UNIVERSIDADE FEDERAL DE GOIÁS
ESCOLA DE ENGENHARIA ELÉTRICA E DE COMPUTAÇÃO

FERNANDO PIRKEL TSUKAHARA

# Organização Topológica de Objetos baseada em Algoritmos Genéticos Hierárquicos

## Organização de objetos multidimensionais em matrizes bidimensionais

Goiânia
2007

FERNANDO PIRKEL TSUKAHARA

# Organização Topológica de Objetos baseada em Algoritmos Genéticos Hierárquicos

## Organização de objetos multidimensionais em matrizes bidimensionais

Dissertação apresentada ao Programa de Pós–Graduação do Escola de Engenharia Elétrica e de Computação da Universidade Federal de Goiás, como requisito parcial para obtenção do título de Mestre no Programa de Mestrado em Engenharia Elétrica e de Computação.

**Área de concentração:** Sistemas Inteligentes

**Orientador:** Prof. Weber Martins

Goiânia
2007

FERNANDO PIRKEL TSUKAHARA

# Organização Topológica de Objetos baseada em Algoritmos Genéticos Hierárquicos

## Organização de objetos multidimensionais em matrizes bidimensionais

Dissertação defendida no Programa de Pós–Graduação da Escola de Engenharia Elétrica e de Computação da Universidade Federal de Goiás como requisito parcial para obtenção do título de Mestre em Programa de Mestrado em Engenharia Elétrica e de Computação, aprovada em 27 de Setembro de 2007, pela Banca Examinadora constituída pelos professores:

---

**Prof. Weber Martins**
Escola de Engenharia Elétrica e de Computação – UFG
Presidente da Banca

---

**Prof. <Nome do membro da banca>**
<Unidade acadêmica> – <Sigla da universidade>

---

**Prof. <Nome do membro da banca>**
<Unidade acadêmica> – <Sigla da universidade>

**Fernando Pirkel Tsukahara**

Bacharel em Ciência da Computação pela Universidade Católica de Goiás (2001). Especialista em Redes de Computadores, cursou o Mestrado na Escola de Engenharia Elétrica na linha de pesquisa de Sistemas Inteligentes sob orientação do Profº PhD Weber Martins com conclusão em Setembro de 2007. Atualmente é Profº na Faculdade de Tecnologia SENAC Goiás e desenvolve pesquisas na área de mercado financeiro.

O trabalho é dedicado à memória de meu pai Fernandes Massao Tsukahara, por todos os ensinamentos de vida e apoio incondicional nesta empreitada.

# Agradecimentos

Agradecimento especial à toda minha família pelo apoio, compreensão e tempo ausente.

Agradeço à minha esposa Renatha por todo o carinho e compreensão.

Agradeço aos companheiros do Grupo Pireneus: Ulisses Afonseca, Delermando Branquinho Filho, Lena Lúcia de Moraes, Viviane Margarida, Lauro Nalini e Carol.

Agradeço aos professores do mestrado: Leonardo Brito, Gelson Cruz, Antônio Baleeiro e Rodrigo Lemos.

Agradeço aos colegas do SENAC, SENAI e UCG pela força e incentivo profissional.

Agradecimento especial ao ilustrador e amigo Galvão Henrique Bertazzi (http://www.vidabesta.com), responsável pelas ilustrações do Capítulo 2 (páginas 41 e 42).

Agradecimentos especiais ao orientador e amigo Weber Martins!

Valeu!

Só temos consciência do belo
Quando conhecemos o feio.
Só temos consciência do bom
Quando conhecemos o mau.
Porquanto o Ser e o Existir
Se engendram mutuamente.
O fácil e o difícil se completam.
O grande e o pequeno são complementares.
O alto e o baixo formam um todo.
O som e o silêncio formam a harmonia.
O passado e o futuro geram o tempo.
Eis por que o sábio age
Pelo não-agir.
E ensina sem falar.
Aceita tudo que lhe acontece.
Produz tudo e não fica com nada.
O sábio tudo realiza - e nada considera seu.
Tudo faz - e não se apega à sua obra.
Não se prende aos frutos da sua atividade.
Termina a sua obra
E está sempre no princípio.
E por isto a sua obra prospera.

**Lao-Tsé**,
*Tao Te Ching*.

# Resumo

Tsukahara, Fernando Pirkel. **Organização Topológica de Objetos baseada em Algoritmos Genéticos Hierárquicos**. Goiânia, 2007. 131p. Dissertação de Mestrado. Escola de Engenharia Elétrica e de Computação, Universidade Federal de Goiás.

Objetos no mundo real são usualmente entidades complexas com muitos atributos, sendo representados como instâncias de dados n-dimensionais. Quando seres humanos (e máquinas) necessitam procurar em imensas bases de dados multidimensionais, a ordenação (e sua manutenção) facilita o processo de busca. Apesar do fato de que a ordenação unidimensional tradicional é um caso da ordenação n-dimensional topológica, a Ciência da Computação tem estado historicamente interessada em ordenação unidimensional apenas, ou seja, na ordenação baseada em um único atributo-chave. Mapas Auto-Organizáveis de Kohonen (SOM), típica abordagem da Computação Flexível para lidar com ordenação topológica, não funcionam bem para esta tarefa também. Muitas aplicações (tais como bibliotecas virtuais, comércio eletrônico, mecanismos de busca no ambiente web etc), contudo, devem apresentar muitos objetos n-dimensionais específicos em telas bidimensionais, onde os usuários deveriam encontrar objetos desejados com eficiência. No contexto de métodos de Escalamento de Dados Multidimensionais, este problema foi introduzido como "matrizes de proximidade" com densidade igual a 100%, e Algoritmos Genéticos surgiram como a melhor solução entre vários métodos. Aqui, é proposta uma alternativa baseada em hierarquia de Algoritmos Genéticos com meta adaptação. Resultados empíricos mostram que o sistema proposto é mais rápido para convergir em grandes espaços de busca, comparado à versão não hierárquica.

**Palavras–chave**

Hierarquia de Algoritmos Genéticos, Ordenação Topológica N-Dimensional, Meta Adaptação

# Abstract

Tsukahara, Fernando Pirkel. **Topological Ordering of Objects Based On Hierarchical Genetic Algorithms**. Goiânia, 2007. 131p. MSc. Dissertation. Escola de Engenharia Elétrica e de Computação, Universidade Federal de Goiás.

Objects in real world are usually complex entities with many attributes, being represented as instances of $n$-dimensional data. When human beings (and machines) need to search huge multidimensional datasets, sorting (and its maintenance) ease the search process. Despite the fact that traditional sorting is just a case of topological $n$-dimensional ordering, Computer Science has been historically interested on 1-dimensional sorting only, that is, on sorting based on a single key attribute. On the other hand, Kohonen's Self-Organizing Maps (SOM), Soft Computing's typical approach to deal with topological ordering, do not work properly for this task either. Many applications (such as virtual libraries, E-commerce catalogues, web search engines etc), however, must present many specific $n$-dimensional objects in 2-dimensional displays where users should find desired objects with efficiency. In the context of Multidimensional Data Scaling methods, this problem has been introduced as "proximity grids" with 100% density, and Genetic Algoritms has emerged as the best solution among several methods. In this work, it is proposed an alternative based on hierarchical genetic algorithm with meta adaptation. Empirical results have shown that the proposed system is faster to converge compared to the non-hierarchical version.

**Keywords**

Hierarchical Genetic Algorithms, Topological N-Dimensional Ordering, Meta Adaptation

# Sumário

# Lista de Figuras

# Lista de Tabelas

# Lista de Códigos de Programas

CAPÍTULO 1

# Introdução

## 1.1 Visão Geral

O presente trabalho trata de algoritmos de ordenação e visualização de conjuntos específicos de objetos (pertencentes a uma mesma classe), que compartilham um ou mais atributos comuns. Para desempenhar tal tarefa, apresentamos e validamos empiricamente um sistema baseado em hierarquia de algoritmos genéticos. O Sistema Proposto organiza topologicamente objetos específicos (instâncias de dados multidimensionais), com vistas na facilitação de processos posteriores de busca. Essa situação é tipicamente encontrada em catálogos eletrônicos de lojas virtuais (Comércio Eletrônico) [Randolph 2002] e cenários onde é necessária a busca de objetos específicos (bibliotecas virtuais e resultados de mecanismos de busca da internet, por exemplo). Ressaltam-se, ainda, as características únicas do problema sob o ponto de vista de Computação Flexível (Inteligência Artificial) e Ciência da Computação.

No contexto de Computação Flexível, os Mapas Auto-Organizáveis de Kohonen [Kohonen 1989] são a abordagem típica quando é necessária a ordenação topológica de dados multidimensionais. Entretanto existem deficiências na organização topológica de um conjunto de objetos específicos realizada por Mapas Auto-Organizáveis, pois os protótipos construídos durante o treinamento não preservam os objetos iniciais, apenas modelam estatisticamente as informações dos objetos (compondo protótipos a partir das características de tais objetos). Por exemplo, se um mapa de Kohonen for empregado para organizar um conjunto específico de cores (representados tridimensionalmente pelas componentes vermelho, verde e azul), usualmente termina com cores, registradas em seus protótipos, diferentes daquelas usadas no processo de treinamento.

A principal meta do presente trabalho é organizar topologicamente um conjunto específico de objetos pré-existentes enquanto os protótipos gerados pelos Mapas Auto-Organizáveis deveriam ser exatamente os objetos espe-

cíficos utilizados no treinamento. Neste sentido, mapas de Kohonen foram, de grande motivação para desenvolver uma alternativa adequada ao problema de interesse.

Historicamente, a Ciência da Computação interessa-se principalmente em ordenação unidimensional, ou seja, ordenação baseada em um único atributo-chave, tais como, *quicksort* [Hoare 1962], *shellsort* [Shell 1959] e *heapsort* [Williams 1964]. Mesmo quando $k$ atributos-chave são empregados na ordenação somente um dos atributos-chave é utilizado por vez obedecendo uma ordem de prioridade. Ordenação é necessária (e justificada) com vistas a consultas futuras em dados armazenados. A manutenção de uma estrutura ordenada de dados é vital para propiciar buscas eficientes e muitas estruturas diferentes têm sido propostas com esse objetivo, como árvores simétricas binárias: b-trees [Bayer 1971], por exemplo.

O presente trabalho trata de uma versão generalizada da ordenação tradicional de única dimensão. Um vetor ordenado na ordem ascendente ou descendente, pode ser considarada uma permutação topológica, onde as distâncias (dissimilaridades) entre os objetos vizinhos estão sendo minimizadas (a Figura 1.1 exibe um exemplo desse argumento). A idéia de ordenação aplicando como meta a minimização das distâncias entre objetos de um vetor, foi expandida ao escopo de objetos multidimensionais sendo ordenados numa matriz bidimensinal.

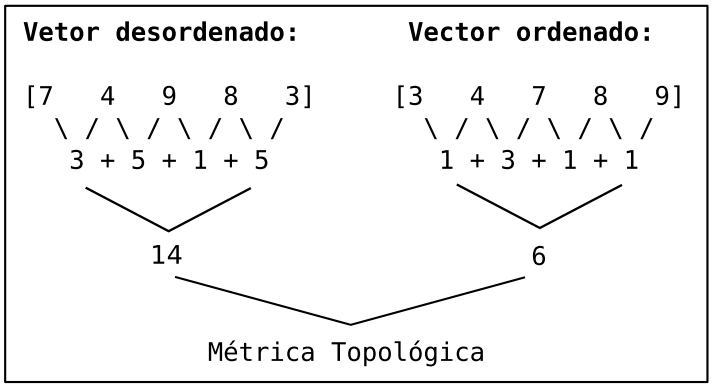


**Figura 1.1:** *Métrica topológica na ordenação de vetores.*

## 1.2 Justificativa

A ordenação de dados ou objetos participa do cotidiano de todas as pessoas de forma direta ou indireta. Os produtos da loja estão *ordenados* por marca, os livros da prateleira estão *ordenados* por editora, os CDs e

DVDs estão *ordenados* por estilo musical, os nomes dos alunos na pauta de frequência estão *ordenados* por *ordem* alfabética, o relatório de vendas do mês foi emitido por *ordem* de data da transação comercial e muitos outros exemplos estão à nossa volta, mesmo quando imperceptíveis. Nos exemplos citados, a ordem é aplicável a objetos (produtos, livros, CDs) ou dados (pauta de frequência, relatório de vendas) e referem-se a itens pertencentes à mesma classe, compartilhando características comuns.

No cenário estudado neste trabalho, um detalhe importante é o critério de ordenação, pois é comum a adoção de um único critério (dimensão, aspecto, variável) para ordenar os conjuntos de objetos. A marca do produto, a editora do livro, o estilo musical do CD, o alfabeto e a data no relatório são exemplos de possíveis critérios unidimensionais para ordenação de objetos. Porém, cada objeto é definido multidimensionalmente, ou seja, possui diversas características ou atributos necessários à sua descrição. A multidimensionalidade torna-se mais nítida se citarmos, por exemplo, aparelhos de telefonia celular. Quantas características descrevem um aparelho de telefonia celular? Eis algumas dimensões importantes: modelo, fabricante, *design*, cor, operadora, valor, tempo de duração da bateria, presença de antena externa, tamanho da agenda, despertador, câmera fotográfica, resolução da câmera fotográfica, rádio FM, capacidade de executar arquivos MP3, expansão da memória do celular via cartões de memória, jogos, além de, obviamente, executar ligações telefônicas. Quando um consumidor deseja adquirir um desses aparelhos celulares, diversos aspectos serão considerados, pois o consumidor irá até a loja para avaliá-los e compará-los, segundo vários desses critérios (ou até mesmo todos). A tarefa de comparar objetos semelhantes torna-se particularmente dispendiosa, quando os objetos de comparação apresentam combinações complexas entre suas características, como no caso dos aparelhos celulares.

A multidimensionalidade de características dos aparelhos celulares limitam a capacidade humana em comparar todos os atributos simultaneamente, executando um julgamento sobre qual aparelho oferece a melhor relação custo-benefício. Algumas vezes, tal procedimento requer a ajuda de especialistas para a decisão na escolha de um aparelho adequado ao perfil do consumidor. Portanto, é inegável a importante contribuição de uma forma de apresentação de objetos (a serem comparados) que considere **todas** as características multidimensionais intrínsecas à representação de tais objetos.

As comparações entre objetos multidimensionais podem ser simplificadas de forma mais racional caso sofram uma redução de dimensionalidade.

Para clarificar essa redução de dimensionalidade, cabe lembrar o emprego da paleta de cores de softwares gráficos, onde as três dimensões de cada cor estão organizadas (reduzidas) visualmente em duas dimensões. Os objetos ajustados à visualização em duas dimensões necessitam de ordenação topológica (o Capítulo 2 discute e define o termo "topologia" [1] no escopo deste trabalho), a fim de obter o efeito da redução. A cor é determinada por suas quantidades de pigmentos vermelho, verde e azul, assumindo valores entre 0 e 255. A ordenação topológica de cores impõe que cores semelhantes estejam fisicamente próximas, ou seja, as cores cujos atributos possuem valores próximos serão vizinhas. Ao observar a cor vermelha na paleta de cores do software gráfico (Figura 1.2), é possível localizar diretamente, em sua vizinhança, os tons mais próximos, como o vermelho-escuro e o vermelho-claro, além dos vizinhos, rosa, amarelo e alaranjado.

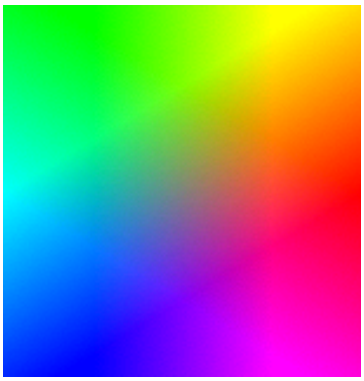

**Figura 1.2:** *Exemplo de paleta de cores.*

A redução de dimensionalidade da paleta de cores é um exemplo simples e direto de ordenação topológica presente no quotidiano do usuário de software de Computação Gráfica. Tal redução é aplicável a qualquer conjunto de objetos e beneficia o usuário de modo contundente.

