	$NP=1$ e $NB=0$	1 cor na posição correta.
6	$NP=1$ e $NB=1$	1 cor e posição correta e 1 cor na posição incorreta.
7	$NP=1$ e $NB=2$	1 cor e posição correta e 2 cores na posição incorreta.
8	$NP=2$ e $NB=0$	2 cores na posição correta.
9	$NP=3$ e $NB=0$	3 cores na posição correta.

A partir das condições descritas na tabela 6.13 foram instanciados oito grupos de operadores ($\alpha, \beta, \chi, \delta, \epsilon, \phi, \gamma, \eta$) apresentados na tabela 6.14. Cada grupo contém uma ou mais operadores que cobrem todas as possibilidades viáveis para um determinado lance.

Tabela 6.14 - Conjunto de operadores instanciados.

	Condição	Operador	Descrição do Operador
1	$NP=0$ e $NB=0$	α_1	Troca todas as cores por cores ainda não utilizadas.
2	$NP=0$ e $NB=1$	β_2	Troca à posição da primeira cor com a segunda. Substitui a primeira e a terceira cor por novas cores (ainda não utilizadas).
3	$NP=0$ e $NB=1$	β_3	Troca à posição da primeira cor com a terceira posição substitui a primeira e a segunda cor por novas cores
4	$NP=0$ e	β_4	Troca à posição da segunda cor com a primeira

	Condição	Operador	Descrição do Operador
	NB=1		posição. Substitui a segunda e a terceira cor por novas cores
5	NP=0 e NB=1	β_5	Troca à posição da segunda cor com a terceira posição. Substitui a primeira e a segunda cor por novas cores.
6	NP=0 e NB=1	β_6	Troca à posição da terceira cor com a primeira posição. Substitui a segunda e a terceira cor por novas cores.
7	NP=0 e NB=1	β_7	Troca à posição da terceira cor com a segunda posição. Substitui a primeira e a terceira cor por novas cores.
8	NP=0 e NB=2	χ_8	Troca a primeira com a segunda cor de posição e substitui a terceira pela primeira cor disponível.
9	NP=0 e NB=2	χ_9	Troca a primeira com a segunda cor de posição e substitui a terceira pela segunda cor disponível.
10	NP=0 e NB=2	χ_{10}	Troca a primeira com a terceira cor de posição e substitui a segunda pela primeira cor disponível.
11	NP=0 e NB=2	χ_{11}	Troca a primeira com a terceira cor de posição e substitui a segunda pela segunda cor disponível.
12	NP=0 e NB=2	χ_{12}	Troca a segunda com a terceira cor de posição e substitui a primeira pela primeira cor disponível.
13	NP=0 e NB=2	χ_{13}	troca a primeira com a segunda cor de posição, a segunda com a terceira cor e substitui a terceira pela primeira cor disponível.
14	NP=0 e NB=2	χ_{14}	troca a primeira com a segunda cor de posição, a segunda com a terceira cor e substitui a terceira pela segunda cor disponível.
15	NP=0 e NB=2	χ_{15}	Troca a segunda com a terceira cor de posição e substitui a primeira pela segunda cor disponível.
16	NP=0 e NB=3	δ_{16}	Move a primeira para a segunda. Move a segunda para a terceira. Move a terceira para a primeira.
17	NP=0 e NB=3	δ_{17}	Move a primeira para a terceira. Move a segunda para a primeira. Move a terceira para a segunda.
18	NP=1 e NB=0	ε_{18}	Manter a primeira cor e trocar a segunda e a terceira.
19	NP=1 e NB=0	ε_{19}	Manter a segunda cor e trocar a primeira e a terceira.
20	NP=1 e NB=0	ε_{20}	Manter a terceira cor e trocar a primeira e a segunda.
21	NP=1 e NB=1	ϕ_{21}	Manter a primeira cor. Trocar a segunda cor pela primeira disponível. Trocar de posição a segunda com a terceira.
22	NP=1 e NB=1	ϕ_{22}	Manter a primeira cor. Trocar a segunda cor pela segunda disponível. Trocar de posição a segunda com a terceira.
23	NP=1 e NB=1	ϕ_{23}	Manter a primeira cor. Trocar a terceira cor pela primeira disponível. Trocar de posição a segunda com a terceira.
24	NP=1 e NB=1	ϕ_{24}	Manter a primeira cor. Trocar a terceira cor pela segunda disponível. Trocar de posição a segunda com a terceira.
25	NP=1 e NB=1	ϕ_{25}	Manter a segunda cor. Trocar a primeira cor pela primeira disponível. Trocar de posição a primeira com a terceira.

	Condição	Operador	Descrição do Operador
26	NP=1 e NB=1	$\phi 26$	Manter a segunda cor. Trocar a primeira cor pela segunda disponível. Trocar de posição a primeira com a terceira.
27	NP=1 e NB=1	$\phi 27$	Manter a segunda cor trocar a terceira cor pela primeira disponível trocar de posição a primeira com a terceira
28	NP=1 e NB=1	$\phi 28$	Manter a segunda cor Trocar a terceira cor pela segunda disponível. Trocar de posição a primeira com a terceira.
29	NP=1 e NB=1	$\phi 29$	Manter a terceira cor trocar a primeira cor pela primeira disponível trocar de posição a primeira com a segunda
30	NP=1 e NB=1	$\phi 30$	Manter a terceira cor. Trocar a primeira cor pela segunda disponível. Trocar de posição a primeira com a segunda.
31	NP=1 e NB=1	$\phi 31$	Manter a terceira cor. Trocar a segunda cor pela primeira disponível. Trocar de posição a primeira com a segunda.
32	NP=1 e NB=1	$\phi 32$	Manter a terceira cor. Trocar a segunda cor pela segunda disponível. Trocar de posição a primeira com a segunda.
33	NP=1 e NB=2	$\gamma 33$	Trocar de posição a primeira com a segunda.
34	NP=1 e NB=2	$\gamma 34$	Trocar de posição a primeira com a terceira.
35	NP=1 e NB=2	$\gamma 35$	Trocar de posição a segunda com a terceira.
36	NP=2 e NB=0	$\eta 36$	Trocar a terceira cor pela primeira disponível.
37	NP=2 e NB=0	$\eta 37$	Trocar a terceira cor pela segunda disponível.
38	NP=2 e NB=0	$\eta 38$	Trocar a segunda cor pela primeira disponível.
39	NP=2 e NB=0	$\eta 39$	Trocar a segunda cor pela segunda disponível.
40	NP=2 e NB=0	$\eta 40$	Trocar a primeira cor pela primeira disponível
41	NP=2 e NB=0	$\eta 41$	Trocar a primeira cor pela segunda disponível

A seguir são analisados os resultados obtidos com a simulação dos algoritmos descritos na tabela 5.5.

6.5.1 CG1E (conjunto de operadores instanciados – busca solução) – Análise dos Resultados

Seguindo o procedimento já adotado nos ensaios anteriores, a primeira versão do algoritmo, aqui denominada de CG1E, busca um conjunto de operadores

extraído da tabela 6.14, capaz de solucionar o problema independentemente do procedimento do jogador humano. Nessa versão, procura-se identificar conjuntos capazes de solucionar o problema em exatamente cinco lances. O número de lances foi escolhido baseado no número médio de lances utilizado por um jogador humano. É importante ressaltar que com o modelo de operadores utilizado nesse ensaio é possível após o primeiro lance, se ele não for o código procurado, encontrar um único operador que aplicado ao lance inicial solucione o problema. A tabela 6.15 mostra os parâmetros de simulação e dez conjuntos de quatro operadores capazes de encontrar o código (4,1,5) a partir do lance inicial (1,2,3).