Ao aplicar a redução de dimensionalidade e preservação topológica no exemplo dos celulares, teríamos os aparelhos com características semelhantes sendo vizinhos no mapa e estariam reunidos numa região específica, enquanto os aparelhos com características mais dissimilares estariam afastados. Ao reduzir as $k$ características dos aparelhos celulares num mapa de duas dimensões, a filtragem visual oferecida pelo mapa focaliza-se em regiões de interesse, descartando rapidamente regiões que não apresentem celulares interessantes ao comprador.

[1] Na Engenharia está relacionado à disposição lógico de objetos ou elementos. Na matemática, é a área em que se estudam os espaços topológicos.

A construção de permutações topológicas abre um leque infinito de possibilidades de aplicação. Se aplicarmos a catálogos eletrônicos de produtos, o consumidor pode localizar a área de interesse no mapa e focar nos produtos daquela região. Ele gastará menos tempo para escolher produtos com as características desejadas. Ao levar-se em consideração áreas promissoras como o comércio eletrônico, podemos citar o trabalho de [Moraes 2006], onde o catálogo eletrônico é baseado numa rede neural do tipo SOM, tendo como principal característica a preservação topológica entre elementos. A pesquisa difere das intenções do trabalho citado, pois a rede neural SOM busca a criação dos melhores prótotipos, a partir de um conjunto de treinamento e não tem a preocupação em alocar todos os espaços do mapa, enquanto o objetivo deste trabalho é encontrar permutações com preservação topológica a partir de um conjunto finito de objetos, levando em consideração a possível natureza multidimensional dos mesmos.

## 1.3 Problema e Hipóteses

O problema original de ordenação topológica de objetos foi definido em [Basalaj 2001], como "Matriz de Proximidade", utilizando algoritmos MDS (Escalamento Multidimensional) e Algoritmos Genéticos para buscas em coleções de imagens. Na construção de matrizes de proximidade, os resultados apontam algoritmos genéticos como melhor solução para matrizes totalmente preenchidas, ou seja, com objetos em todas as posições.

A geração de matrizes com preservação topológica é tratada como um problema de permutação, tanto em [Basalaj 2001] quanto nesta pesquisa. No entanto problemas de permutação, usualmente, resultam em explosões combinatoriais. A complexidade aumenta exponencialmente, de acordo com a quantidade de objetos a serem ordenados. Por exemplo, para permutações numa matriz de tamanho 5x5, teríamos aproximadamente $1,55$ x $10^{25}$ $(25!)$ possibilidades de ordenação, para resolver o problema, considerando a inexistência de objetos repetidos. Neste mesmo contexto, uma matriz 10x10 tem o número de possibilidades elevado para $9,33$ x $10^{157}$ $(100!)$.

O tamanho do espaço de busca afeta diretamente o desempenho do algoritmo, pois reconhecidamente a busca num espaço com $10^{157}$ possibilidades é uma tarefa computacional extremamente dispendiosa.

O problema desta pesquisa é: como acelerar a busca de mapas topológicos de objetos multidimensionais baseado em Algoritmos Genéticos?

A hipótese principal deste trabalho é:

- A presença de hierarquia em Algoritmos Genéticos acelera a busca de mapas topológicos de objetos multidimensionais.

Quanto à hipótese secundária temos:

- A presença de hierarquia em Algoritmos Genéticos melhora qualitativamente os mapas topológicos de objetos com múltiplas dimensões.

## 1.4 Estrutura dos Capítulos

O Capítulo 2 apresenta a Fundamentação Teórica sobre **Algoritmos Genéticos** e **Organização Topológica**. Dentro de Computação Evolucionária, abordamos a história de Algoritmos Genéticos, seus conceitos principais e operadores. Sobre Organização Topológica, definimos o termo topologia no escopo deste trabalho, além de apresentar brevemente Mapas Auto-Organizáveis de Kohonen e Matriz de Proximidade.

O Capítulo 3 dedica-se à descrição do **Sistema Proposto**. Detalhamos o uso de hierarquia de Algoritmos Genéticos, a função de avaliação e os novos operadores de variação e seleção. Com base nessa estrutura, o sistema atende à sua proposta de organizar topologicamente objetos multidimensionais.

O Capítulo 4 apresenta os **Experimentos e Resultados** da validação empírica. Expomos a origem das amostras, material e instrumentos utilizados, bem como os resultados obtidos. Analisamos os dados tanto do ponto de vista da Estatística Descritiva quanto da Estatística Inferencial, para os cenários de números inteiros e cores.

No Capítulo 5, a **Conclusão** salienta os resultados obtidos e a corroboração da hipótese principal para grandes espaços de busca. Ressaltamos ainda a rejeição da hipótese secundária, principais contribuições e trabalhos futuros.

CAPÍTULO 2

# Fundamentação Teórica

O presente Capítulo apresenta a fundamentação teórica do trabalho. Algoritmo Genéticos e Organização Topológica formam a base teórica do sistema proposto (Capítulo 3), para busca de permutações topológicas, utilizando hierarquia de Algoritmos Genéticos.

## 2.1 Algoritmos Genéticos

### 2.1.1 Visão Geral

O advento da computação eletrônica tem provado ser o desenvolvimento mais revolucionário da história da ciência e tecnologia. Os objetivos de criar inteligência e vida artificiais remetem-nos de volta ao início da era da computação. Os pioneiros cientistas da computação, como Alan Turing e John von Neumann[1], foram motivados pela visão de criar programas inteligentes com habilidades de auto-replicação e adaptação ao ambiente. Tais cientistas estavam interessados em biologia e psicologia, tanto quanto em eletrônica, e observaram sistemas da natureza, como guias metafóricos, para alcançar os objetivos desejados.

Desde o início dos anos 80, ressurgiu o interesse da comunidade científica em pesquisar temas com motivação biológica, como Redes Neurais e Computação Evolucionária. Na área de Computação Evolucionária, os algoritmos formam uma família de modelos computacionais, cujo funcionamento metaforiza a teoria da evolução natural de Charles Darwin [Darwin 1906]. Dentre os algoritmos evolucionários destacamos a programação genética, estratégias evolucionárias, programação evolucionária e os algoritmos genéticos [Whitley 1994].

[1]Os cientistas utilizaram computação em tarefas como calcular rotas de mísseis e máquinas para decriptografia de comunicações durante a II Guerra Mundial

Os Algoritmos Genéticos são a forma mais popular dos Algoritmos Evolucionários e têm sido aplicados com sucesso nas mais diferentes áreas, como ciência, telecomunicações, engenharia, física, indústria e comércio. Tais algoritmos otimizam, por exemplo, síntese de circuitos analógicos, agenda de plantão médico e projetos de redes óticas, entre outros [Melanie 1998].

### 2.1.2 Breve Histórico

Nas décadas de 1950 e 1960 muitos cientistas estudaram independentemente sistemas evolucionários baseados na idéia de utilização da evolução para resolver problemas de otimização na engenharia. A idéia central dos sistemas era evoluir populações de soluções candidatas, utilizando operadores inspirados na variação e seleção naturais.

O cientista alemão Ingo Rechenberg introduziu em meados de 1960 e começo de 1970 as "estratégias evolucionárias", método utilizado na otimização de parâmetros para dispositivos como aerofólios [Rechenberg 1994], a primeira aplicação registrada na área de evolução artificial. A mesma idéia desenvolveu-se posteriormente por Schwefel [Schwefel 1995] em sua tese de doutorado entitulada "Otimização numérica com modelos computacionais", de 1974. A pesquisa de estratégias evolucionárias permaneceu ativa, independentemente da área de algoritmos genéticos.

Em 1966, Fogel, Owens, and Walsh (1966) [Fogel 1995] desenvolveram a "programação evolucionária", onde soluções candidatas, representadas por máquinas finitas de estados, evoluem com mutação randômica entre diagramas de transições de estados e seleção das soluções mais adequadas. Outros pesquisadores desenvolveram algoritmos inspirados na evolução para aprendizado de máquinas. Box (1957), Friedman (1959), Bledsoe (1961), Bremermann (1962), e Reed, Toombs, e Baricelli (1967) [Rawlins 1991] desenvolveram trabalhos nesta área, mas receberam pouca atenção da comunidade científica, comparado as estratégias evolucionárias, programação evolucionária e algoritmos genéticos.

John Holland criou os Algoritmos Genéticos, em meados de 1960, e continuou o desenvolvimento da técnica na Universidade de Michigan, nas décadas de 1960 e 1970. Em contraste com as estratégias evolucionárias e a programação evolucionária, o objetivo de Holland não foi projetar um algoritmo para resolver problemas específicos, mas formalizar o fenômeno da adaptação, como ocorre na natureza, desenvolvendo estes mecanismos em sistemas computacionais. A obra de Holland de 1975 [Holland 1975]

(Adaptação em Sistemas Naturais e Artificiais) apresentou os Algoritmos Genéticos na forma de abstração da evolução biológica e apresentou um *framework* teórico adaptativo.

O Algoritmo Genético tradicional de Holland evolui populações de indivíduos codificados em representação binária, usando um método de "seleção natural" aliado a operadores inspirados biologicamente: cruzamento (reprodução) e mutação. Cada cromossomo possui genes (bits) e cada gene representa instâncias particulares de alelos (por exemplo, 0 ou 1). O operador de seleção dos cromossomos permite a reprodução baseado na adaptação de cada cromossomo, e em média, produz cromossomos mais adaptados em relação à geração anterior. O cruzamento efetua trocas de subpartes entre dois cromossomos e muta randomicamente o valor de alguns genes. A introdução de um algoritmo com operadores de cruzamento (crossover) e mutação representou a evolução da Computação Evolucionária.

### 2.1.3 Conceitos Básicos

A natureza têm sido uma ótima fonte de inspiração para heurísticas consagradas, como Resfriamento Simulado, Redes Neurais e Colônia de Formigas. Cada uma dessas heurísticas simula o funcionamento específico de um sistema da natureza. Os Algoritmos Genéticos também possuem fundamentação biológica e a grande inspiração é o modelo da Evolução Natural de Darwin [Darwin 1906]: a natureza seleciona os indivíduos mais adaptados ao meio, para reproduzirem, enquanto os menos adaptados morrem. Os sobreviventes produzirão descendentes similares aos pais, assim cada nova geração contém indivíduos com características paternas, porém mais aptos ao meio que a geração anterior. Ocasionalmente acontecem mutações aleatórias, e embora, em sua maioria, causem uma morte rápida para o indivíduo, algumas mutações levam a espécimes bem-sucedidos, evitando a estagnação em mínimos ou máximos locais.

Algoritmos Genéticos podem ser definidos de forma ampla, como um modelo de busca e otimização inspirados na evolução Darwiana e implementados na forma de uma **simulação computacional**[2].

Um método de busca analisa várias soluções e utiliza as informações obtidas, durante o processo, para encontrar soluções melhores. No "espaço de busca", estão disponíveis todas as possíveis soluções para o problema. A

[2]Aplicação de técnicas matemáticas em computadores com o propósito de imitar um processo ou operação do mundo real.

otimização tenta encontrar o ponto máximo (ou mínimo) global para a função objetivo, no espaço de busca em questão. A Figura 2.1 exibe um exemplo de um espaço de busca simples, onde os máximos e mínimos locais são identificáveis através de inspeção visual. Outro espaço de busca com vários mínimos e máximos locais é mostrado na Figura 2.2.

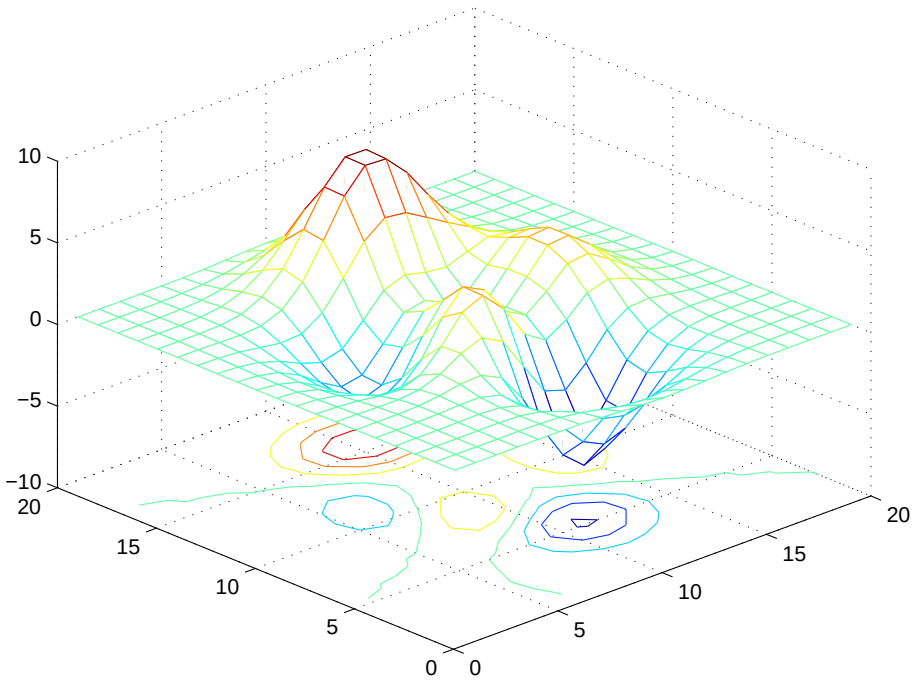


**Figura 2.1:** *Exemplo de um espaço de busca.*

Os algoritmos genéticos definem uma linguagem própria para suas estruturas de dados computacionais. As soluções do problema codificadas recebem o nome de **cromossomos** e representam os indivíduos que farão parte da busca. Os cromossomos são constituídos de **genes**. Os genes codificam as características do indivíduo e os valores que os genes podem assumir são denominados de **alelos**.

A simulação inicia com a população inicial de soluções candidatas, ou seja, cada indivíduo da população codifica uma solução potencial para o problema. Os indivíduos (possíveis soluções) evoluirão através de operadores genéticos de **seleção** e **variação** (cruzamento e mutação). No caso de uma população com 100 indivíduos, teremos inicialmente 100 possíveis soluções, evoluindo, em paralelo, ao longo da execução do AG, justificando então a busca paralela.

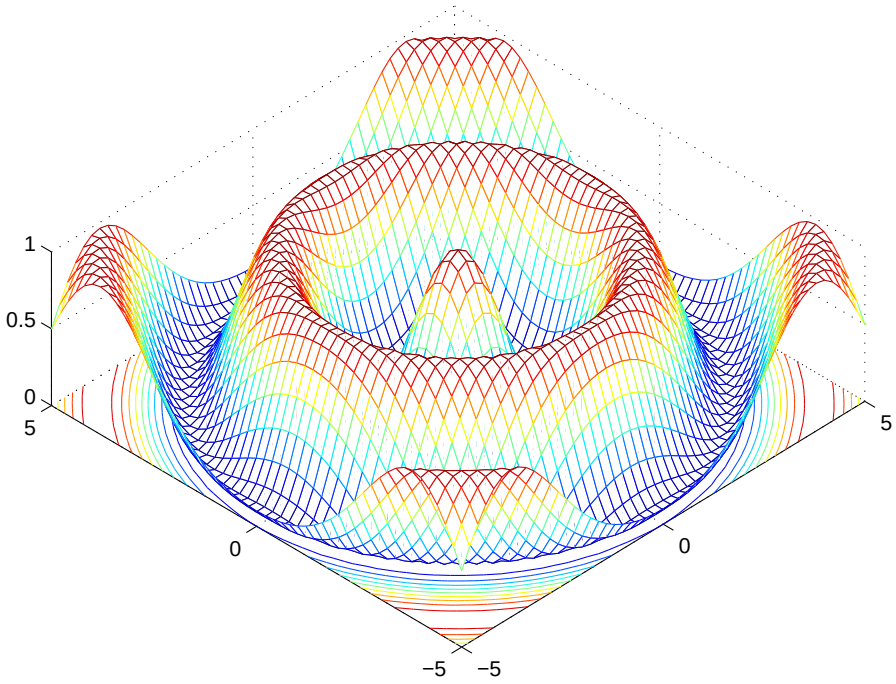


**Figura 2.2:** *Exemplo de um espaço de busca com vários mínimos e máximos locais.*

Se cada indivíduo da população representa uma solução candidata, então é necessário avaliar quão boa é a solução para o domínio do problema. A função de "avaliação", ou função de "*fitness*" é a responsável por avaliar a qualidade da solução representada por cada indivíduo. Após avaliar cada indivíduo (possível solução) da população, a qualidade deste indivíduo é quantificada e representará a chance probabilística de ser escolhido para as operações de variação (cruzamento e mutação). Portanto os melhores indivíduos terão alocadas as melhores chances de reprodução.