Tabela 6.15 – Parâmetros e resultados de simulação do algoritmo CG1E.

	Código Secreto	Nº de lances	Lance inicial	Conjunto de operadores que soluciona o problema			
1	4,1,5	5	1,2,3	$\beta 7$	$\beta 3$	$\phi 31$	$\phi 26$
2	4,1,5	5	1,2,3	$\beta 3$	$\gamma 33$	$\gamma 33$	$\gamma 35$
3	4,1,5	5	1,2,3	$\beta 3$	$\gamma 34$	$\gamma 34$	$\gamma 35$
4	4,1,5	5	1,2,3	$\beta 3$	$\chi 8$	$\chi 9$	$\phi 23$
5	4,1,5	5	1,2,3	$\phi 29$	$\beta 5$	$\delta 17$	$\delta 17$
6	4,1,5	5	1,2,3	$\gamma 34$	$\beta 4$	$\phi 21$	$\chi 15$
7	4,1,5	5	1,2,3	$\chi 8$	$\chi 9$	$\phi 27$	$\beta 7$
8	4,1,5	5	1,2,3	$\phi 28$	$\chi 11$	$\gamma 35$	$\delta 16$
9	4,1,5	5	1,2,3	$\phi 28$	$\chi 11$	$\gamma 34$	$\delta 17$
10	4,1,5	5	1,2,3	$\epsilon 20$	$\phi 27$	$\gamma 35$	$\gamma 33$

A análise dos resultados permite extrair algumas conclusões:

- A tabela 6.15 apresenta dez diferentes combinações de operadores para encontrar o código secreto (4,1,5). Cada combinação representa o melhor indivíduo encontrado pelo algoritmo após várias gerações. Essa combinação de operadores é sempre capaz de encontrar o código secreto proposto, desde que mantido o mesmo lance inicial.
- Os dez conjuntos de operadores apresentados descrevem uma estratégia, pois apresentam uma evolução dos ganhos entre o primeiro e o quinto lance. A figura 6.15 apresenta os lances gerados pelos conjuntos de operadores 1 e 2 (tabela 6.15).

Figura 6.15 – Execução das estratégias 1 e 2 da tabela 6.16.

É importante observar que, de acordo com o que já foi mencionado, para cada lance existe um operador que, se aplicado a ele, leva ao código procurado, portanto, existem inúmeros conjuntos de operadores que, embora resolvam o problema também em cinco lances, não representam uma estratégia de solução. A figura 6.16 apresenta um exemplo de um conjunto de cinco lances no qual não se identifica uma estratégia de solução ($\eta_{40} \eta_{40} \eta_{40} \eta_{40} \varepsilon_{18}$). Observe que não há evolução nos ganhos entre o primeiro e o quarto lance, ocorrendo apenas uma casualidade no surgimento do operador que resolve o problema no quinto lance.

Figura 6.16 – Execução de um conjunto de operadores que soluciona o problema em cinco lances mas não configura uma estratégia.

- Embora as soluções apresentadas na tabela 6.15 sejam todas diferentes entre si, observando-se apenas os grupos aos quais os operadores pertencem é possível identificar alguns *clusters* em torno dos quais surgem as soluções (tabela 6.16).

Tabela 6.16 - Análise da formação de *clusters* em dez soluções para o código (4,1,5).

Partidas	Lances				
	1	2	3	4	5
1	1,2,3	β	β	ϕ	ϕ
2	1,2,3	β	γ	γ	γ
3	1,2,3	β	γ	γ	γ
4	1,2,3	β	χ	χ	ϕ
5	1,2,3	ϕ	β	δ	δ
6	1,2,3	γ	β	ϕ	χ
7	1,2,3	χ	χ	ϕ	β
8	1,2,3	ϕ	χ	γ	δ
9	1,2,3	ϕ	χ	γ	δ
10	1,2,3	ε	ϕ	γ	γ

Por exemplo, as soluções das partidas (1,2,3,4,6) formam um *cluster* com os grupos de operadores ($\beta, \phi, \gamma, \chi$), as soluções (8,9,10) utilizam os grupos ($\phi, \delta, \chi, \gamma$) e as soluções (2,3) os grupos (β, γ).

Com base nessa análise é possível afirmar que toda e qualquer seqüência de lances executada pelo humano pode ser reproduzida a partir da combinação de um ou mais grupos de operadores. A figura 6.17 ilustra graficamente esses *clusters*.

Figura 6.17 – Representação da sobreposição de soluções para o código (4,1,5).

Os resultados anteriores sugerem ser possível a identificação de *clusters* para construção de soluções também para combinações do mesmo código. A tabela 6.17 mostra os seis códigos gerados a partir da tripla (1,4,5) e duas seqüências de operadores que levam a solução do problema em cinco lances para cada combinação.

Tabela 6.17 – Soluções para as seis possíveis combinações do código (4,1,5).

	Soluções	Código Secreto	Nº de lances	Lance inicial	Conjunto de operadores que soluciona o problema			
1	1	4,1,5	5	1,2,3	$\beta 7$	$\beta 3$	$\phi 31$	$\phi 26$
	2		5	1,2,3	$\beta 3$	$\gamma 33$	$\gamma 33$	$\gamma 35$
3	1	5,1,4	5	1,2,3	$\beta 3$	$\gamma 33$	$\gamma 33$	$\delta 17$
	2		5	1,2,3	$\phi 26$	$\chi 15$	$\beta 8$	$\gamma 35$
4	1	5,4,1	5	1,2,3	$\phi 23$	$\phi 22$	$\phi 22$	$\delta 17$
	2		5	1,2,3	$\eta 39$	$\phi 24$	$\gamma 33$	$\delta 16$
5	1	1,4,5	5	1,2,3	$\beta 5$	$\chi 9$	$\chi 8$	$\delta 16$
	2		5	1,2,3	$\phi 28$	$\phi 22$	$\gamma 35$	$\gamma 34$
6	1	1,5,4	5	1,2,3	$\epsilon 18$	$\chi 15$	$\eta 36$	$\phi 26$
	2		5	1,2,3	$\chi 10$	$\chi 14$	$\gamma 33$	$\gamma 35$

Uma análise dos grupos utilizados nas soluções encontradas para os seis códigos (tabela 6.18) permite verificar que com cinco grupos de operadores ($\beta, \delta, \chi, \gamma, \phi$), dos oito disponíveis, é possível encontrar pelo menos uma solução para cada um dos seis códigos (figura 6.18). Este resultado sugere a possibilidade de criação de um mecanismo para, uma vez identificado um *cluster*, recombina os operadores com o objetivo de resolver um número maior de códigos. Essa possibilidade será discutida no próximo capítulo na seção 7.3.

Tabela 6.18 - Análise da formação de *clusters* nas soluções encontradas para os seis códigos gerados a partir da tripla (4,1,5).