As operações de variação são responsáveis por descobrir novas soluções e selecionar quais indivíduos sobreviverão, a fim de compor a base para futuras explorações [Fogel e Michalewics 1999]. A principal operação de variação é o cruzamento entre dois indivíduos, cujos materiais genéticos serão reorganizados. Usualmente resulta-se desta operação, 1 ou 2 novos indivíduos. Após o nascimento de um novo indivíduo, existe a probabilidade de mutação em alguns genes do indivíduo resultante. Cada gene do indivíduo recém-gerado é submetido a um sorteio que obedece a "taxa de mutação" de forma probabilística. O gene sorteado para a operação de mutação sofrerá uma pequena alteração em seu material genético. A operação de mutação caracteriza um ruído aleatório no código genético, prevenindo a estagnação do AG em mí-

nimos locais (ou máximo, dependendo do problema). Os genes codificam as características de um indivíduo de forma n-dimensional, não existindo um limite com número máximo de características que um indivíduo pode codificar.

### 2.1.4 Algoritmo Genético padrão

O funcionamento de um algoritmo genético pode ser representado pelo pseudocódigo a seguir:

```
1. Geração da população inicial (usualmente de forma
aleatória).

2. Enquanto o critério de parada não for alcançado,
repita os itens (a), (b), (c), (d) e (e).

(a) Avaliação: cada indivíduo é avaliado e recebe uma
nota de aptidão (adequação ou fitness) ao domínio do
problema.

(b) Seleção: os indivíduos com as melhores avaliações,
têm as melhores chances alocadas para a reprodução.

(c) Reprodução: o casal de indivíduos selecionado
através da aptidão, recombinam-se para gerar
descendentes.

(d) Mutação: alguns genes dos indivíduos recém-nascidos
podem sofrer pequenas alterações obedecendo a taxa de
mutação (fixa ou variável).

(e) Elitismo: a população antiga (pais) é substituída
pela população nova (descendentes) de acordo com o
critério de elitismo. Os melhores indivíduos da
população antiga são inseridos diretamente na nova
população.

3. Retorne a melhor solução encontrada.
```

O pseudocódigo clarifica a metáfora da evolução natural [Darwin 1906]. Os indivíduos (soluções) mais aptos (melhores notas de apti-

dão) a sobreviver no ambiente (espaço de busca) reproduzem (operadores de variação: reprodução e mutação), gerando descendentes mais adaptados e eliminando naturalmente os menos adaptados (elitismo) por meio da competição entre indivíduos (seleção natural).

### 2.1.5 Configuração Inicial

A abordagem clássica de inicialização de indivíduos na população inicial do algoritmo genético, é a seleção aleatória de pontos no espaço de busca (vide Figura 2.3), pois normalmente não existem informações iniciais sobre o problema. No entanto, em alguns casos é possível fornecer dicas ao algoritmo genético com o objetivo de agilizar a busca, determinando, por exemplo, restrições a certas áreas do espaço de busca ou informar regiões de interesse, como mostra a Figura 2.4.

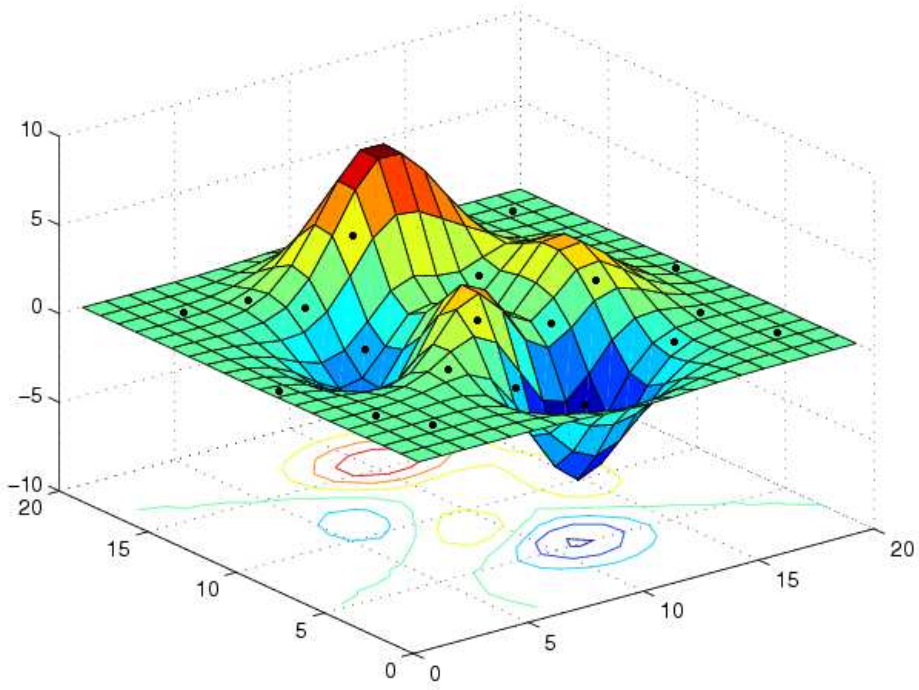


**Figura 2.3:** *Inicialização aleatória de indivíduos no espaço de busca.*

Os indivíduos iniciais da Figura 2.4 têm origem a partir de conhecimento prévio do espaço de busca, com possíveis origens em outras heurísticas como Resfriamento Simulado (Simulated Annealing), Busca Tabu (Tabu Search) ou mesmo execuções anteriores do próprio algoritmo genético.

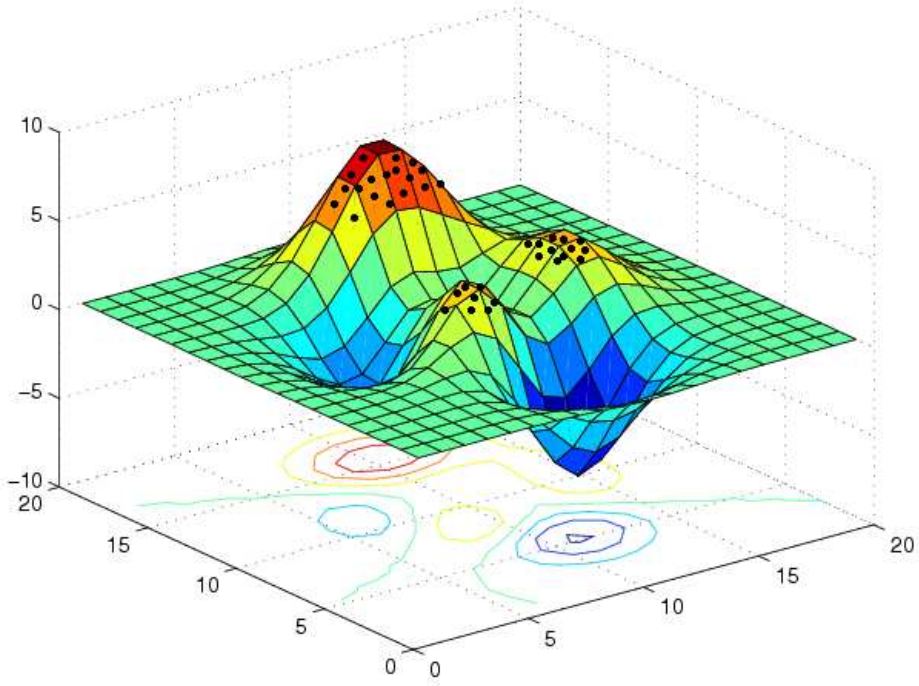


**Figura 2.4:** *Inicialização de indivíduos no espaço de busca a partir de pontos de interesse.*

### 2.1.6 Representação dos indivíduos

As soluções de um determinado problema são codificadas em estruturas de dados computacionais, para que o algoritmo genético possa manipulá-las. Segundo o autor [Fogel e Michalewics 1999]: representação de indivíduos é mapeamento do espaço de possíveis soluções para o espaço de soluções codificadas, dentro de alguma estrutura de dados.

Neste contexto, é necessário representar bem os indivíduos, permitindo uma ligação funcional entre pais e filhos, após a variação. Uma má representação pode levar a busca praticamente aleatória, portanto esta fase merece especial atenção.

Entre os principais tipos de representação destacamos: binária, binária codificando real, inteiros, real e estruturas (vetores, matrizes e listas). Representações binárias, por exemplo, foram utilizadas no algoritmo genético original de Holland e ainda são amplamente usadas nos dias atuais.

Segundo [Fogel e Ghozeil 1997], no artigo "A note on Representations and Variations operators", não existem representações que levam vantagens sobre outras, mas a representação escolhida deve ser aquela que pareça mais intuitiva para o problema. Tomemos como exemplo o problema do caixeiro

viajante: com uma viagem ou *tour*[3] de 10 cidades, a representação mais natural é um vetor com os números de cada cidade e o índice do vetor indicando a ordem em que serão percorridas.

```
[1 9 3 4 7 2 5 8 6 10]
```

Uma nova viagem significa uma alteração de posições no vetor:

```
[1 3 9 7 4 2 5 6 8 10]
```

Problemas de seleção são usualmente representados de forma binária. Neste caso, a palavra seleção indica um subconjunto de itens a serem escolhidos ou a ausência/presença de determinada característica e qualquer outro problema envolvendo duas possibilidades determinantes no domínio do problema. A representação binária também pode ser utilizada para codificar valores reais, nesse caso o algoritmo genético precisará de uma função de "decodificação" para realizar a operação inversa. Representações utilizando valores reais costumam ser utilizadas em maximização ou minimização de funções matemáticas, com parâmetros contínuos, como temperatura ou pressão.

Ao analisar os diferentes tipos de representações (ou criar novas) para cada problema, é necessário optar pela representação que também seja intuitiva para os operadores de variação, pois certos tipos de representação podem levar os operadores a criar indivíduos inválidos ou que necessitem de correção. Se uma viagem (*tour*) do caixeiro viajante fosse representada de forma binária e sofresse um cruzamento uniforme tradicional, fatalmente nasceriam indivíduos onde a viagem completa passaria por uma mesma cidade mais de uma vez.

### 2.1.7 Função de Avaliação

A função de Avaliação (aptidão ou *fitness*) determina a qualidade relativa das soluções e a efetividade da busca como um todo. É responsável pelo elo entre o algoritmo genético e o problema, podendo ser de difícil definição, quando existem fatores de restrições e penalidades internas ou uma combinação de múltiplos objetivos numa mesma função.

Em algoritmos genéticos, avaliar a qualidade das soluções numa população é geralmente a operação mais demorada. Em alguns casos a função

[3]Uma viagem completa percorrida pelo caixeiro viajante, passando por todas as cidades uma única vez com retorno à cidade de origem.

de avaliação pode ser acelerada, utilizando um valor estimado ou um valor relativo.

A medida de avaliação é produzida a partir da função objetivo do problema. A função objetivo também mensura como os indivíduos têm se desempenhado, no domínio do problema, no entanto ela produz uma medida "crua" isolada. Tal resultado deve, portanto, ser transformado numa medida de avaliação relativa, pois num problema de minimização, por exemplo, a função objetivo pode produzir valores negativos.

Na Equação 2-1, $f$ é a função objetivo, e $g$ transforma o valor da função objetivo em um número não-negativo e $F$ é a avaliação relativa resultante.

$$F(x) = g(f(x)) \tag{2-1}$$

Ao longo do trabalho, referiremos diretamente à medida de avaliação com a função objetivo já normalizada.

### 2.1.8 Seleção

Seleção é o processo determinante do número de vezes, ou tentativas, que um certo indivíduo será selecionado para a variação e, portanto, o número de descendentes diretos a serem produzidos.

A seleção da roleta é um procedimento usual de seleção, baseado na aptidão dos indivíduos. Esta técnica simula um jogo de azar que usa uma roleta para sortear números, como uma "roda da fortuna". A roleta é um disco dividido em setores, com formato de fatias de pizza. O indicador estático aponta para o setor sorteado quando o disco pára de girar. Em algoritmos genéticos, cada setor do disco representa um cromossomo da geração pai e a largura de cada setor representa a adequação relativa de um dado cromossomo. A roleta simulada gera um número aleatório, correspondendo a frações da soma das adequações dos cromossomos da população, desta maneira, valores maiores de adequação possuem mais chances de serem escolhidos.

Uma outra estratégia popular é a seleção por torneio. O AG escolhe aleatoriamente dois indivíduos e o melhor deles é selecionado para participar da variação.

A seleção pode levar em consideração, o número de vezes que um mesmo indivíduo poderá participar do processo de variação. Numa estratégia mais restritiva, o indivíduo sorteado para a variação seria retirado da população de candidatos a genitores, enquanto numa estratégia menos restritiva o

indivíduo sorteado seria utilizado na variação e teria suas chances realocadas para a próxima iteração de variação.

Independente do procedimento de seleção a ser utilizado, o principal conceito deste operador é atuar como o filtro através do qual o algoritmo determina a base para compor a geração seguinte.

### 2.1.9 Variação

Os algoritmos genéticos dependem diretamente dos operadores de variação para gerar a nova população, ou seja, é o ponto chave para evoluir as soluções correntes. O **cruzamento** e a **mutação** estão presentes na maioria dos algoritmos genéticos e promovem um debate sobre as vantagens e desvantagens de cada um.

#### Cruzamento

O cruzamento (ou *crossover*) efetua a operação de recombinação entre os genes do casal selecionado. O cruzamento é dependente da escolha de uma boa representação, já que os descendentes devem possuir características semelhantes às dos progenitores em sua carga genética, após a variação. Os operadores típicos de cruzamento são: cruzamento de um ponto, cruzamento de múltiplos pontos e cruzamento uniforme.

O cruzamento de ponto único, combina um par de cromossomos selecionando aleatoriamente o local onde os cromossomos serão divididos e efetuarão a troca do material genético. O exemplo a seguir apresenta um cruzamento de ponto único com ponto de corte na posição 4:

```
Pai 1    [1 0 0 1 | 1 0 0 0 1 1 0 1 1]
                  X
Pai 2    [1 1 1 0 | 1 1 0 1 1 0 1 0 1]

          - - - - 4 - - - - - - - - -

Filho 1 [1 0 0 1 | 1 1 0 1 1 0 1 0 1]

Filho 2 [1 1 1 0 | 1 0 0 0 1 1 0 1 1]
```

O cruzamento de múltiplos pontos sorteia um número $n$ de pontos, indicando onde ocorrerão os cortes no cromossomo afim de gerar os novos descendentes. O exemplo a seguir apresenta um cruzamento de múltiplos pontos com cortes nas posições 2, 7 e 9:

```
Pai 1    [1 0 | 0 1 1 0 0 | 0 1 | 1 0 1 1]
              X           X     X
Pai 2    [1 1 | 1 0 1 1 0 | 1 1 | 0 1 0 1]

          - - 2 - - - - - 7 - - 9 - - - -

Filho 1 [1 0 | 1 0 1 1 0 | 0 1 | 0 1 0 1]

Filho 2 [1 1 | 0 1 1 0 0 | 1 1 | 1 0 1 1]
```

Os cruzamentos de único e múltiplos pontos definem pontos de corte onde o cromossomo será dividido. O cruzamento uniforme generaliza esta idéia para todos os genes, transformando cada gene num potencial ponto de cruzamento. Uma máscara de cruzamento gerada aleatoriamente indica quais pais suprirão os descendentes com seu material genético. No exemplo a seguir, o filho 1 é produzido herdando o gene do pai 1 quando a máscara possui o bit ligado (1). Quando o bit da máscara for 0, o filho 1 herda o gene do pai 2. O filho 2 é gerado utilizando o inverso da máscara: se o bit da máscara for 1, herda o material genético do pai 2 e quando for 0 herda o gene do pai 1.

```
Pai 1    [1 0 1 1 0 0 0 1 1 1]

Pai 2    [0 0 0 1 1 1 1 0 0 0]

Máscara [0 0 1 1 0 0 1 1 0 0]

Filho 1 [0 0 1 1 1 1 0 1 0 0]

Filho 2 [1 0 0 1 0 0 1 0 1 1]
```

A representação escolhida é fundamental para determinar os operadores de variação. As representações com números inteiros ou reais seguem estratégias similares as descritas anteriormente, levando em consideração a

natureza da representação, utilizando por exemplo, a média entre dois genes para geração de descendentes. A estratégia é conhecida como cruzamento aritmético e, a exemplo das estratégias empregadas na representação binária, obedecem o sorteio de um ou vários genes para a troca de material genético. Após o cruzamento aritmético, cada descendente recebe a média matemática obtida a partir dos valores encontrados nos genes paternos previamente sorteados. No exemplo a seguir, os genes 2 e 8 são sorteados durante a variação:

```
Pai 1    [0.1  0.2  0.3  0.4  0.5  0.6  0.7  0.8  0.9]

Pai 2    [0.9  0.4  0.1  0.6  0.2  0.7  0.3  0.2  0.3]

           -    X    -    -    -    -    -    X    -

Filho 1 [0.1  0.3  0.3  0.4  0.5  0.6  0.7  0.5  0.9]

Filho 2 [0.9  0.3  0.1  0.6  0.2  0.7  0.3  0.5  0.3]
```

Os problemas de permutação formam uma classe especial e requerem operadores específicos de variação. O problema do caixeiro viajante, um clássico problema de permutação, terá convergência comprometida caso utilize os operadores descritos anteriormente, pois os filhos resultantes podem ser indivíduos inválidos no espaço de busca, conforme demonstrado no exemplo a seguir:

```
Pai 1    [ 9 3 4 7 | 6 2 5 8 1 10]

Pai 2    [10 8 2 4 | 9 7 3 1 6 5 ]

           - - - - |  - - - - - - -

Filho 1 [ 9 3 4 7 | 9 7 3 1 6 5 ]

Filho 2 [10 8 2 4 | 6 2 5 8 1 10]
```

A repetição de cidades na mesma viagem e a ausência de outras, torna a rota de viagem inválida. Uma nova operação para corrigir os recém-nascidos pode ser adotada, mas em geral as operações de comparação demandam mais tempo e processamento. Entre os operadores de variação para permutações

destacamos: cruzamento de primeira ordem, cruzamento parcialmente mapeado, cruzamento cíclico e recombinação de extremos [Eiben e Smith 2003].