	Solução	Código Secreto	Lances				
			1	2	3	4	5
1	1	4,1,5	1,2,3	β	β	ϕ	ϕ
	2	4,1,5	1,2,3	β	γ	γ	γ
2	3	4,5,1	1,2,3	β	ϕ	η	η
	4	4,5,1	1,2,3	χ	ϕ	β	χ
3	5	5,1,4	1,2,3	β	γ	γ	δ
	6	5,1,4	1,2,3	ϕ	χ	β	γ
4	7	5,4,1	1,2,3	ϕ	ϕ	ϕ	δ
	8	5,4,1	1,2,3	η	ϕ	γ	δ
5	9	1,4,5	1,2,3	β	χ	χ	δ
	10	1,4,5	1,2,3	ϕ	ϕ	γ	γ
6	11	1,5,4	1,2,3	ε	χ	η	ϕ
	12	1,5,4	1,2,3	χ	χ	γ	γ

A figura 6.17 ilustra os *clusters* e as intersecções formadas pelos cinco grupos.

Figura 6.18 – Representação da sobreposição das soluções encontradas para os códigos gerados a partir da tripla (4,1,5).

Na próxima seção o conjunto de operadores instanciados será aplicado ao caso em que se procura uma solução que reproduza os lances de um jogador humano.

6.5.2 CG2E (conjunto de operadores instanciados – busca reproduzir lances do humano) – Análise dos Resultados

O algoritmo apresentado busca um conjunto de operadores capaz de reproduzir os lances de um jogador humano através da recombinação de um conjunto de operadores instanciados. A figura 6.19 apresenta os cinco lances realizados pelo jogador humano para encontrar o código secreto, ao lado de cada lance é apresentado o operador equivalente extraído da tabela 6.14. Para permitir avaliações e comparações, assim como nas simulações anteriores, o primeiro lance de cada indivíduo da população utilizada pelo algoritmo evolutivo é sempre igual ao primeiro lance do jogador humano.

Figura 6.19 – Solução apresentada pelo jogador humano para solução do código em cinco lances.

Da forma como os operadores foram implementados nesse algoritmo, só existe um conjunto que reproduza fielmente os lances do jogador humano. Esse conjunto foi encontrado manipulando-se uma população de 300 indivíduos durante 62 gerações (figura 6.20). Esse ensaio tem como principal resultado a tradução da estratégia utilizada pelo jogador humano em um conjunto de operadores bem definidos que podem ser reproduzidos, embora com a limitação de resolver apenas o código em questão.

Figura 6.20 – Execução do conjunto de operadores do jogador virtual (lado esquerdo) que reproduz os lances do jogador humano (lado direito) para a solução do código (5,2,1).

O próximo algoritmo explora a possibilidade de haver outras seqüências de operadores que reproduzam os ganhos do jogador humano e que possam demonstrar a equivalência entre as seqüências.

6.5.3 CG3E (conjunto de operadores instanciados – busca obter mesma avaliação do humano) – Análise dos Resultados

O algoritmo CG3E procura uma combinação de operadores extraído da tabela 6.15 que reproduza os ganhos do jogador humano. Na figura 6.21 são apresentados os lances do jogador humano (lado direito da figura 6.21) e a seqüência de operadores selecionada pelo algoritmo após manipular uma população de 300 indivíduos por 46 gerações (lado esquerdo da figura 6.21).

Figura 6.21 – Execução dos lances do jogador virtual (lado esquerdo) que reproduz a pontuação do jogador humano (lado direito).

A tabela 6.19 apresenta os lances e as descrições dos operadores utilizados pelos jogadores virtual e humano.

Tabela 6.19 - Análise da similaridade entre os lances do jogador humano e virtual.

		Jogador Virtual			Jogador Humano		
	Pinos (P,B)	Lance	Operador	Descrição Do Operador	Lance	Operador	Descrição Do operador
1	0,1	134	0,1	Lance inicial	134		Lance inicial
2	1,1	532	ϵ_{19}	Manter a segunda cor e trocar a primeira e a terceira.	325	β_4	Troca à posição da segunda cor com a primeira posição. Substitui a segunda e a terceira cor por novas cores

		Jogador Virtual			Jogador Humano		
	Pinos (P,B)	Lance	Operador	Descrição Do Operador	Lance	Operador	Descrição Do operador
3	0,2	415	β_3	Troca à posição da primeira cor com a terceira posição. Substitui a primeira e a segunda cor por novas cores	312	ϕ_{23}	Manter a primeira cor. Trocar a terceira cor pela primeira disponível. Trocar de posição a segunda com a terceira.
4	0,3	152	χ_{13}	troca a primeira com a segunda cor de posição, a segunda com a terceira cor e substitui a terceira pela primeira cor disponível.	215	ϕ_{26}	Manter a segunda cor. Trocar a primeira cor pela segunda disponível. Trocar de posição a primeira com a terceira.
5	3,0	521	δ_{17}	Move a primeira para a terceira. Move a segunda para a primeira. Move a terceira para a segunda.	521	δ_{16}	Move a primeira para a segunda. Move a segunda para a terceira. Move a terceira para a primeira.

Com base no comparativo entre as duas estratégias é possível extrair algumas conclusões:

- Embora o resultado seja restrito a uma combinação específica, é possível afirmar que houve extração de conhecimento, uma vez que o procedimento obtido pelo algoritmo permite que se reproduza a solução. Além disso, um outro jogador humano poderá aprender e extrair procedimentos válidos para outras partidas através da análise da sequência de operadores obtida pelo algoritmo.
- A figura 6.20 permite concluir que a estratégia do jogador virtual é equivalente à utilizada pelo jogador humano. Entretanto, não é possível

estabelecer uma relação unívoca entre os procedimentos, uma vez que os operadores utilizados não são os mesmos. Portanto, o que se pode concluir é que eles obtêm resultados iguais por caminhos diferentes.

O próximo capítulo reúne os principais resultados desse trabalho, apresenta as conclusões e sugestões para futuros trabalhos.

CAPÍTULO 7

CONCLUSÃO E TRABALHOS FUTUROS

A obtenção de modelos computacionais para análise e solução de problemas tem sido objeto de intensos estudos por parte dos cientistas da computação. Nesta tese de doutorado, pretendeu-se dar uma pequena contribuição ao estudo dos modelos baseados em algoritmos evolutivos. Os estudos apresentados visaram à utilização de um modelo baseado em um algoritmo evolutivo com o objetivo de identificação de estratégias utilizadas por humanos na solução de problemas de raciocínio dedutivo, tendo como caso de estudo o jogo *Mastermind*.

7.1 DISCUSSÃO DOS RESULTADOS

A pesquisa foi desenvolvida com o objetivo de construir gradativamente uma solução para o problema proposto. Os principais resultados obtidos em cada uma das etapas da investigação foram:

- A primeira etapa analisou as partidas executadas por três jogadores humanos (ensaio I), foi possível extrair um conjunto de operadores utilizados pelos jogadores para execução dos seus lances (tabela 5.5) e estabelecer relações entre as estratégias dos jogadores e seus diferentes níveis de habilidades (seção 6.1).
- O estudo com algoritmos evolutivos iniciou-se com a proposta do algoritmo genético descrito no ensaio II. Foram definidos três modelos investigativos e os respectivos algoritmos que nortearam o restante da pesquisa: 1. busca de soluções independentes do jogador humano. 2. busca de soluções que reproduzissem os lances do jogador humano (posição e cor). 3. busca de soluções que reproduzissem os resultados do jogador humano (pinos brancos e pretos). As simulações mostraram a eficiência do algoritmo

genético em encontrar uma solução para o problema nas três versões do algoritmo (seções 6.2.1, 6.2.2, 6.2.3).