### Mutação

A operação de mutação garante uma maior varredura do espaço de busca e evita que o algoritmo genético convirja prematuramente ou fique estagnado num mínimo (ou máximo) local. Os genes de cada cromossomo são submetidos a um sorteio obedecendo uma taxa de mutação. Se o gene for sorteado para mutar, o gene sorteado tem seu valor alterado. Se o cromossomo sorteado utiliza a representação binária, basta inverter o valor do bit: se o gene for "1" será mutado para "0" e vice-versa. No caso de representações com inteiro ou real, o valor do gene será alterado para um ponto próximo do ponto onde se encontra. Por exemplo, um gene com valor 9,35 pode ter sua mutação limitada para um ponto dentro da faixa 7,35 e 11,35, garantindo que o valor encontra-se na vizinhança, mas explorando locais ainda não visitados. No caso de problemas de permutação a mutação normalmente consiste em trocar a posição de dois genes, garantindo um ruído aleatório que continua gerando soluções viáveis.

### Discussão

Os debates acerca de qual estratégia é melhor ou mais necessária para a convergência de algoritmos genéticos, resultam em alguns pontos de ampla concordância [Eiben e Smith 2003]:

1. Depende do problema, mas em geral é recomendável utilizar as duas estratégias;
2. Ambas estratégias possuem papéis diferenciados;
3. Um algoritmo genético que possua somente mutação é possível, enquanto outro somente com cruzamento é inviável.

O cruzamento oferece ao algoritmo genético chances de efetuar saltos no espaço de busca para uma área "entre" a área dos progenitores, enquanto a mutação insere pequenos ruídos mantendo o filho numa região do espaço de busca próxima à área dos pais. Segundo [Eiben e Smith 2003], somente a mutação pode introduzir novas informações e alcançar um resultado ótimo, freqüentemente resultante de uma mutação de "sorte".

### 2.1.10 Elitismo

O elitismo é a operação de descarte dos indivíduos menos adaptados no contexto do ambiente em que estão inseridos. A operação é executada após as operações de variação, selecionando os melhores indivíduos, entre população nova e antiga.

A seleção da elite obedece taxas variáveis, de acordo com a escolha do projetista do algoritmo genético, podendo salvar apenas o melhor indivíduo da população antiga e compor o restante com os novo indivíduos ou fixar uma taxa para salvar os 50% mais adaptados da população antiga, somados aos 50% mais adaptados da população recém-gerada.

## 2.2 Organização Topológica

A presente Seção destina-se a definir o termo **topologia** e sua abordagem no escopo do trabalho.

### 2.2.1 Introdução

A topologia é uma extensão da geometria e inicialmente foi conhecida como "geometria de lugar" (do latim *geometriam situs*) e "análise de lugar" (do latim *analysis situs*). Observando as raízes gregas da palavra topologia temos *topos* = lugar e *logos* = estudo. O **lugar**, definido na etimologia da palavra, refere-se a investigação de um espaço para verificar o relacionamento dos elementos existentes nele [Munkres 1999].

Historicamente, a topologia teve sua investigação iniciada por Leonard Euler, físico e matemático suíço. Em 1736, Euler resolveu o problema das "Sete pontes de Königsberg" (figura 2.5). Euler transformou o problema num grafo e provou que não era possível executar um circuito cruzando as 7 pontes uma única vez cada, e retornar ao ponto de partida [Euler 1736]. Com base nessa demonstração publicou-se o artigo "Geometriam Situs" ou Geometria de lugar, formulando inicialmente a Teoria dos Grafos. Embora o termo topologia ainda não fosse usado, o problema foi resolvido de forma topológica, pois as informações chave para montar o grafo foram o número de pontes (vértices) e seus pontos finais (porções de terra representadas pelos nós). De forma genética, considerou-se os elementos do espaço e como estavam conectados estruturalmente, abstraindo o problema de forma matemática.

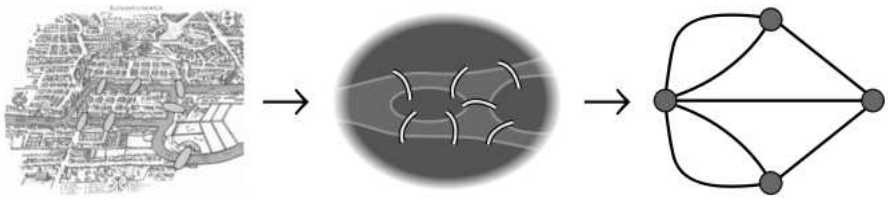

**Figura 2.5:** *As sete pontes de Königsberg*

A Figura 2.6 exibe o desenho original de Euler publicado no artigo [Euler 1736]. Euler transformou as porções de terra em nós, e interligou os nós através de vértices.

Oficialmente, o termo topologia foi introduzido em 1847, por Johann Benedict Listing, em sua obra *Vorstudien zur Topologie* (Estudos introdutórios em topologia). Na língua inglesa, a topologia foi definida em 1883, no periódico *Nature*, como uma forma de distinguir "geometria qualitativa da geome-

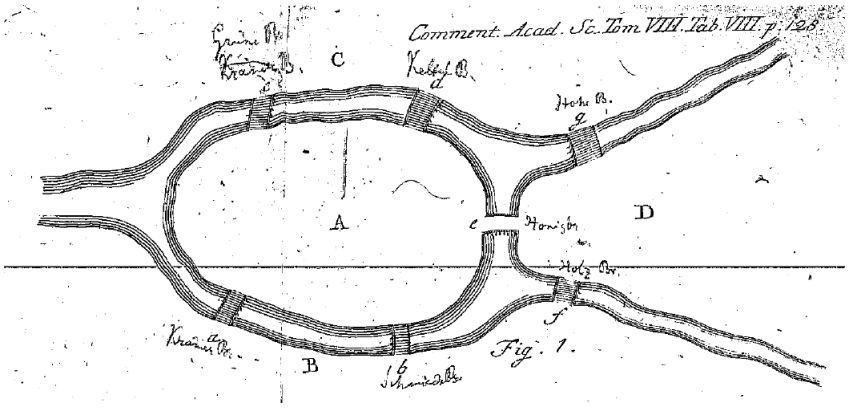


**Figura 2.6:** *Desenho original de Euler.*

tria ordinária, onde as relações quantitativas são tratadas prioritariamente" [Kelley 1975]

Esta Seção é reservada à **organização topológica** e, no contexto da Engenharia e Computação Flexível, tal organização refere-se a algum tipo de ordem ou hierarquia entre os elementos do espaço dito topológico. A próxima Subseção discute os mapas auto-organizáveis de Kohonen (SOM) e suas métricas para definir qualidade na organização topológica. Em seguida, apresentamos a matriz de proximidade.

### 2.2.2 Mapas Auto-Organizáveis

Os Mapas Auto-Organizáveis, do inglês Self-Organizing Maps (SOM), foram desenvolvidos na década de 80 pelo finlandês Teuvo Kohonen e constituem um dos principais modelos na área de **Redes Neurais Artificiais**.

Uma rede neural pode ser definida como unidades simples de processamento, conectadas de forma paralela e distribuída, com capacidade de armazenar conhecimento experimental [Haykin 2001]. Posteriormente, tal conhecimento armazenado, é disponibilizado para uso em sistemas que tirem proveito deste recurso.

A semelhança com a rede neural biológica acontece ao adquirir conhecimento a partir do ambiente (num processo de aprendizagem) e armazenar o conhecimento nas sinapses (forças de conexão entre neurônios, conhecidos como pesos sinápticos). A rede neural produz melhores resultados à medida que existam mais informações de entrada disponíveis no processo de aprendizado [Braga, Ludermir e Carvalho 2000]. O aprendizado da rede neural acontece através de um algoritmo de treinamento, onde ajusta-se continuamente os pesos das conexões, até que a rede obtenha estabilidade no conhecimento adquirido.

As Redes Neurais Artificiais têm sido utilizadas em diferentes áreas de conhecimento, seja em pesquisas científicas, tecnologia ou negócios, através de uma enorme gama de modelos e arquiteturas consagradas. Neste estudo, dedicaremo-nos na abordagem de um modelo específico de rede neural, conhecida por SOM (Self Organizing Maps), cujas características de aprendizado competitivo e não-supervisionado possuem inspiração no córtex cerebral [Kohonen 1989]. A principal propriedade deste modelo é a de se auto organizar, preservando a estrutura topológica de padrões apresentados.

O SOM possui duas camadas (entrada e saída) de neurônios artificiais que ajustam um conjunto de vetores de pesos para modelar uma aproximação com os dados de entrada (os neurônios de saída refletem o aprendizado obtido a partir das entradas do conjunto de treinamento). Cada um desses vetores de pesos está associado com um dos neurônios da camada de apresentação, disposta normalmente na forma de um mapa retangular. Se o mapa tiver a dimensionalidade do espaço desejado, a associação dos vetores de peso com os neurônios de saída pode ser interpretado como um mapeamento multidimensional para um mapa bidimensional. A projeção de um espaço não-linear de grande dimensão $R^D$ em um espaço regular $R^P$ de menor dimensão, normalmente unidimensional (P=1) ou bidimensional (P=2), é a principal característica do SOM. A formação em um mapa, de agrupamentos de dados com características semelhantes, facilita a visualização das relações e similaridades dos itens de dados [Haykin 2001].

Para apresentar a idéia de topologia e mapeamento de espaços multidimensionais em espaços bidimensionais, apresentamos na Figura 2.7 a ilustração de um grupo de pessoas a serem utilizadas no treinamento de uma rede neural do tipo SOM. O grupo de pessoas será descrito segundo os atributos sexo, idade e altura.

O problema em questão deseja obter agrupamentos, cujos integrantes tenham características semelhantes. A preservação das relações topológicas são as informações semelhantes mapeadas em neurônios próximos (vizinhos), ou no mesmo neurônio, caracterizando o agrupamento do espaço de entrada.

Durante o treinamento, os exemplos são apresentados à rede neural SOM, cujos vetores de peso modelam estatisticamente aproximações dos valores de entrada, usando competição não-supervisionada. Tal competição ocorre entre os neurônios da camada de saída e disputam entre si a representação da informação apresentada aos neurônios da camada de entrada. O neurônio vencedor, para a entrada apresentada, tem seu vetor de pesos alterado. Tal alteração sensibiliza também os pesos dos neurônios vizinhos. O raio de

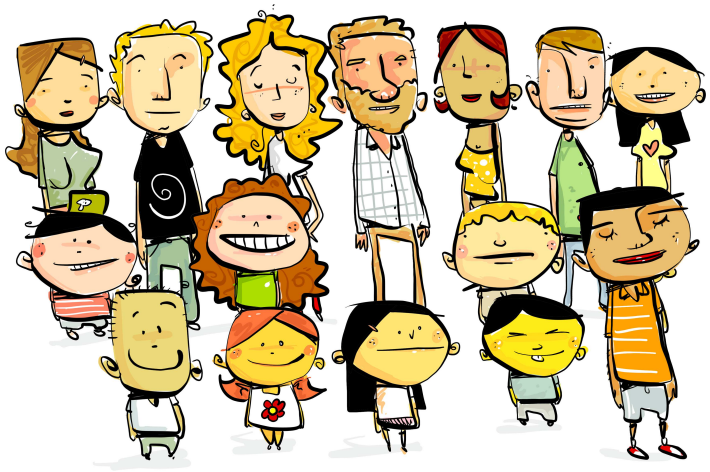

**Figura 2.7:** *Grupo de pessoas.*

alcance da sensibilização da vizinhança varia ao longo do treinamento, onde inicia afetando muitos vizinhos e o valor aproxima-se de zero ao final do treinamento.

A Figura 2.8 ilustra os neurônios da camada de entrada (sexo, idade e altura) ligados diretamente à camada de saída. A cada ciclo de treinamento, o neurônio vencedor é reajustado para responder ao estímulo recebido.

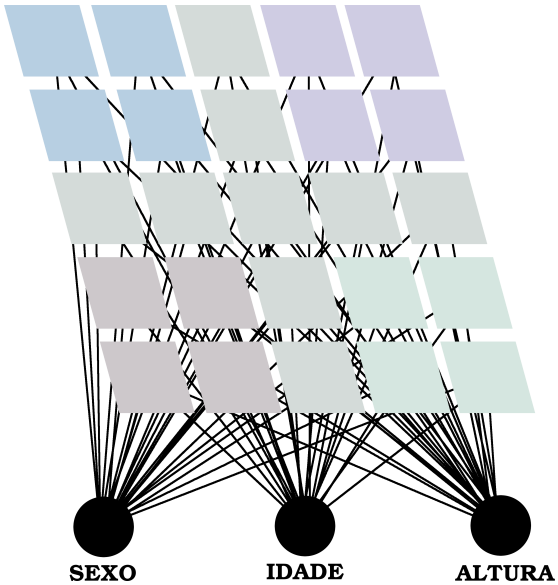


**Figura 2.8:** *Camada de entrada e camada de saída da rede SOM.*

Ao final do treinamento, teremos o conjunto de pessoas ordenadas segundo os critérios sexo, idade e altura. Para ilustrar tal situação, apresentamos a Figura 2.9.