As conclusões desse ensaio permitiram o avanço dos trabalhos e a proposta do classificador genético descrito no ensaio III-a e III-b:

- O classificador genético descrito no ensaio III-a apresentou uma mudança importante com relação ao modelo puramente combinatório descrito pelo algoritmo genético do ensaio II. Os indivíduos utilizados no ensaio II, que eram seqüência de cores candidatas à solução do problema, foram substituídos por seqüências de operadores (sem memória) que, se combinados corretamente, poderiam levar o indivíduo à solução do problema. As simulações com as versões CG1S e CG2S mostraram a impossibilidade de gerar estratégias gerais e eficientes a partir de recombinação dos operadores propostos (seções 6.3.1 e 6.3.2). Além disso, os resultados com a versão 3 do classificador genético (CG3S), mostrou que após várias gerações o algoritmo resultante possui algum grau de similaridade com a estratégia do jogador humano (tabela 6.4). Entretanto, essa estratégia também não é geral, pois não garante a solução do problema, mesmo quando aplicada novamente ao mesmo código secreto.

A análise desses resultados permitiu concluir que os jogadores humanos possuem uma alta capacidade de reavaliação - fruto da capacidade de memória e análise dos lances anteriores - que lhes permite escolher o operador mais adequado a cada nova situação. Além disso, os dados mostraram que um conjunto de operadores capaz de resolver o caso "A" não é capaz de resolver qualquer caso, mas apenas um pequeno conjunto de casos (tabela 6.5).

- Com base nos resultados do ensaio III-a foi proposto uma nova versão do algoritmo, no qual foi implementado um conjunto de operadores com memória. Os resultados mostraram (tabela 6.13) que a introdução do mecanismo de memória auxiliou para que algumas soluções específicas fossem encontradas melhorando os resultados quando comparado ao modelo sem memória. Além disso, o ensaio permitiu mostrar que os operadores, até então utilizados e extraídos a partir dos lances executados por humanos (tabela 5.7), possuem um grau de indecisão caracterizado pelos mecanismos

aleatórios de seleção de cores e posições que impedem que o algoritmo convirja para uma solução geral (figuras 6.10 e 6.11). O ensaio ainda permitiu concluir que o jogador humano faz uso de funções cognitivas complexas que englobam mecanismos de memória e análise dos lances futuros, o que não pode ser contemplado pelo modelo até então proposto (figuras 6.12).

Para a solução do problema da aleatoriedade dos operadores utilizados por jogadores humanos foi proposta no ensaio IV a substituição dos operadores da tabela 5.7 por um conjunto de operadores instanciados. Os operadores desse novo conjunto foram criados a partir de um estudo analítico das situações possíveis dentro de uma partida. Os principais resultados do ensaio IV foram:

- O algoritmo evolutivo utilizado foi capaz de produzir conjuntos de operadores que efetivamente descrevem uma estratégia, pois são passíveis de reprodução e apresentam uma evolução dos ganhos entre o primeiro lance e o último lance. Entretanto, tais estratégias não são gerais e são capazes de resolver um único código.
- A análise das diferentes soluções produzidas pelo algoritmo possibilitou identificar interseções entre as estratégias e formação *clusters* de operadores capazes de gerar mais de uma solução a partir da sua combinação (tabela 6.16 e figura 6.15).
- Os resultados das análises dos *clusters* encontrados nas soluções para combinações do mesmo código demonstram ser possível produzir soluções locais (para todas as combinações de um mesmo código) recombinao adequadamente os grupos de operadores (tabelas 6.17, 6.18 e figura 6.16).

O estudo de caso proposto mostrou ser um complexo problema de raciocínio dedutivo, bastante suscetível a condições iniciais e a fatores de aleatoriedade que são tratados pelo jogador humano com o uso combinado da memória dos lances passados e avaliação de lances futuros. O fator aleatoriedade presentes nos operadores utilizados pelo jogador humano foi contornado através da utilização de operadores determinísticos, cuja aplicação não permite dúvidas, não havendo, portanto fator aleatório na escolha do lance. Como resultado, apresentou-se um método capaz de identificar procedimentos de raciocínio através da utilização de um algoritmo genético que recombina tais operadores. Além disso, é possível afirmar

que houve extração de conhecimento, pois os procedimentos propostos pelo algoritmo podem ser reproduzidos tanto pelo jogador virtual quanto pelo jogador humano. Entretanto, os procedimentos encontrados não se mostraram procedimentos gerais, solucionando apenas casos específicos do problema. Em contrapartida, os resultados identificaram *clusters* formados de operadores que sugerem que poderão emergir, da combinação correta desses grupos, procedimentos capazes de resolver um número maior de problemas. O mecanismo de seleção de operadores deverá simular o procedimento humano de avaliação de lances futuros para a solução do problema, tal como foi demonstrado no item 6.4.3 figuras 6.13 e 6.14. O modelo investigado nesse trabalho pode ser um ótimo ponto de partida para o estudo da solução automática de outros problemas com características semelhantes ao estudado, com vários caminhos e procedimentos para se chegar a uma mesma solução.

7.2 CONCLUSÕES FINAIS

A proposta deste trabalho nasceu de inúmeras discussões e de algumas questões fundamentais inicialmente colocadas:

- Como modelos de aprendizagem humana podem auxiliar na aquisição de conhecimento em sistemas artificiais?
- Uma máquina pode aprender como um humano raciocina ao tentar resolver um problema?
- Quais passos um humano executa ao tentar aprender a resolver um problema?

Esta tese de doutorado se propõe, antes de mais nada, a tratar de tais questões, partindo da proposta do uso de um algoritmo evolutivo. A medida que os ensaios foram sendo realizados, particularidades do processo de solução apresentado pelo humano e de limitações do modelo inicial foram evidenciados. Com base nessas evidências, a proposta foi atualizada e variantes mais e mais complexas do algoritmo inicial foram implementadas e sugeridas (algoritmos genéticos, classificadores genéticos, classificadores genéticos com memória, classificadores genéticos com operadores instanciados, máquinas de estado, máquinas de estado adaptativas).

Desta forma, o trabalho apresentou como contribuição um modelo no qual a representação do conhecimento é descrita por uma seqüência de operadores, sucessivamente alterada num processo evolutivo por um algoritmo genético, cujo referencial é um personagem humano. Outras formas ainda mais complexas se farão necessárias para se aproximar do objetivo principal dos estudos nesta linha de pesquisa, que é desenvolvimento de máquinas que possam atuar de forma inteligente, efetivamente ajudando a desenvolver ou extrair linhas de raciocínio para melhor resolver os mais diversos problemas que venham a ser apresentados. O jogo de raciocínio dedutivo utilizado (*Mastermind*) foi, portanto apenas um estudo de caso, que se mostrou conveniente e interessante para iniciar as discussões dessas questões fundamentais.