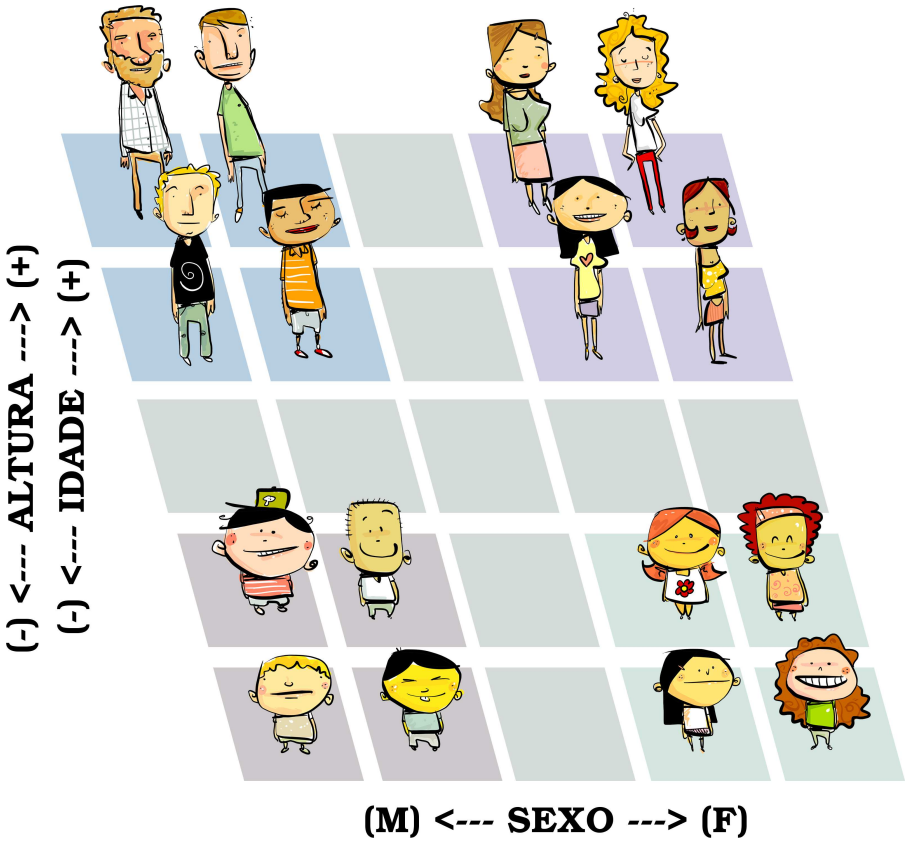


**Figura 2.9:** *Pessoas ordenadas topologicamente.*

Um exemplo clássico da aplicação dos mapas de Kohonen é a visualização de correlações complexas em dados estatísticos. Os dados, a seguir, utilizaram indicadores macro econômicos do Banco Mundial de 1992. Ao todo 39 indicadores foram empregados para descrever qualidade de vida (como saúde, nutrição e serviços educacionais). Países, cujos indicadores sócio-econômicos possuíam valores próximos, ficaram posicionados em regiões próximas no mapa. A organização topológica de todos os fatores pode ser visualizada num mapa auto-organizável de Kohonen na Figura 2.10.

Os diferentes agrupamentos do mapa foram automaticamente coloridos para descrever a situação de cada país em relação aos outros países. O mapa ficou conhecido como “mapa da pobreza” e sua estrutura pode ser visualizada no mapa geográfico da Figura 2.11.

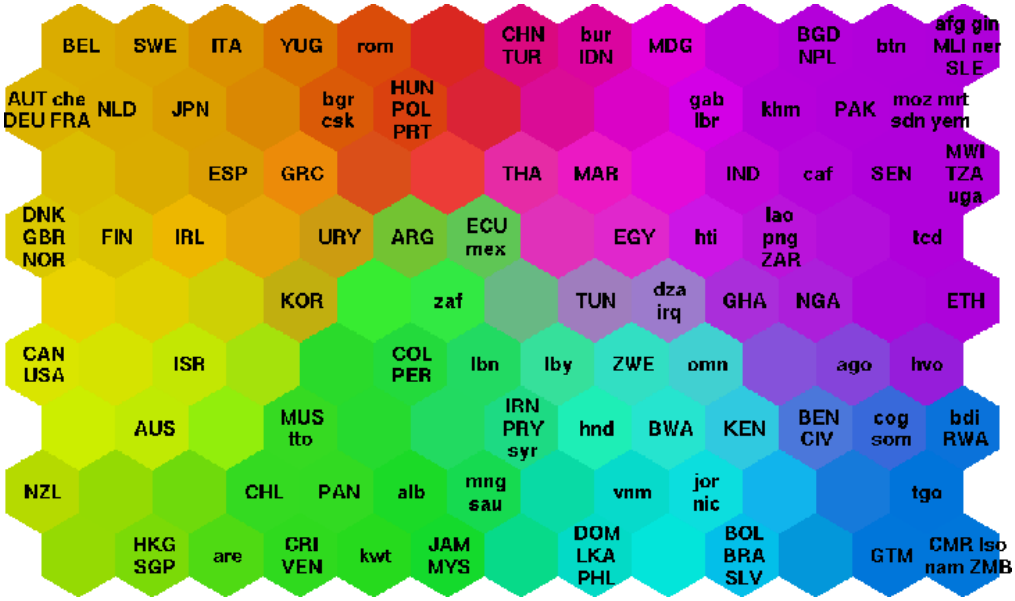


**Figura 2.10:** *Mapa de Kohonen para os indicadores macroeconômicos de 1992.*

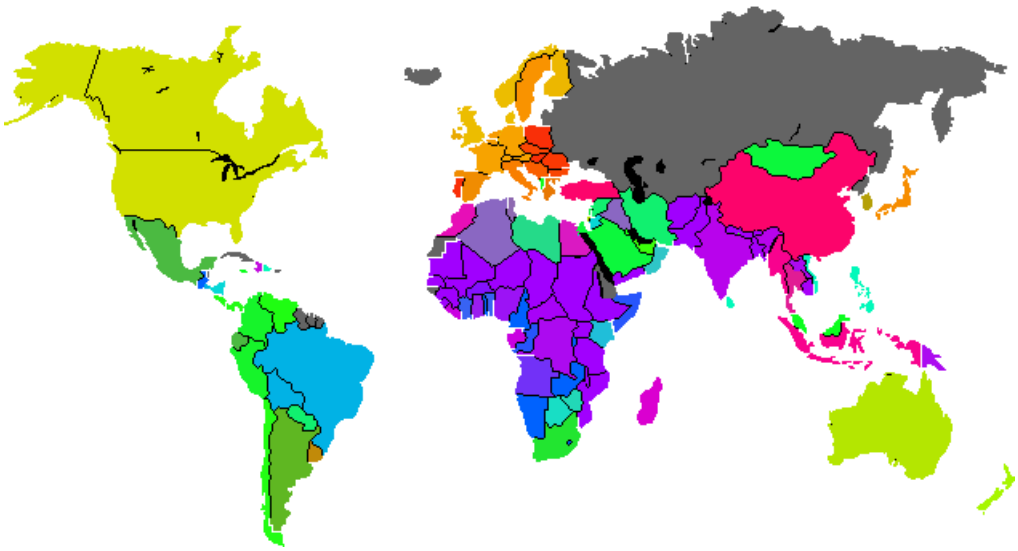

**Figura 2.11:** *Mapa geográfico para os indicadores macroeconômicos de 1992.*

**Preservação topológica em redes SOM**

A propriedade espacial ou topológica da rede é determinada pelo modo como são definidos os seus vizinhos. No início do treinamento, o conjunto de vizinhos é extenso e pouco definido, fase conhecida por ordenação global. Com o tempo o raio da vizinhança diminui e, no final do treinamento, haverá um ajuste fino (convergência) para chegar na melhor auto-organização. A preservação topológica pode ser avaliada segundo a métrica de qualidade topológica.

Uma forma simples de medir a qualidade topológica de uma rede SOM é obtendo a média da soma das distâncias entre o primeiro neurônio vencedor e o segundo neurônio vencedor. Assim quanto maior o resultado obtido, menor ordenação global terá sido alcançada. A Equação 2-2 demonstra a avaliação de qualidade para mapas bi-dimensionais.

$$Qualidade\ Topologica = \frac{1}{N}\sum_{i=1}^{N}\alpha \tag{2-2}$$

Onde:

$$\alpha = \begin{cases} max(|Lv - Lv'|, |Cv - Cv'|), & \text{se } max(|Lv - Lv'|, |Cv - Cv'|) > 1 \\ 0, & \text{caso contrário} \end{cases}$$

$N$ = Quantidade de neurônios;
$Lv$ = Linha do neurônio vencedor;
$Lv'$ = Linha do segundo vencedor;
$Cv$ = Coluna do neurônio vencedor;
$Cv'$ = Coluna do segundo vencedor;

As redes SOM são grande fonte de inspiração para nosso estudo nos quesitos organização topológica e redução de dimensionalidade, objetivos compartilhados com o algoritmo genético proposto.

### 2.2.3 Matriz de Proximidade

O problema de organização topológica de objetos multidimensionais foi originalmente definido por [Basalaj 2001] como uma **matriz de proximidade**. A matriz de proximidade é uma técnica de visualização que permite apresentar informações em alta densidade (redução de dimensionalidade). Cada elemento abstrato de uma coleção de dados é representado dentro de

uma célula na matriz, a fim de evitar a sobreposição de informações. Os relacionamentos de proximidade são preservados ao agrupar elementos similares na matriz e mantendo distantes os mais dissimilares.

O tipo de informação a ser apresentada, em alta densidade, diz respeito a qualquer tipo de dados cujas representações visuais não são diretas, como dados extraídos de tabelas num banco de dados, gráficos, imagens, ou qualquer coleção de dados abstratos, passível de ser submetida ao cálculo de similaridade entre os itens da coleção.

A apresentação refere-se à visualização das informações num mapa com preservação topológica, onde cada elemento a ser ordenado é representado dentro de uma das células da matriz, sem sobreposição e as relações de proximidade são preservadas, agrupando elementos similares e mantendo distantes os dissimilares.

### Origens do problema

Basalaj iniciou o estudo do problema ao ordenar uma coleção de imagens utilizando algoritmos MDS (Escalamento Multidimensional) para gerar matrizes de proximidade. O pesquisador obteve sucesso parcial em seu experimento mas verificou que havia perda de informação nas diversas sobreposições de imagens detectadas nos mapas gerados pelos algoritmos MDS. A análise do MDS não leva em consideração a quantidade total de elementos e o tamanho do espaço onde os elementos serão alocados, originando uma sobreposição parcial ou até mesmo completa de alguns itens da coleção.

A partir da capacidade do MDS em calcular similaridade e dissimilaridade entre pares de elementos, Basalaj criou algoritmos para forçar a alocação dos elementos a espaços não ocupados na matriz, a partir da saída de um algoritmo MDS, trabalhando com diferentes densidades de preenchimento na matriz de proximidade. O problema da sobreposição pode ser resolvido satisfatoriamente se apresentarmos o resultado na forma tridimensional, assim seria possível apresentar as n-dimensões num eixo de profundidade, gerado a partir do algoritmo MDS. No entanto, esta seria uma nova extensão do problema.

### Experimentos e resultados com a Matriz de Proximidade

Os algoritmos utilizados por [Basalaj 2001] ordenam um conjunto de $n$ objetos numa matriz, de forma que objetos similares ocupem células vizinhas

e estejam separados dos objetos dissimilares.

O autor executou testes utilizando quatro algoritmos diferentes para o agrupamento dos itens na matriz de proximidade: Greedy, Improved Greedy, Otimização "Squeaky Wheel" e Algoritmos Genéticos. Os três primeiros algoritmos utilizam técnicas baseadas no MDS (Multi Dimensional Scaling), um conjunto de técnicas estatísticas usadas na visualização de dados para explorar similaridades e dissimilaridades entre pares de dados. O algoritmo genético foi utilizado para gerar uma matriz de proximidade e permutar os genes dentro da matriz através dos operadores de variação: *crossover* cíclico [Goldberg 1989] e mutação do tipo *swap* [Reeves 1995].

A qualidade da solução é aproximada através de uma função de "perda", baseada nos coeficientes de dissimilaridade entre os dados. Os três algoritmos baseados em MDS são heurísticas para minimizar a função de perda, garantindo um mínimo de dissimilaridade entre os dados arranjados na matriz.

Na maior parte do experimento, o pesquisador utilizou um número de células na matriz maior que a quantidade de elementos a serem ordenados. No entanto, testes estatísticos mostraram que os melhores resultados eram obtidos ao agrupar dados em matrizes preenchidas totalmente (densidade de 100%).

## 2.3 Conclusão

Neste Capítulo apresentamos a fundamentação teórica do Sistema Proposto: Mapas Auto-organizáveis e as Matrizes de Proximidade. A organização topológica, essencial nos Mapas Auto-organizáveis, permanecem como a grande motivação na construção de matrizes organizadas topologicamente.

As Matrizes de Proximidade apoiaram-se principalmente nos algoritmos MDS tradicionais e suas respectivas funções de "stress" ou "loss" (perda) para indicar o quanto a matriz resultante representa o desajuste entre os objetos. A função de "stress" será referenciada no Sistema Proposto simplesmente como função de avaliação, onde, tratando de um problema de minimização, o algoritmo genético tem como objetivo minimizar o custo calculado na função de avaliação. A função de avaliação, apresentada no próximo Capítulo, trará uma nova abordagem no cálculo de distância e dissimilaridade, mas inicialmente apoiou-se na métrica de qualidade dos mapas de Kohonen. Neste intuito, o próximo Capítulo destina-se a clarificar o Sistema Proposto e sua relação com as Matrizes de Proximidade e Mapas Auto-organizáveis.

CAPÍTULO 3

# Sistema Proposto

## 3.1 Introdução

Neste capítulo, apresentamos um sistema destinado a organizar, topologicamente numa matriz, um conjunto de dados multidimensionais, usando hierarquia de algoritmos genéticos. No capítulo seguinte, esse sistema será empiricamente testado e comparado à versão genética não hierárquica, a qual estaremos nos referindo como versão genética plana.

Algoritmos Genéticos são otimizadores de uso geral e adaptam-se a problemas de maximização ou minimização, dependendo apenas do objetivo da função de avaliação. O Sistema Proposto aplica algoritmos genéticos na busca por matrizes cujas permutações entre os objetos (ou itens de dados alocados em cada posição da matriz) minimizem a distância total de dissimilaridade. Os tradicionais algoritmos de ordenação unidimensional, tais como *quicksort* [Hoare 1962], *shellsort* [Shell 1959] e *heapsort* [Williams 1964], também realizam uma minimização entre as distâncias de cada item no vetor. A Figura 3.1 demonstra a minimização das distâncias de dissimilaridade entre os objetos (aqui representados por números inteiros) de vetores com dimensão única:

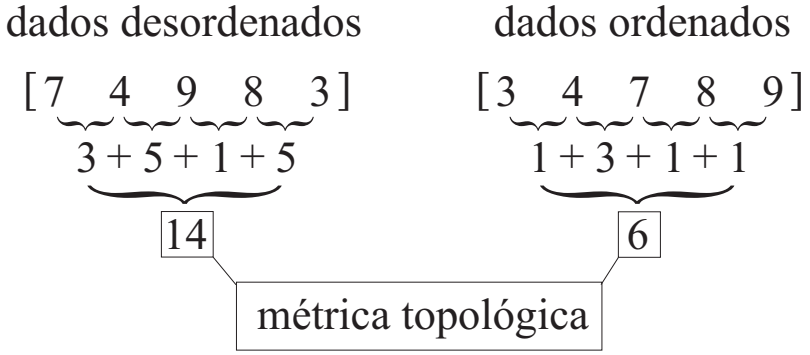


**Figura 3.1:** *Métrica topológica em vetores ordenados e desordenados.*

A idéia de minimização, obtida a partir da distância de dissimilaridade entre os itens do vetor, foi expandida ao escopo bidimensional. A distância de dissimilaridade de cada item da matriz é calculada em relação a todos os outros itens da matriz. Ao minimizar a dissimilaridade entre os objetos, resulta-se diretamente no mapa topologicamente ordenado.

O Sistema Proposto busca permutações topologicamente ordenadas, a partir da geração e avaliação de indivíduos (possíveis soluções). Assim o sistema proposto gera permutações (matrizes específicas) originadas das trocas de posições (coordenadas) entre objetos da matriz. Os objetos a serem organizados de forma topológica podem ser unidimensionais (como os números inteiros) ou multidimensionais (cores, dados estatísticos ou qualquer objeto com múltiplos atributos).

Os algoritmos genéticos agem selecionando as melhores permutações, ou seja, aquelas cujas distâncias de dissimilaridade inter-objetos estejam com o custo minimizado. O vocabulário de MDS (Escalamento Multidimensional) das matrizes de proximidade [Basalaj 2001] trata este custo (valor a ser minimizado) como função de "stress" ou desajuste, o qual referenciaremos simplesmente por função de avaliação do algoritmo genético.