7.3 TRABALHOS FUTUROS

Como trabalhos futuros, propõem-se a continuação da pesquisa evoluindo o modelo apresentado com o objetivo de se conseguir a extração de procedimentos que resolvam classes de códigos como, por exemplo, todos os códigos gerados por uma tripla de números. A proposta é implementar um mecanismo que simule a capacidade do jogador humano de avaliação de lances futuros. Partindo-se do modelo utilizado no ensaio IV (conjunto de operadores instanciados) propõe-se um novo modelo de indivíduo para formação da população que será manipulada pelo algoritmo genético. Nesse modelo, cada indivíduo representa um conjunto de grupos de operadores (figura 7.1). O cálculo do *fitness* será executado para todos os operadores de cada um dos grupos que formam um indivíduo, sendo que, o operador de melhor desempenho será utilizado como parâmetro de seleção.

Figura 7.1 – Indivíduos formados por conjuntos de operador.

Após o algoritmo evoluir por várias gerações o melhor indivíduo, formado por um conjunto de grupos de operadores, será executado a partir de um mecanismo que simula a avaliação e os procedimentos dedutivos do jogador humano. O mecanismo é uma máquina de estado com características adaptativas [NETO2007; PISTORI2005], que a partir do lance inicial avalia as condições e seleciona o operador mais adequado para execução (figura 7.2).

Figura 7.2 – Máquina de estados para seleção de operadores.

Espera-se que a máquina de estados possa recombinar adequadamente os grupos e seus operadores, de tal forma a resolver um número maior de códigos. A figura 7.3 ilustra o funcionamento da máquina de estado para o caso em que o conjunto de operadores ($\zeta, \kappa, \lambda, \mu$), se combinado adequadamente, é capaz de gerar diferentes soluções.

Figura 7.3 – Máquina de estado para seleção automática de operadores.

REFERÊNCIAS BIBLIOGRÁFICAS

[AGOSTINHO2003] AGOSTINHO, R. (2003) Solving Puzzles and Games by Evolutionary Algorithms. In: Proceedings of the Mathematical Techniques and Problems in Telecommunications, CIM 24, pp 281-312.
<http://www.laseeb.org/publications/index.html>

[ANDREAE1995] ANDREAE, J. H. (1995) The future of associative learning. Artificial Neural Networks and Expert Systems. In: Proceedings of the Second New Zealand International Two-Stream Conference on, 194-197.

[ATMAR1992] ATMAR, W. (1992) On the Rules and Nature of Simulated Evolutionary Programming. In: Proceedings of the of the First Ann. Conf. On Evolutionary Programming, 17-2.

[ATMAR1994] ATMAR W. (1994) Notes on the Simulation of Evolution. In: IEEE Transactions on Neural Networks, 5, 1, 130-148.

[AZEVEDO1999] AZEVEDO, F. M. (1999) Algoritmos Genéticos em Redes Neurais Artificiais. V Escola de Redes Neurais, ITA.

[BÄCK1997] BÄCK, T.; HAMMEL, U. e SCHWEFEL, H. P. (1997) Evolutionary Computation: Comments on the History and Current State. In: IEEE Transaction on Evolutionary Computation, 1, 2.

[BANZHAF2000] BANZHAF, W.; KOZA, J.R.; RYAN, C.; SPECTOR, L.; JACOB, C. (2000) Genetic programming, Intelligent Systems and Their Applications. IEEE, 15, 3, 74 – 84.

[BAUM1995] BAUM, C. G. E. B. e BOMEH, D. (1995) On genetic algorithms. In: Proceedings of the COLT'95. ACM Inc., Expanded version: "Where Genetic Algorithms Excel".

[BAUM1999] BAUM, W. M. (1999) Compreender o Behaviorismo. Artmed.

[BELEW1988] BELEW, R. K. e FORREST, S. (1988) Learning and programming in classifier systems. Machine Learning, Springer Netherlands, 3, 2-3, 193-223.

[BENTO1999] BENTO, L.; PEREIRA, L.; e ROSA A. (1999) Mastermind by evolutionary algorithms. In: Proceedings of the SAC 99.

[BERNIER1996] BERNIER, J. L.; HERRAIZ, C. I.; MERELO, J. J.; OLMEDA, S. e PIETRO, A. (1996) Solving Master Mind Using GAs and Simulated Annealing: A Case of Dynamic Constraint Optimization. Lecture Notes in Computer Science, 1141, 554-563.

[BERSINI2002] BERSINI, H. (2002) The immune and the chemical crossover. In: IEEE Transactions on Evolutionary Computation, 6, 3, 306-313.

- [BOGOMOLNY1999] BOGOMOLNY, A. (1999) Mastermind. <http://www.cut-the-knot.org/ctk/Mastermind.shtml> acessado em: 05/05/2005.
- [BRIAN1993] BRIAN ARTHUR, W. (1993) On designing economic agents that behave like human agents. *Evolutionary Economics*, 3:1-22.
- [CASTILLO2003] CASTILLO, C.; LURGI, M. e MARTINEZ, I. (2003) Chimps: an evolutionary reinforcement learning approach for soccer agents, in: *Systems, Man and Cybernetics*. In *Proceedings of the IEEE International Conference on*, 1, 60-65.
- [CHEN1996] CHEN, Z.; CUNHA, C. e HOMER, S. (1996) Finding a hidden code by asking questions. In: *Proceedings of the Computing and Combinatorics (COCOON'96)*, 50-55.
- [CHIEW2004] CHIEW, V. e YINGXU W. (2004) Formal description of the cognitive process of problem solving. In: *Proceedings of the Third IEEE International Conference on Cognitive Informatics*, 74-83.
- [CHVATAL1983] CHVATAL, V. (1983) "Mastermind," *Combinatorica*, 325-329.
- [CONEN1995] CONEN, W.; ADELSBERGER, N.H. (1995) Systems design for learning Systems Engineering of Computer Based Systems. *Proceedings of the 1995 International Symposium and Workshop on*, 159-166.
- [COX2005] COX, M. T. (2005) Metacognition in computation: A selected research review. *Artificial Intelligence*, 169, 2, 104-141.
- [CROOKS1988] CROOKS, R. L. e STEIN, J. (1988) *Psychology: science, behavior, and life*. Holt, Rinehart and Winston, Inc.
- [DIETTERICH2003] DIETTERICH, T. G. (2003) Machine Learning. In *Nature Encyclopedia of Cognitive Science*, London: Macmillan. <http://web.engr.oregonstate.edu/~tgd/publications/learning-and-reasoning.pdf> (acessado em: 12/05).
- [DRESCHER1991] DRESCHER, G. (1991) *Mide-Up Minds: A Constructivist Approach to Artificial Intelligence*. MIT Press.
- [FOGEL1966] FOGEL, L. J., OWENS, A. J. e WALSH, M. J., (1966) *Artificial Intelligent Through Simulated Evolution*, John Wiley.
- [GAREY1979] GAREY, M. R. e JOHNSON, D. S. (1979) *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W.H. Freeman.
- [GOLDBERG1985] GOLDBERG, D. E. e LINGLE, R. (1985) Alleles, loci, and traveling salesman problem. In: *Proceedings of an International Conference on Genetic Algorithms and their Applications*, 154-159.

[GOLDBERG1989] GOLDBERG, D. E. (1989) Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley, MA.

[GOLDBERG1994] GOLDBERG, D. E. (1994) Genetic and Evolutionary Algorithms Come of Age. Communications of the ACM, 37, 3, 113-119.