Apesar das matrizes modelarem tipicamente formas retangulares em espaços bidimensionais, o sistema proposto pode lidar com qualquer tipo de forma desde que as relações de vizinhança (entre os objetos do espaço resultante) estejam claramente definidas. Em outras palavras, pode-se trabalhar na otimização, por exemplo, de um conjunto de dados multidimensionais dispostos sobre a superfície de uma esfera (3D). Aplicações em realidade virtual aumentada podem usufruir deste tipo de disposição futuramente. Alguns exemplos de formas adequadas à ação do sistema proposto estão mostrados na Figura 3.2. As formas apresentadas na Figura 3.2, exibem os genes alocados em formas retangulares, porém mantendo-se as relações de vizinhança, os genes podem estar inseridos em triângulos, pentágonos, hexágonos ou mesmo círculos.

No presente experimento, optou-se pela utilização do formato quadrado (primeiro exemplo da Figura 3.2), por sua forma geométrica simplificada. Seguindo os princípios de metodologia científica, a pesquisa é iniciada a partir do que acredita-se ser sua forma mais simples, para mais tarde, partir em direção aos aspectos mais complexos do problema [Lakatos e Marconi 1991].

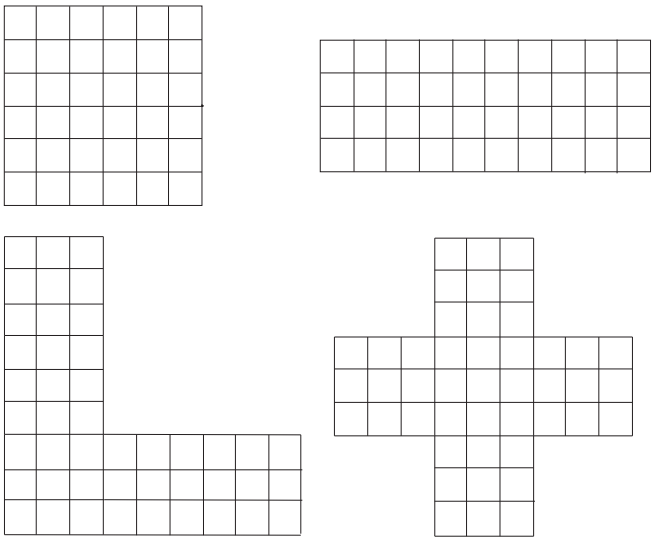

**Figura 3.2:** *Exemplos de formas tratáveis pelo sistema proposto.*

## 3.2 Hierarquia de Algoritmos Genéticos

O Sistema Proposto aplica ao problema uma Hierarquia de Algoritmos Genéticos. Dois algoritmos genéticos distintos são empregados no problema: o primeiro deles faz a busca por soluções, enquanto o segundo, altera parâmetros no algoritmo que está fazendo a busca de soluções. A hipótese principal do Capítulo introdutório supõe que a Hierarquia de Algoritmo Genéticos acelera tal busca. Os indivíduos do algoritmo genético posicionado no topo da hierarquia constituem a população de controle, e referem-se aos parâmetros do problema, aplicados ao algoritmo inferior da hierarquia. A população localizada no domínio de soluções (parte inferior da hierarquia) possui indivíduos representando as soluções candidatas para o problema. A Figura 3.3 exibe as populações e seus respectivos domínios: parâmetros e soluções.

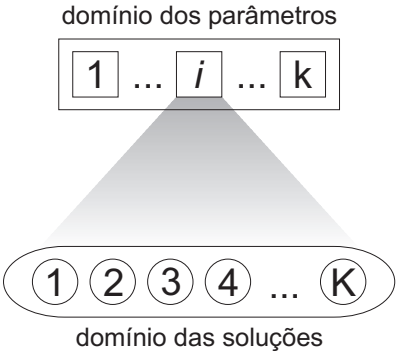


**Figura 3.3:** *Hierarquia de algoritmos genéticos no sistema proposto.*

A solução genética plana (tradicional) mostrou-se lenta na busca

por ótimos globais. A motivação de acelerar a solução plana levou-nos a experimentar novos operadores de seleção de indivíduos, seleção de genes (objetos) e operadores de variação.

Tratando-se de um problema de permutação (explosão combinatória), o cruzamento convencional (através da seleção de pontos de corte) resultava em descendentes inválidos no domínio do problema, ou seja, após o cruzamento, o indivíduo poderia ter objetos repetidos na matriz de resultados. O custo computacional para repará-lo era superior ao de gerar um novo indivíduo. A matriz de proximidade [Basalaj 2001] utilizou o cruzamento cíclico para evitar tal inconveniente. No entanto, optamos por desenvolver o cruzamento inter-objetos, um tipo de cruzamento pouco ortodoxo onde apenas um único pai gera um único filho alterando as coordenadas do conjunto de objetos. A vantagem deste tipo de cruzamento, e justificativa da escolha, é evitar situações de estagnação em mínimos ou máximos locais [Everett 2001].

Inicialmente três novos operadores de variação foram criados para o problema de permutação e evitavam a geração indivíduos inválidos. Cada operador do cruzamento inter-objetos efetuava de forma particular a troca de coordenadas entre os objetos, sugerindo então, que fossem aplicados em diferentes momentos.

Durante o cálculo da aptidão do indivíduo, era possível avaliar qual dos genes estava contribuindo de forma mais negativa na avaliação. Aproveitando esta característica, 3 novos operadores foram criados para serem utilizados na recombinação gênica: randômico, determinístico e probabilístico. Durante a recombinação gênica, é necessário escolher qual gene será recombinado e o gene passou a ser escolhido através de um dos 3 novos operadores. O operador randômico sorteia aleatoriamente o gene a ser realocado. O operador determinístico, a partir dos cálculos pré-efetuados na função de avaliação, escolhe o gene que mais contribui negativamente na nota de aptidão, enquanto o operador probabilístico atribui probabilidades a cada gene, onde aos genes mais adaptados ao problema, atribuem-se maiores probabilidades para serem escolhidos na recombinação.

Os operadores foram criados sem o conhecimento sobre qual renderia melhores resultados. O algoritmo genético posicionado na parte superior controla exatamente os operadores de variação, de seleção gênica e de seleção de indivíduos, apostando nos operadores obtendo melhores resultados durante a evolução. A meta-adaptação acerca do problema, garante um auto-ajuste dos parâmetros aplicados à população inferior (domínio das soluções).

A seção seguinte detalha o Sistema Proposto nos domínios de solução

e parâmetros.

## 3.3 Detalhamento do Sistema Proposto

A presente seção descreverá a representação, funções de aptidão, variação, seleção de indivíduos, seleção de genes e elitismo do algoritmo genético posicionado na parte inferior da hierarquia (domínio das soluções).

### 3.3.1 Representação

Os indivíduos são representados por arranjos de elementos dispostos numa matriz totalmente preenchida (densidade de 100% de uso do espaço útil da matriz), pois trata-se do caso em que os algoritmos genéticos obtém os melhores resultados [Basalaj 2001]. Cada indivíduo representa uma possível permutação topológica, onde a alteração de posições (coordenadas) entre elementos da matriz, geram novas permutações, ou seja, novos indivíduos. Posições na matriz representam genes carregando informações sobre as características do objeto, elemento ou dado. Ao tomarmos como exemplo a representação de cores no formato RGB (dados tridimensionais de pigmentos vermelho, verde e azul) numa matriz de tamanho 5x5, teremos a representação das características do indivíduo dispostos em uma matriz multidimensional 5x5x3 conforme exemplo da Figura 3.4.

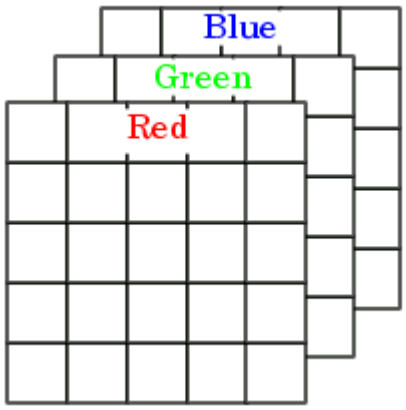


**Figura 3.4:** *Representação matricial no formato RGB*

Se os objetos a serem permutados tivessem 15 características e desejássemos uma permutação de tamanho 7 x 7 , cada indivíduo seria representado por uma matriz multidimensional de tamanho 7 x 7 x 15, ou seja, não existem limites para a quantidade de características ou atributos definidos

numa coleção de objetos. As múltiplas dimensões dos objetos afetam diretamente o desempenho do algoritmo genético, pois uma quantidade maior de atributos, implica em complexidade superior no momento de avaliar a adaptação de cada indivíduo. A multidimensionalidade dos objetos a serem representados não deve ser confundida com a multidimensionalidade da matriz, neste caso, bidimensional.

### 3.3.2 Avaliação

Parte significativa dos esforços da presente pesquisa, concentraram-se na função de avaliação dos indivíduos. A função de avaliação calcula a aptidão de um determinado indivíduo da população frente ao problema, ou seja, avalia-se a qualidade da solução representada pelo indivíduo. A função de custo (a ser minimizada pelo processo de otimização), na linguagem dos Algoritmos Genéticos, é o complemento da função de avaliação. Para cada objeto $a_{ij}$ existe um custo correspondente, onde a distância euclidiana entre os objetos $|a_{ij} - a_{lc}|$ é a medida de dissimilaridade entre os objetos do mapa. A medida de avaliação de qualidade dos mapas de kohonen, foi o ponto de partida para a função de avaliação inicial e mostrou-se ineficiente, pois apesar de avaliar com eficiência a dissimilaridade entre vizinhos, deixava a desejar no qualidade global, ou seja, o resultado final era um mapa composto de pequenas áreas bem ordenadas que não se harmonizavam com o restante do mapa globalmente. A Equação 3-1 define a função de avaliação inicial incluindo o respectivo custo de dissimilaridade entre os objetos. A função avalia apenas a dissimilaridade entre os vizinhos ao redor do objeto.

$$f(x) = \sum_{c=i-1}^{i+1} \sum_{l=j-1}^{j+1} .|a_{ij} - a_{lc}| \tag{3-1}$$

A segunda função de avaliação baseia-se na métrica de custo apresentada em [Basalaj 2001] e apresentou bons resultados na ordenação global, mas deixava deficiente as relações entre objetos vizinhos. Ressaltamos que a pesquisa de [Basalaj 2001] era aplicada a coleções de imagens, obtendo níveis satisfatórios de ordenação, no entanto para ordenação de objetos n-dimensionais, o mapa final produzido pelo algoritmo genético alcançava uma qualidade global aceitável, mas as relações entre vizinhos diretos apresentavam qualidade inferior pois estas relações não eram preservadas. A aptidão do indivíduo é obtida calculando a dissimilaridade de cada indivíduo em relação a todos os outros elementos da matriz, diferente da função de avaliação

anterior, cuja avaliação era baseada numa vizinhança de raio um (1) e todos os elementos são ponderados pela distância Manhattan ($max(|l-i|+|c-j|)$).

$$f(x) = \sum_{\substack{i=1 \\ j=1}}^{n} \sum_{\substack{l=1 \\ c=1}}^{n} max(|l-i|+|c-j|).|a_{ij}-a_{lc}| \tag{3-2}$$

Uma terceira função de avaliação tentou unir as duas métricas de avaliação, ou seja, uma ordenação global de qualidade, mantendo também uma boa qualidade local entre vizinhos diretos. Os resultados da união das métricas demonstraram ineficiência, pois para cada tamanho de matriz, era necessário balancear de forma precisa a forma de ponderação de cada função ao uni-las numa função mono objetivo. Uma possível solução para este problema é utilizar algoritmos genéticos de múltiplos objetivos, ficando como sugestão para um trabalho futuro.

A quarta e última função de avaliação obteve resultados satisfatórios e significativos. Analisando as funções de avaliação anteriores, ressaltam-se os atributos dissimilaridade e a distância afim de alcançar uma organização topológica entre os objetos no mapa. Objetos similares que estão próximos, devem permanecer próximos e objetos dissimilares que estão distantes, devem permanecer distantes. A função de aptidão calcula o custo e penaliza objetos dissimilares que estão próximos e objetos similares que estão distantes. A Figura 3.5 descreve o problema da função de avaliação:

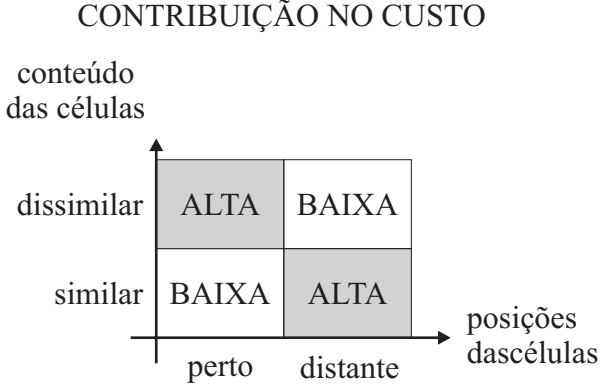


**Figura 3.5:** *Dissimilaridade x Distância entre objetos.*

Ao observar a Figura 3.5,é possível associar o comportamento desejado a função “sela”, apresentada na Figura 3.6 por um parabolóide hiperbólico.

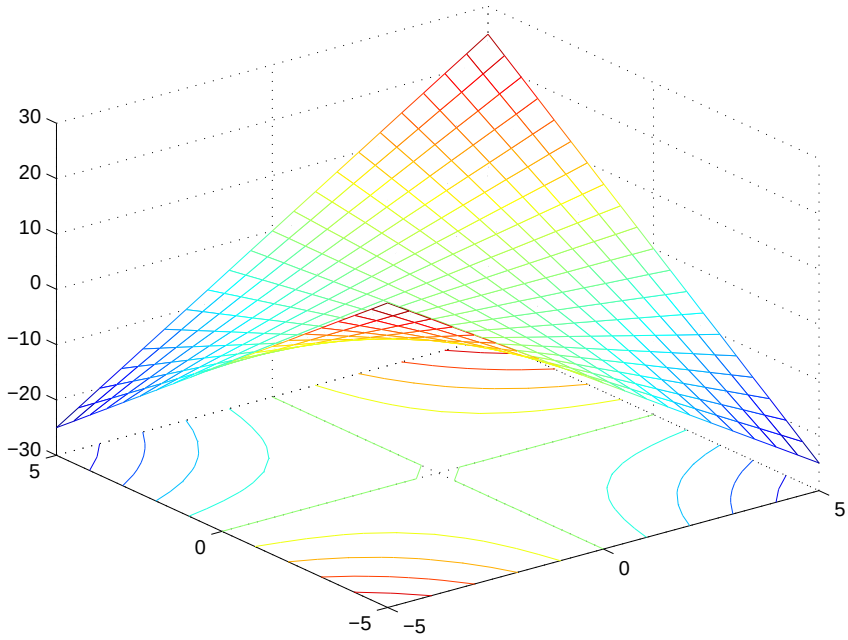


**Figura 3.6:** *Parabolóide Hiperbólico.*

A sela é um parabolóide hiperbólico que modela exatamente o comportamento desejado para a função de avaliação, onde tanto objetos dissimilares e próximos quanto objetos similares e distantes são penalizados através da função de custo no momento de calcular a aptidão do indivíduo em questão. A Equação 3-3 define a função de avaliação do Sistema Proposto:

$$f(P) = \sum_{i=1,j=1}^{n} custo_{ij} \tag{3-3}$$

onde o custo é representado pela Equação 3-4:

$$custo_{ij} = -\sum_{l=1,c=1}^{n} \left\{ \left( X_{ij,lc} - \frac{X_{max}}{2} \right) \left( Y_{ij,lc} - \frac{Y_{max}}{2} \right) + \frac{X_{max} Y_{max}}{4} \right\} \tag{3-4}$$

A sela foi deslocada para o quadrante de valores positivos, evitando a inclusão de valores negativos nas avaliações. A Figura 3.7 exibe o ajuste da sela segundo descrita na Equação 3-4.

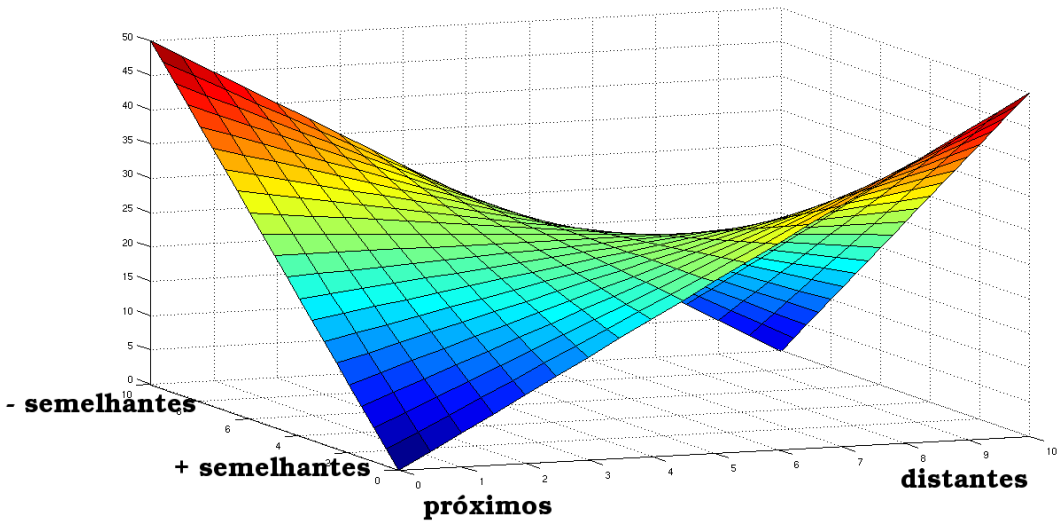


**Figura 3.7:** *Parabolóide Hiperbólico ajustado ao problema.*

Apresentamos a seguir um exemplo completo do cálculo de uma avaliação para o indivíduo representado pela matriz $ind$ [1]. Neste cenário utilizaremos objetos do tipo números inteiros (cujos valores estão entre 1 e 25) a fim de facilitar a demonstração (o Capítulo 4 apresenta os experimentos empiricamente testados nos cenários de números inteiros e cores).

```
ind =
    17    24     1     8    15
    23     5     7    14    16
     4     6    13    20    22
    10    12    19    21     3
    11    18    25     2     9
```

A avaliação pode ser descrita em 3 fases distintas:

1. Cálculo das distâncias euclidianas máximas de cada posição até o canto mais distante;

2. Cálculo das distâncias euclidianas máximas dos valores associados a cada posição até o objeto mais dissimilar presente no mapa;

3. Cálculo do custo com a função sela (aplicando os valores obtidos nos itens 1 e 2, penalizando objetos similares distantes e objetos próximos dissimilares).

[1] A matriz de exemplo $ind$ é um quadrado mágico: o somatório de qualquer linha, coluna ou diagonais principais sempre resultam no valor 65.

O cálculo da matriz com as distâncias euclidianas máximas de cada posição do mapa até o canto mais distante é exemplificado no trecho de código 3.1. O resultado é acumulado na matriz $mdmax$.

**Código 3.1** Cálculo das distâncias euclidianas máximas até a extremidade mais distante.

```
msize = 5;
for ia = 1:msize
    for ib = 1:msize
        distcanto1 = sqrt( abs((ia-1)*(ia-1))
                     + abs((ib-1)*(ib-1)) );
        distcanto2 = sqrt( abs((ia-1)*(ia-1))
                     + abs((ib-msize)*(ib-msize)));
        distcanto3 = sqrt( abs((ia-msize)*(ia-msize))
                     + abs((ib-1)*(ib-1)));
        distcanto4 = sqrt( abs((ia-msize)*(ia-msize))
                     + abs((ib-msize)*(ib-msize)));
        winner1 = max(distcanto1,distcanto2);
        winner2 = max(distcanto3,distcanto4);
        mdmax(ia,ib) = max(winner1,winner2);
    end
end
```

O exemplo a seguir exibe o cálculo detalhado das iterações apresentadas no trecho de código 3.1.

```
Para ia=1 e ib=1 temos:
distcanto1 = 0;
distcanto2 = 4;
distcanto3 = 4;
distcanto4 = 5.6569;
winner1 = 4;
winner2 = 5.6569;
mdmax(1,1) = 5.6569;

Para ia=1 e ib=2 temos:
distcanto1 = 1;
distcanto2 = 3;
distcanto3 = 4.1231;
distcanto4 = 5;
winner1 = 3;
winner2 = 5;
```

```
mdmax(1,2) = 5;

...

Ao final das iterações teremos ia=5 e ib=5, obtendo:
distcanto1 = 5.6569
distcanto2 = 4
distcanto3 = 4
distcanto4 = 0
winner1 = 5.6569
winner2 = 4
mdmax(5,5) = 5.6569;
```

A matriz $mdmax$ é utilizada no cálculo da função de avaliação, especialmente no custo associado à sela. Ao final das iterações para uma matriz de dimensão 5x5, $mdmax$ apresenta os seguintes valores:

```
mdmax =
    5.6569    5.0000    4.4721    5.0000    5.6569
    5.0000    4.2426    3.6056    4.2426    5.0000
    4.4721    3.6056    2.8284    3.6056    4.4721
    5.0000    4.2426    3.6056    4.2426    5.0000
    5.6569    5.0000    4.4721    5.0000    5.6569
```

Na segunda fase são calculadas as distâncias máximas de dissimilaridade (através de distâncias euclidianas) entre os valores presentes em cada coordenada (x,y) da matriz. O cálculo é apresentado no trecho de código 3.2, cuja saída é a matriz $ddmax$.

**Código 3.2** Cálculo das distâncias euclidianas máximas entre os valores de cada gene.

```
1 msize = 5;
2 menorValor = 1;
3 maiorValor = 25;
4 for ia = 1:msize
5     for ib = 1:msize
6         ddmax(ia,ib) = max(abs(ind(ia,ib) - menorValor),
7                            abs(ind(ia,ib) - maiorValor));
8     end
9 end
```

O exemplo a seguir apresenta uma demonstração das iterações apresentadas no trecho de código 3.2.

```
Variando ia de 1 a 5 e ib de 1 a 5 teremos:

ddmax(1,1) = max(abs(17 - 1), abs(17 - 25)) = 16
ddmax(1,2) = max(abs(24 - 1), abs(24 - 25)) = 23
ddmax(1,3) = max(abs( 1 - 1), abs( 1 - 25)) = 24
ddmax(1,4) = max(abs( 8 - 1), abs( 8 - 25)) = 17
ddmax(1,5) = max(abs(15 - 1), abs(15 - 25)) = 14

ddmax(2,1) = max(abs(23 - 1), abs(23 - 25)) = 22
ddmax(2,2) = max(abs( 5 - 1), abs( 5 - 25)) = 20
ddmax(2,3) = max(abs( 7 - 1), abs( 7 - 25)) = 18
ddmax(2,4) = max(abs(14 - 1), abs(14 - 25)) = 13
ddmax(2,5) = max(abs(16 - 1), abs(16 - 25)) = 15

ddmax(3,1) = max(abs( 4 - 1), abs( 4 - 25)) = 21
ddmax(3,2) = max(abs( 6 - 1), abs( 6 - 25)) = 19
ddmax(3,3) = max(abs(13 - 1), abs(13 - 25)) = 12
ddmax(3,4) = max(abs(20 - 1), abs(20 - 25)) = 19
ddmax(3,5) = max(abs(22 - 1), abs(22 - 25)) = 21

ddmax(4,1) = max(abs(10 - 1), abs(10 - 25)) = 15
ddmax(4,2) = max(abs(12 - 1), abs(12 - 25)) = 13
ddmax(4,3) = max(abs(19 - 1), abs(19 - 25)) = 18
ddmax(4,4) = max(abs(21 - 1), abs(21 - 25)) = 20
ddmax(4,5) = max(abs( 3 - 1), abs( 3 - 25)) = 22

ddmax(5,1) = max(abs(11 - 1), abs(11 - 25)) = 14
ddmax(5,2) = max(abs(18 - 1), abs(18 - 25)) = 17
ddmax(5,3) = max(abs(25 - 1), abs(25 - 25)) = 24
ddmax(5,4) = max(abs( 2 - 1), abs( 2 - 25)) = 23
ddmax(5,5) = max(abs( 9 - 1), abs( 9 - 25)) = 16
```

A matriz $ddmax$ contém as distâncias máximas de dissimilaridade de cada posição da matriz $ind$ até o gene mais dissimilar.

```
ddmax =
    16    23    24    17    14
```

```
22    20    18    13    15
21    19    12    19    21
15    13    18    20    22
14    17    24    23    16
```

A última fase da avaliação do indivíduo efetivamente calcula o custo "distância X dissimilaridade", penalizando os objetos desajustados, obedecendo a função da sela. O trecho de código 3.3 exemplifica o cálculo para matrizes de tamanho 5x5 com permutações topológicas no cenário de números inteiros.

**Código 3.3** Cálculo do custo "distância X dissimilaridade".

```
adaptacao = 0;
msize = 5;
for ia = 1:msize
    for ib = 1:msize
        total = 0;
        for ic = 1:msize
            for id = 1:msize
                dist = sqrt(ind(ia,ib)^2 + ind(ic,id)^2);
                md = sqrt((ia-ic)^2 + (ib-id)^2);
                valorMDmax = mdmax(ia,ib);
                valorDDmax = ddmax(ia,ib);
                custo =   -1 * (( md - (valorMDmax/2) )
                              * (dist - (valorDDmax/2) ))
                              + (valorMDmax*valorDDmax)/4
                total = total + custo;
            end
        end
        adaptacao = adaptacao + total;
    end
end
```

O Apêndice III apresenta a evolução das iterações (conforme apresentadas no código 3.3) para o primeiro item da matriz $ind(1,1)$.

```
Variando ia de 1 a 5 e ib de 1 a 5 teremos:
adaptacao = 0
ia = 1
ib = 1
ind(1,1) = 17
```

```
Variando ic de 1 a 5 e id de 1 a 5 teremos:
Para ic = 1 e id = 1:
ind(1,1) = 17
dist = 0
md = 0
valorMDmax = 5.6569
valorDDmax = 16
custo = 0
totalGene = 0

...

Para ic = 5 e id = 5:
ind(5,5) = 9
dist = 8
md = 5.6569
valorMDmax = 5.6569
valorDDmax = 16
custo = 22.6274
total = 600.6631

adaptacao = adaptacao + 600.6631
```

Até este ponto, foram calculados o custo das distâncias e dissimilaridades do item $ind(1,1)$ em relação a todos os outros itens da matriz, onde a variável cumulativa $adaptacao$ alcançou o valor $600,6631$. As iterações demonstradas para o item $ind(1,1)$ repetem-se para todos os demais itens da matriz $ind$. As variáveis $ia$, $ib$, $ic$ e $id$ continuam a variar de 1 a 5. Apresentamos a seguir, o cálculo para o último gene na última iteração, onde ic e id assumem o valor 5:

```
Para ia = 5 e ib = 5:
adaptacao = 1.2246e+04

Para ic = 5 e id = 5:
ind(5,5) = 9
dist = 0
md = 0
valorMDmax = 5.6569
```

```
valorDDmax = 16
custo = 0
total = 578.0357

adaptacao = 12823.9001
```

Ao comparar a adaptação da matriz exemplo $ind$ (vide Tabela 3.1) a uma matriz ordenada topologicamente (conforme apresentado na Tabela 3.2), verifica-se visualmente a organização topológica otimizada entre os objetos da segunda matriz. A aptidão calculada para a matriz exemplo alcançou o valor $12823,9001$, enquanto a matriz organizada topologicamente obteve o valor $10588,4532$.

**Tabela 3.1:** *Matriz $ind$ sem organização topológica.*

| | | | | |
|---|---|---|---|---|
| 17 | 24 | 1 | 8 | 15 |
| 23 | 5 | 7 | 14 | 16 |
| 4 | 6 | 13 | 20 | 22 |
| 10 | 12 | 19 | 21 | 3 |
| 11 | 18 | 25 | 2 | 9 |

**Tabela 3.2:** *Matriz organizada topologicamente.*

| | | | | |
|---|---|---|---|---|
| 8 | 6 | 4 | 3 | 1 |
| 10 | 9 | 7 | 5 | 2 |
| 13 | 12 | 11 | 15 | 14 |
| 23 | 21 | 19 | 17 | 16 |
| 25 | 24 | 22 | 20 | 18 |

### 3.3.3 Seleção de Indivíduos

A seleção escolhe os indivíduos mais adaptados ao ambiente para participarem da variação (reprodução). Duas estratégias de seleção de indivíduos foram aplicadas na seleção dos progenitores:

- Probabilística - cada indivíduo recebe uma probabilidade proporcional a sua posição no *ranking* de avaliação. Ao somar as probabilidades atribuídas a cada indivíduo o valor total é um (1).
- Contraste - o melhor indivíduo recebe uma probabilidade de 0,9 enquanto o pior indivíduo recebe a probabilidade de 0,1. Os indivíduos intermediários recebem as probabilidades por interpolação aritmética simples.

Em ambas as estratégias de seleção, é aplicada uma roleta para sortear os indivíduos participantes da operação de variação. O tipo de seleção de indivíduos a ser aplicado é um dos parâmetros a ser controlado através do algoritmo superior da hierarquia, assumindo o valor um (1) para a seleção "probabilística" e valor dois (2) para a seleção de "contraste".

### 3.3.4 Seleção de Genes

A natureza do problema, sugestionou melhorias na maneira de selecionar os genes participantes da variação, com o objetivo de otimizar a performance do algoritmo, ao invés de somente sortear os genes aleatoriamente.

Os genes de cada indivíduo são exatamente os objetos a serem permutados, no cenário ao qual estiverem participando. Os três (3) operadores de seleção de genes são:

- Randômica - a primeira estratégia de seleção de genes foi a "Randômica", uma escolha natural para decidir onde iniciar a variação.
- Determinística - o segundo tipo de seleção de genes chamada "Determinística", derivou do processo de avaliação dos indivíduos. Durante a avaliação, obtém-se o somatório das distâncias de um objeto para todos os seus vizinhos (fase 3 do processo de avaliação). Portanto, é possível determinar qual gene mais contribui de forma incisiva para piorar a nota de avaliação. O gene (objeto) com o valor mais desajustado é exatamente o objeto selecionado para participar da operação de variação.

- Probabilística - a terceira estratégia de seleção de genes derivou da estratégia determinística. Sendo possível determinar durante a avaliação, o quanto cada gene contribui positivamente ou negativamente para sua nota de adaptação, é possível estabelecer um *ranking* qualitativo dos genes e atribuir probabilidades de serem escolhidos na variação. Os genes mais desajustados topologicamente (contribuindo negativamente para a aptidão) terão maiores probabilidades de serem escolhidos.

Os operadores de seleção de genes fazem parte dos parâmetros a serem controlados pelo algoritmo genético hierárquico, e assumem respectivamente os valores 1 (randômica), 2 (determinística) ou 3 (probabilística).

### 3.3.5 Variação

O operador inicial de cruzamento (crossover) foi implementado (num experimento inicial) da maneira tradicional: os casais eram selecionados para o cruzamento de acordo com suas respectivas probabilidades e após o sorteio do ponto de corte, os indivíduo eram recombinados para compor a nova geração. Nesta abordagem tradicional, a maioria dos indivíduos nasciam defeituosos, ou seja, eram soluções inválidas por se tratar de um problema de permutação onde os objetos não poderiam repetir-se dentro da matriz de resultados. Os indivíduos foram submetidos a correções, encontrando os genes repetidos e substituindo-os pelos genes omissos. A operação de "correção" dos indivíduos inválidos consumia mais tempo que gerar um novo indivíduo. Segundo [Eiben e Smith 2003], é possível obter um algoritmo genético funcional somente com mutação, e sem cruzamento, mas o contrário seria impossível. Testes iniciais foram realizados aplicando somente a mutação durante a variação. O operador de mutação utilizado foi a "Permutação de Troca" (Swap Permutation) onde dois genes eram selecionados randomicamente e trocavam de posição. Segundo [Everett 2001], este tipo de variação é denominado "cruzamento inter-objetos", onde um único pai gera um único filho e apesar de pouco usual, alterar as coordenadas dos objetos é uma forma de prevenção para evitar mínimos locais. Apoiado neste relevante argumento, outros 2 novos operadores foram aplicados ao problemas, indo de encontro aos operadores tradicionais de permutação como "Crossover Cíclico" (Cyclic Crossover), "Crossover Parcialmente Mapeado" (Partial Mapped Crossover) e "Recombinação de Extremos" (Edge Recombination) [Eiben e Smith 2003]. O crossover cíclico foi o tipo de cruzamento utilizado por [Basalaj 2001], e num trabalho

futuro sugere-se a comparação da eficácia entre a variação inter-objetos e os demais operadores tradicionais de permutação.