[GRUAU1993] GRUAU, F. e WHITLEY, D. (1993) Adding Learning to the Cellular Development of Neural Networks: Evolution and the Baldwin Effect. Evolutionary Computation, 1, 213-244.

[GUTIN2002] GUTIN, G. e PUNNEN, A. P. (2002) The Traveling Salesman Problem and Its Variations. Springer.

[HAYWARD2004] HAYWARD, S. (2004) Setting up performance surface of an artificial neural network with genetic algorithm optimization: in search of an accurate and profitable prediction of stock trading Evolutionary Computation, CEC2004. Congress on, 1, 948 – 954.

[HOLLAND1975] HOLLAND, J. H. (1975) Adaptation in Natural and Artificial Systems. The University of Michigan Press, Ann Arbor, MI.

[JONES2000] JONES, P. M.; JACOBS, J. L. (2000) Cooperative problem solving in human-machine systems: theory, models, and intelligent associate systems. In: IEEE Transactions on Systems, Man and Cybernetics, Part C, 30, 4, 397-407, 2000.

[KABATIANSKI2000] KABATIANSKI, G.; LEBEDEV, V. e THORPE, J. (2000) The Mastermind game and the rigidity of the Hamming space, Information Theory. In: Proceedings of the IEEE International Symposium on, 25-30.

[KELLERER2005] KELLERER, H.; PFERSCHY, U. E PISINGER, D. (2005) Knapsack Problems. Springer Verlag.

[KNUTH1977] KNUTH, D. E. (1977) "The Computer as Mastermind." Journal of Recreational Mathematics, 9, 1-6.

[KOYAMA1993] KOYAMA, K. e LAI, T. (1993) "An Optimal Mastermind Strategy," Journal of Recreational Mathematics, 25, 4, 251-256.

[KOZA1992a] KOZA, J. R. (1992) Genetic Programming: On the Programming of Computers by Means of Natural Selection. MIT Press.

[KOZA1992b] KOZA, J. R. (1992) Genetic Evolution and Co-evolution of Games Strategies, International Conference on Game Theory and Its Applications held at Stony Brook, New York, 15.

[KOZA2003] KOZA, J. R. (2003) KEANE, M. A. e STREETER, M. J. What's AI done for me lately? Genetic programming's human-competitive results. In: IEEE Intelligent Systems. 18, 3, 25-31.

[KUBOTA1996] KUBOTA, N.; SHIMOJIMA, K. e FUKUDA, T. (1996) Virus-evolutionary genetic algorithm-ecological model on planar grid, Fuzzy Information Processing Society, NAFIPS, 505-509.

[KURAHASHI1999] KURAHASHI, S.; MINAMI, U. e TERANO, T. (1999) Why not multiple solutions: agent-based social interaction analysis via inverse simulation, Systems, Man, and Cybernetics, IEEE SMC '99 Conference Proceedings. In: Proceedings of the IEEE International Conference on 2, 522–527.

[MAGUITMAN2004] MAGUITMAN, A. G. (2004) Intelligent Support for Knowledge Capture and Construction, Tese submetida para obtenção grau de Doutor, Department of Computer Science, Indiana University, Indiana.

[MARTELLO1990] MARTELLO, S. E TOTH, P. (1990) Knapsack Problems: Algorithms and Computer Implementations. John Wiley & Sons.

[MAYR1988] MAYR, E. (1988) Toward a New Philosophy of Biology: Observations of an Evolutionist, Belknap Press.

[MCAULAY1994] MCAULAY, A. D. e JAE CHAN OH, (1994) Improving learning of genetic rule-based classifier systems Systems, In: IEEE Transactions on Man and Cybernetics, 24, 1, 152-159.

[MERELO1996] MERELO, J. (1996) Genetic Mastermind, a case of dynamic constrain optimisation. GeNeura Technical Report G-96-1. Univ. Granada.

[MERELO1999] MERELO, J.; CARPIO, P. C.; RIVAS, V. e ROMERO G. (1999) Finding a Needle in a Haystack using Hints and Evolutionary Computation: the Case of Genetic Mastermind. <http://geneura.ugr.es/~jmerelo/newGenMM/>

[MEYER1991] MEYER, T. P. e PACKARD, N. H. (1991) Local forecasting of high dimensional chaotic dynamics. Technical Report CCSR-91-1, Center for Complex Systems Research, Beckman Institute, University of Illinois at Urbana Champaign.

[MICHALEWICZ1996] MICHALEWICZ, Z. (1996) Genetic algorithms + Data Structures = Evolution Programs. 3, Springer-Verlag.

[MILES2004] MILES, C.; LOUIS, S. J.; COLE, N. e MCDONNELL, J. (2004) Learning to play like a human: case injected genetic algorithms for strategic computer gaming, Evolutionary Computation. CEC2004. 1441–1448.

[MITCHELL1992] MITCHELL, M.; STEPHANIE. F. e HOLLAND J. H. (1992) The royal road for genetic algorithms: fitness landscapes and GA performance. In F. J. Varela and P. Bourgine. In: Proceedings of the First European Conference on Artificial Life. Toward a Practice of Autonomous Systems, 245-254.

[MITCHELL1994] MITCHELL, M.; CRUTCH_ELD, J. P. e HRABER, P. T. (1994) Evolving cellular automata to perform computations: Mechanisms and impediments. In: *Physica D*, 75, 361-391.

[MITCHELL1999a] MITCHELL, M. (1999) *MASTERMIND Mathematics: Logic, Strategies, and Proofs*. Berkeley, Key Curriculum Press.

[MITCHELL1999b] MITCHELL, M. (1999) *An Introduction to Genetic Algorithms*. The MIT Press.

[MOREIRA1999] MOREIRA, M. A. (1999) *Teorias de Aprendizagem*. Editora Pedagógica e Universitária (E.P.U.), São Paulo, Brasil.

[MUNTJEWERFF1999] MUNTJEWERFF, A. J. (1999) PROSA an instructional environment for learning legal case solving, 14th BILETA Conference: "CYBERSPACE 1999: Crime, Criminal Justice and the Internet", 1, 1-14.

[NETO2007] NETO, J., J. (2007) Adaptatividade e Tecnologia Adaptativa - Tutorial baseado no WTA 2007. *Revista IEEE América Latina*, 5, 495-496.

[NETTO2007] NETTO, M. L. (2007) *Computação Evolutiva e Cognitiva: Simulação em Vida Artificial e Cognição*, Tese de Livre Docência – Escola Politécnica da USP.

[NEWELL1955] NEWELL, A. (1955) The chess machine: An example of dealing with a complex task by adaptation. In *Proceedings of Western Joint Computer Conference*. Institute of Radio Engineers, New York, 101-108.

[NEWELL1958] NEWELL, A.; Shaw, J. C. e Simon, H. A. (1958) Chess-playing programs and the problem of complexity. *IBM J. Res. Develop.* 2, 320-25.

[NEWELL1959] NEWELL, A.; SHAW, J.C.; SIMON, H.A. (1959) Report on a general problem-solving program. In: *Proceedings of the International Conference on Information Processing*. pp. 256-264.

[NEWELL1963] NEWELL, A. e Simon, H.A. (1963) *GPS, a Program that Simulates Human Thought*, Computers and Thought, E. Feigenbaum and J. Feldman, McGraw-Hill, New York, 279-293.

[NEWIRTH1982] NEWIRTH, E. (1982) Some Strategies For Mastermind. *Zeitschrift fur Operations Research, Série B*, 26(8); 257-278.

[PEROTTO2004] PEROTTO, F. S.; VICARI, R. M. e ÁLVARES, L. O. (2004) *A Alternativa Construtivista em Inteligência Artificial*, I Workshop de Computação da Região Sul Florianópolis, SC.
(disponível em: <http://inf.unisul.br/~ines/workcomp/cd/pdfs/2890.pdf>).

[PIAGET1982] PIAGET, J. (1982) O nascimento da inteligência na criança. 4. ed. Rio de Janeiro, Zahar.

[PISTORI2005] PISTORI, H., MARTINS, P. S., CASTRO JR., A. A. (2005) Adaptive Finite State Automata and Genetic Algorithms - Merging Individual Adaptation and Population Evolution. Proceedings of International Conference on Adaptive and Natural Computing Algorithms - ICANNGA 2005, Coimbra, 21-23.

[POSNER1993] POSNER, M. I. (1993) Foundations of Cognitive Science, MIT Press, 12, 469-497.

[PUCHKIN1969] PUCHKIN, V. N. (1969) Heurística - A Ciência do Pensamento Criador, Zahar.

[PYLYSHYN1980] PYLYSHYN, Z. (1980) Computation and cognition: Issues in the foundation of cognitive science. Behavioral and Brain Sciences 3: 111–132.

[REGGIA2001] REGGIA, J. A.; SCHULZ, R.; WILKINSON, G. S. e JUAN U. (2001) Conditions Enabling the Evolution of Inter-Agent Signaling in an Artificial World, Artificial Life, 7,1, 3–32.

[REZENDE2003] REZENDE, S. O. (2003) Sistemas Inteligentes, Manole.

[RIBEIRO-FILHO1994] RIBEIRO-FILHO, J. L.; TRELEAVEN, P. C. (1994) GAME: a framework for programming genetic algorithms applications. Evolutionary Computation, IEEE World Congress on Computational Intelligence, Proceedings of the First IEEE Conference on Jun 1994, 840 – 845, 2.

[ROCHE1997] ROCHE, J. R. (1997) The value of adaptive questions in generalized Mastermind; in: Information Theory. In: Proceedings Of the IEEE International Symposium on, Center for Commun. Res., Princeton, USA.

[ROISENBERG1998] ROISENBERG, M. (1998) Emergência de inteligência em agentes autônomos através de modelos inspirados na natureza. Doutorado em engenharia elétrica, Universidade Federal de Santa Catarina.

[ROSA1999] ROSA, A. C.; PEREIRA L.; BENTO L. (1999) Mastermind by Evolutionary Algorithms, Symposium on Applied Computing ACM SAC 99, 307-311.

[RUSSEL1995] RUSSEL, S. e NORVIG, P (1995) Artificial Intelligence: A Modern Approach, Prentice Hall.

[SAMUEL1967] SAMUEL, A. L. (1967) Some studies in machine learning using the game of checkers. II-Recent progress. IBM Journal on Research and Development, 11, 601-617.

[SANDOZ1993] SANDOZ, F. e BAPTISTE, P. (1993) A cognitive system allowing learning. In Proceedings of the International Conference on Systems, Man and Cybernetics, Systems Engineering in the Service of Humans, 632-637, 4.

[SHANNON1950] SHANNON, C. E. (1950) Programming a computer for playing chess. Philosophical Magazine, 41, 4, 256–275.

[SKINNER1953] SKINNER, B. (1953) Science of Human Behavior. Macmillan.

[SMITH1980] SMITH, S. F. (1980) *A Learning System Based on Genetic Adaptive Algorithms*. PhD dissertation. Pittsburgh: University of Pittsburgh.

[STUCKMAN2005] STUCKMAN, JEFF; ZHANG, GUO-QIANG (2005) Mastermind is NP-Complete, arXiv:cs/0512049.
http://arxiv.org/PS_cache/cs/pdf/0512/0512049.pdf

[TANOMARU1995] TANOMARU, J. (1995) Motivação, Fundamentos e Aplicações de Algoritmos Genéticos; II Congresso Brasileiro de Redes Neurais, Curitiba, Paraná, Brasil.

[VOMLEL2004] VOMLEL J. (2004) Bayesian networks in Mastermind. In: Proceedings of the 7th Czech-Japan Seminar on Data Analysis and Decision Making under Uncertainty, 185-190.

[WANG2004] WANG, YINGXU, YING WANG, S PATE, e D. PATEL (2004) A Layered Reference Model of the Brain, In: IEEE Transactions on Systems, Man, and Cybernetics, 34.

[WAZLAWICK1997] WAZLAWICK, R. S. (1997) Functional concept acquisition using action schemata. In: Proceedings of the Intelligent Information Systems (IIS '97), 282-286.

[WHITLEY1989] WHITLEY, D.; STARKWEATHER, T. e Fuquay D. (1989) Scheduling problems and travelling salesman: The genetic edge recombination operator. In: Proceedings of the Third International Conference on Genetic Algorithms, 133-140, 36.

[WILSON1999] WILSON, R. A. e KEIL, F. C. (1999) The MIT Encyclopedia of the Cognitive Sciences, MIT Press.

BIBLIOGRAFIA COMPLEMENTAR

- [ADAMI1998] ADAMI, C.(1998) Introduction to Artificial Life, Springer Verlag.
- [BARONE2002] BARONE, D. et al. (2002) Sociedades Artificiais, Bookman.
- [BODEN1998] BODEN, M. A. (1998) Piaget, Fontana Modern Masters.
- [BONABEAU1998] BONABEAU, E. W. e THERULAZ, G. (1994) Why do we need artificial life? Artificial Life, MIT Press.
- [CARROLL2002] CARROLL, J. M. (2002) Scenarios and design cognition. Requirements Engineering. In: Proceedings of the IEEE Joint International Conference on, 3–5.
- [DAVID1989] DAVID, T. G. (1989) Genetic Algorithms in Search, Optimisation and Machine Learning, Addison-Wesley.
- [FOGEL1996] FOGEL, L. J.; OWENS, A.J.; WALSH, M. J. (1996) Artificial intelligence through simulated evolution, Wiley.
- [GARDNER1995] GARDNER, H. (1995) A Nova Ciência da Mente; trad. Cláudia M. Caon. - São Paulo: EDUSP, 1995.
- [GREGOR2003] GREGOR, H. (2003) On the Genetic Evolution of a Perfect Tic-Tac-Toe Strategy. Computer Science Program, Stanford University Stanford, CA 94305, 2003. Disponível em: <http://www.genetic-programming.org/sp2003/Hochmuth.pdf> (acessado em: 02/2006).
- [LAI2003] LAI, T. (2003) Discovery of Understandable Math Formulas Using Genetic Programming, Computer Science Program, Stanford University Stanford, CA 94305. Disponível em: <http://www.genetic-programming.org/sp2003/Lai.pdf> (acessado em: 02/2006).
- [MAYER1992] MAYER, R. E. (1992) Thinking, Problem Solving, Cognition, 36-38, W.H.Freeman and Company.
- [NETTO2002] NETTO, L. N.; DEL NERO, H. S. e MARQUES, E. A. R. (2002) An Evolutionary Learning Strategy for Multi Agents. In: SEAL2002 - 4th Asia-Pacific Conference on Simulated Evolution and Learning.
- [PIAGET 1983] PIAGET, J. (1983) A epistemologia genética. 2. ed. São Paulo, Abril Cultural, (Col. Os pensadores).
- [PINKER 1998] PINKER, S. (1998) Como a Mente Funciona; trad. Laura T. Motta. - São Paulo: Companhia das Letras.
- [RICH1991] RICH, E. E KNIGHT, K. (1991) Artificial Intelligence, McGraw-Hill.

[ROOK1993] ROOK, F. W. e DONNELL, M. L. (1993) Human cognition and the expert system interface: mental models and inference explanations Systems, Man and Cybernetics, IEEE Transactions on, 23, 6, 1649–1661.

[NELSON2007] NELSON, T. A brief history of the master mind board game. <http://www.tnelson.demon.co.uk/mastermind/history.html>. (acessado em: 03/2007).

[WAZLAWICK1992] WAZLAWICK, R. S. (1992) Um modelo operatório para construção de conhecimento, Tese submetida para obtenção grau de Doutor, Florianópolis.

[YUFIK2002] YUFIK, Y.M. (2002) How the mind works: an exercise in pragmatism In: Proceedings of the International Joint Conference on Neural Networks (IJCNN '02), 3, 2265-2269.

APÊNDICE A

MÉTODOS DE CÁLCULO PARA A FUNÇÃO FITNESS

A seguir, são apresentados os resultados da comparação entre o método para o cálculo do *fitness* descrito na seção 5.6.3, o qual utiliza a distância euclidiana entre os resultados de dois lances como medida de similaridade e um outro método descrito na literatura [BENTO1999]. O objetivo da comparação é avaliar a qualidade do método utilizado nos diversos algoritmos propostos neste trabalho. A seção seguinte descreve o método alternativo e aqui denominado de “*fitness* condicional” e a seção A.2 descreve os resultados de um conjunto de simulações utilizando os dois métodos e apresenta uma análise comparativa entre eles. As simulações foram realizadas tendo como base o algoritmo AG1 descrito na tabela 6.2

A.1 FITNESS CONDICIONAL

A função de mérito ou *fitness* calcula em que medida cada um dos indivíduos da população respeita os ganhos já obtidos nos lances anteriores. O valor do *fitness* de cada indivíduo i é calculado através do seguinte procedimento [BENTO1999]:

1. Primeiro calcula-se as variáveis NP , NB , NN , número de pinos pretos, número de pinos brancos e número de cores incorretas respectivamente obtidos no lance LN (melhor indivíduo da geração n), quando confrontado com o código secreto. Sendo NN dado por: $NN = P - NP - NB$ em que P é número de cores existentes no código secreto

2. Em seguida, calcula-se as variáveis XP , XB , XN , número de pinos pretos, número de pinos brancos e número de cores incorretas respectivamente obtidos pelo indivíduo i quando confrontado com LN (melhor indivíduo da geração n).

3. Então, calculam-se os coeficientes de ponderação $A_i(n)$, $B_i(n)$ e $C_i(n)$. Os coeficientes de ponderação são valores proporcionais ao número de posições e cores corretas no indivíduo i quando comparado a um lance qualquer LN . Os valores de $A_i(n)$ e $C_i(n)$ são proporcionais ao número de posições concordantes entre o

indivíduo i e a seqüência avaliada na geração n (LN), sendo $C_i(n)$ o parâmetro utilizado para o caso em que todas as cores do código secreto já são conhecidas, faltando apenas decifrar as posições. O parâmetro $B_i(n)$ é proporcional ao número de cores concordantes entre o indivíduo i e a seqüência avaliada na geração n (LN).

$$A_i(n) = 2P - |NP - XP_i| - |NB - XB_i| \quad (A.1)$$

$$B_i(n) = P - |NN - XN_i| \text{ com } NN = P - NP - NB \text{ e } XN = P - XP - XB \quad (A.2)$$

$$C(n) = P - |NP - XP_i| \quad (A.3)$$

Em seguida, calcula-se o *fitness* de cada indivíduo através das equações A.4 e A.5 de acordo com o seguinte critério:

a. Se o último indivíduo avaliado possui cores não pertencentes ao código secreto, ou seja, se $NN > 0$ então temos:

$$f_i = \sum_{n=1}^{NG} (\text{pesodaposicao} * A_i(n) + \text{pesodacor} * B_i(n)) \quad (A.4)$$

sendo:

pesodaposicao=5

pesodacor=10

b. Se no último lance avaliado todas as cores do código secreto estão presentes ($NN=0$) então temos:

$$f_i = \sum_{n=1}^{NG} (\text{pesodaposicao} * C_i(n) + \text{pesodacor} * B_i(n)) \quad (A.5)$$

A.2 COMPARAÇÃO ENTRE MÉTODOS PARA CÁLCULO DO FITNESS

A seguir, apresentam-se os resultados obtidos a partir da simulação de um conjunto de cem partidas no modo clássico (4x6) partindo sempre do mesmo código secreto e com uma população inicial de 100 indivíduos. Os histogramas descrevem o número de gerações necessárias para que o AG1S (ensaio II seção 6.3) encontre o código secreto utilizando-se duas metodologias diferentes para o cálculo da função *fitness*. A figura A.1 mostra os resultados obtidos com a metodologia para o cálculo do *fitness* descrita na seção A.1 e a figura A.2 apresenta os resultados obtidos utilizando-se a função *fitness* baseada na distância euclidiana descrita na seção 6.5.5 (equação 6.8).

Figura A.1 – Distribuição do número de lances necessários para solução do jogo em 100 partidas utilizando o Algoritmo Genético AG1 utilizando a função *fitness* condicional e uma população de 100 indivíduos.

Figura A.2 – Distribuição do número de lances necessários para solução do jogo em 100 partidas utilizando o Algoritmo Genético AG1 utilizando o *fitness* euclidiano e uma população de 100 indivíduos.

A tabela A.1 mostra um comparativo estatístico entre os dados obtidos com as duas diferentes metodologias para o cálculo da adaptabilidade dos indivíduos da população.

Tabela A.1 – Dados estatísticos comparando as duas metodologias utilizadas para o cálculo do *fitness* dos indivíduos de uma população.

	<i>Fitness</i> (1) (coeficiente de ponderação)	<i>Fitness</i> (2) (medida de distância)
Número Médio de	6,8	6,44
Mediana	7	6
Desvio Padrão	1,65	3,02
Máximo	10	16
Mínimo	3	2

Os dados dos dois conjuntos de simulações apresentam distribuição que se aproxima de uma distribuição normal a qual é caracterizada pela simetria em torno da média. Os valores médios obtidos pelos dois métodos ficaram bastante próximos havendo uma variação a favor do método proposto nesse trabalho (medida de distância) de apenas 5%. Embora o valor médio obtido pelos dois métodos permita concluir a equivalência entre eles, acredita-se que o cálculo da adaptabilidade utilizando-se a medida de distância apresente pelo menos uma vantagem significativa em relação ao método proposto no trabalho de Bento e col. [Bento1999]. O *fitness* por distância é uma medida direta e não utiliza parâmetros subjetivos como os coeficientes de ponderação denominados de “peso da cor” e “peso da posição”. O *fitness* euclidiano ou *fitness* por distância será utilizado nos ensaios seguintes.