Três operadores de variação foram desenvolvidos e cada pai gera um único filho, evitando que o indivíduo proveniente do cruzamento possua uma permutação inválida, contemplando assim, a inexistência de objetos repetidos na matriz.

Os três operadores de variação a serem descritos a seguir são:

- Simples;
- Simples com 3 genes;
- Variação na Sub-área.

O primeiro operador nomeado “Simples”, altera as coordenadas de dois objetos (obedecendo um dos operadores da seleção de genes). A Figura 3.8 exibe um exemplo da operação.

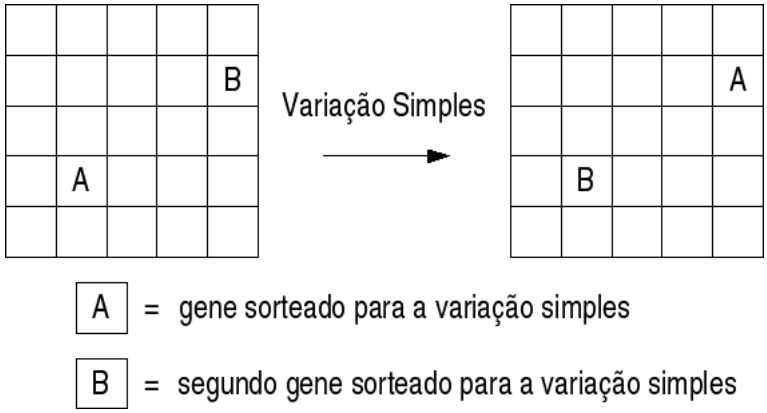


**Figura 3.8:** *Variação Simples*

O segundo operador de variação, “Simples com 3 Genes”, seleciona 3 genes e faz um rodízio das posições: o primeiro gene substitui a posição do segundo gene, o segundo gene substitui o terceiro e o terceiro assume as coordenadas do primeiro gene. A Figura 3.9 ilustra a variação.

O terceiro operador nomeado “Variação na área sorteada”, sorteia uma sub-área da matriz e os objetos na sub-área sorteada trocam de posição aleatoriamente de 1 a $n$ vezes onde $n$ é número de genes dentro da área sorteada. A Figura 3.10 ilustra o funcionamento do operador.

As três estratégias estão inseridas no contexto de cruzamento inter-objetos [Everett 2001], com um único pai gerando um único filho. Os três (3) tipos de variação fazem parte dos parâmetros a serem controlados pelo algoritmo genético hierárquico, assumindo respectivamente os valores 1 (Simples), 2 (Simples com 3 genes) e 3 (Variação na sub-área).

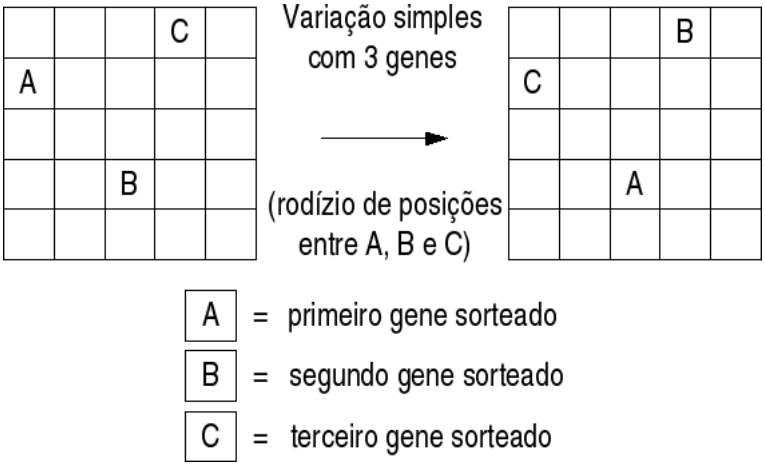


**Figura 3.9:** *Variação Simples com 3 genes*

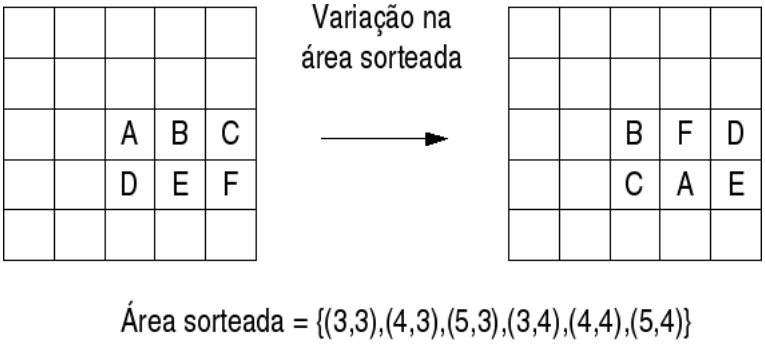


**Figura 3.10:** *Variação na sub-área do mapa*

### 3.3.6 Elitismo

O elitismo na solução genética plana preserva os 50% melhores indivíduos da população antiga e une aos outros 50% melhores indivíduos da nova geração nascida da operação de variação.

O elitismo na solução genética hierárquica salva os indivíduos de acordo com o número de indivíduos na população de parâmetros. Uma população de parâmetros com 8 indivíduos e uma população de soluções com 64 indivíduos, resulta numa população final de 512 indivíduos pertencentes a população de soluções, pois cada conjunto de parâmetros (indivíduo da população de parâmetros) será aplicado aos 64 indivíduos da população de soluções. O elitismo acontece reunindo os 8 melhores indivíduos gerados por cada conjunto de parâmetros resultando numa nova geração com 64 indivíduos na população de soluções compondo a novas base para futuras explorações.

## 3.4 Detalhamento do Algoritmo superior na hierarquia

A solução genética plana expandiu a quantidade de operadores de variação, seleção de indivíduos e seleção de genes na tentativa de incrementar a performance do algoritmo. A complexidade de ajustar todos estes parâmetros de forma a dar performance máxima ao algoritmo, mostrou-se um problema digno de ser avaliado por um segundo algoritmo genético dadas as possíveis combinações. A solução genética Hierárquica posiciona um algoritmo mais alto na hierarquia para controlar os parâmetros do algoritmo de soluções. Os indivíduos da população alta carregam em seus genes as configurações de seleção de indivíduos, seleção de genes e operador de variação. A presente seção descreve os operadores do algoritmo mais alto da hierarquia e suas respectivas especificidades.

### 3.4.1 Interação entre os algoritmos

O algoritmo superior na hierarquia interage diretamente com o algoritmo inferior aplicando o conjunto de operadores e recebendo um retorno qualitativo sobre a configuração aplicada. A situação pode ser clarificada através do seguinte exemplo: se tivermos uma população de controle com 4 indivíduos e uma população de soluções com 32 indivíduos, cada conjunto de parâmetros dos indivíduos da população alta é aplicada durante uma geração na população baixa. A média das notas de aptidão obtidas a partir do conjunto de parâmetros aplicados, será a aptidão do indivíduo da população de controle com os respectivos parâmetros. Ao fim das quatro (4) gerações (1 geração para cada indivíduo da população de controle), o algoritmo seleciona os oito (8) melhores indivíduos de cada geração para compor a nova população de soluções. O número de oito (8) indivíduos é obtido através de (Número Indivíduos População Soluções / Número Indivíduos População Controle), ou seja, 32/4. Ressalta-se então uma restrição com relação ao número de indivíduos em cada população, ou seja, o número de indivíduos da população de controle deve ser um múltiplo do número de indivíduos escolhidos para a população de soluções.

A população de controle aplica os parâmetros na população de soluções, obtendo 4 novas gerações totalizando 128 indivíduos. Destes apenas 32 comporão a nova população de soluções, que receberá os parâmetros da população de parâmetros. O elitismo na solução hierárquica, seleciona os 8 melhores indivíduos gerados por cada conjunto de parâmetros.

Os parâmetros de controle aplicados pela população alta, são alterados a cada nova geração da população alta, pois os mesmos também passar por operações de seleção, variação e mutação.

### 3.4.2 Representação

Os indivíduos do algoritmo do topo da hierarquia possuem 3 genes codificados como valores inteiros:

- O primeiro gene determina a estratégia de seleção de indivíduos que pode ser do tipo Probabilística (1) ou Contraste (2).
- O segundo gene representa as estratégias de variação: Simples (1), Simples com 3 genes (2), Variação na área sorteada (3).
- O terceiro gene representa o esquema de seleção de genes, assumindo os valores Randômico (1), Determinístico (2) ou Probabilístico (3).

### 3.4.3 Avaliação

A avaliação dos indivíduos no algoritmo superior, é a média das notas da função de avaliação recebidas quando seus parâmetros são aplicados à população de soluções.

### 3.4.4 Seleção

Os indivíduos da população de controle são selecionados para a operação de variação seguindo uma distribuição probabilística, de acordo com a nota de aptidão recebida após aplicar seus respectivos parâmetros. Uma roleta trata de selecionar os indivíduos para o cruzamento.

### 3.4.5 Variação

Os genes da população de controle são representados por números inteiros. O cruzamento na população de controle efetua um sorteio aleatório entre os valores dos genes de cada pai resultando em apenas um (1) novo indivíduo. A exemplo a seguir ilustra a operação de cruzamento.

```
Pai 1 = [ 2   1   3 ]
          |   |   |
         ou  ou  ou
          |   |   |
Pai 2 = [ 1   1   2 ]
-------------
Filho = [ 1   1   3 ]
```

### 3.4.6 Mutação

A taxa de mutação é fixa em 5% porque os cromossomos não são longos (apenas 3 genes) e pela necessidade de manter diversidade suficiente no domínio de parâmetros para evitar convergência prematura. Quando um gene é sorteado para mutar, o valor da operação pode assumir qualquer valor na faixa de valores disponíveis. Por exemplo, se o gene de variação é escolhido para a mutação, ele receberá aleatoriamente um valor entre 1 e 3 (possíveis valores para o gene).

### 3.4.7 Elitismo

Na população de controle não foi empregado nenhum tipo de elitismo, mas a nova população do domínio de soluções é formada por contribuições iguais a partir da aplicação do conjunto de parâmetros de cada indivíduo do domínio superior. As contribuições são compostas pelos melhores indivíduos de cada geração.

### 3.4.8 Conclusão

O Capítulo atual apresentou as especificidades do Sistema Proposto contemplando detalhes da implementação e exemplos para ilustrar o funcionamento dos mecanismos. O próximo Capítulo descreve os experimentos realizados com a solução plana e com a solução hierárquica apontando uma comparação entre ambos através de estatística descritiva e inferencial para cenários de objetos com números inteiros e cores.

CAPÍTULO 4

# Experimentos e Resultados

## 4.1 Introdução

A principal característica, comum aos algoritmos de ordenação, é o escopo unidimensional, ou seja, o produto final é um *ranking* (seqüência ordenada), contendo os dados dispostos de forma crescente ou decrescente baseado em algum campo predefinido (chave de ordenação). O desejo de expandir a noção de ordenação ao escopo multidimensional (chaves múltiplas), tomando como base todo o registro (e não apenas o campo-chave), foi a principal motivação no desenvolvimento de algoritmos capazes de superar a limitação de unidimensionalidade.

Neste presente estudo, deseja-se ordenar objetos ou dados multidimensionais e apresentar o resultado final em menores dimensões, particularmente, em mapas bidimensionais. Para concretizar a ordenação em duas dimensões, a primeira tentativa buscou avaliar, empiricamente, a complexidade do problema, escrevendo números aleatórios no papel e tentando ajustá-los numa matriz $3$ x $3$. Com base nesta tentativa, percebeu-se a necessidade de pesquisa computacional específica.

A Solução Genética Hierárquica (Sistema Proposto) foi comparada a Solução Genética Plana nos cenários de números inteiros e cores. No cenário de números inteiros, os objetos são unidimensionais. Enquanto no cenário de cores, os objetos são multidimensionais. Nos dois cenários, utilizou-se matrizes de ordem 5, 7 e 10. As Seções seguintes apresentam maiores informações sobre a realização dos experimentos.

## 4.2 Origem das amostras

As amostras originam-se de simulações computacionais, executando 30 casos por experimento. Um caso refere-se a uma simulação completa

(até alcançar o critério de parada), utilizando o sistema sem hierarquia (Solução Plana) ou o sistema com hierarquia de Algoritmos Genéticos (Solução Hierárquica). Cada caso é composto de um cenário (números inteiros ou cores), um tamanho de matriz (para o mapa resultante) e a presença ou ausência de hierarquia no algoritmo genético. Os casos são submetidos aos experimentos com matrizes de dimensões 5x5, 7x7 e 10x10, nos sistemas plano e hierárquico, contabilizando um total de 360 casos. A Tabela 4.1 exibe o resumo de casos para o experimento.

**Tabela 4.1:** *Casos por experimento.*

| Cenário | Tamanho da matriz | AG Plano | AG Hierárquico |
|---|---|---|---|
| Números Inteiros | 5 x 5 | 30 casos | 30 casos |
| Números Inteiros | 7 x 7 | 30 casos | 30 casos |
| Números Inteiros | 10 x 10 | 30 casos | 30 casos |
| Cores | 5 x 5 | 30 casos | 30 casos |
| Cores | 7 x 7 | 30 casos | 30 casos |
| Cores | 10 x 10 | 30 casos | 30 casos |

## 4.3 Material e Instrumentos

A seção a seguir descreve as variáveis e ferramentas (recursos computacionais) dos experimentos.

### 4.3.1 Variáveis

O experimento efetua uma comparação entre os sistemas plano e hierárquico, através de manipulação das variáveis dependentes Custo ($VDC$) e Adaptação ($VDA$) a partir da variável independente ($VI$). Detalhamos a seguir as principais variáveis avaliadas nesta pesquisa.

- Variável Independente ($VI$) - presença de Hierarquia de Algoritmos Genéticos.

As Variáveis Dependentes são detalhadas de acordo com o custo e adaptação alcançados pelo Algoritmo Genético:

- Custo ($VDC$) - indica o número de avaliações efetuadas até alcançar o valor final obtido na variável Adaptação, ou seja, quanto tempo o algoritmo levou para encontrar a solução em número de avaliações.

- Adaptação ($VDA$) - indica a melhor aptidão obtida ao final de 1 caso completo de simulação. Tratando-se de um problema de minimização, quanto menor o valor da variável, melhor a qualidade, pois indica que a distância de dissimilaridade entre os objetos internos da matriz resultado foram minimizados.

Este trabalho apresenta a seguinte hipótese básica: a presença de hierarquia em Algoritmos Genéticos acelera a busca de mapas topológicos de objetos multidimensionais.

- Em termos de variáveis temos: $VI \Longrightarrow VDC$.

A hipótese principal refere-se a aceleração da busca empregando Algoritmos Genéticos Hierárquicos ($VI$), ou seja, deseja-se um número menor de avaliações ($VDC$) até encontrar uma solução global.

Como hipótese secundária, temos: a presença de hierarquia em Algoritmos Genéticos melhora qualitativamente os mapas topológicos de objetos com múltiplas dimensões.

- Expressando tal hipótese em variáveis, temos: $VI \Longrightarrow VDA$

A hipótese secundária refere-se à melhoria na qualidade das soluções apresentadas, ou seja, a presença de Algoritmos Genéticos Hierárquicos ($VI$) resulta em menores valores de adaptação ($VDA$), comparado ao Algoritmo Genético Plano.

A presença de Hierarquia de Algoritmos Genéticos no Sistema Proposto é referenciado como Solução Hierárquica, enquanto o sistema caracterizado pela ausência de hierarquia é identificado como Solução Plana.

### 4.3.2 Ferramentas de Desenvolvimento

A ferramenta MATLAB©[1] (versão 7.0.4.352 (R14) Service Pack 2) foi utilizada na construção dos algoritmos genéticos hierárquico e plano. As simulações foram realizadas no próprio software, usando individualmente seis computadores pessoais da plataforma x86. Não foi utilizado nenhum tipo de "Toolbox"[2] para algoritmos genéticos, além dos recursos de programação do

[1] Matlab é uma marca registrada da MathWorks.

[2] "Caixa de ferramentas", conjunto de software computacional aplicável ao MATLAB, codificado por especialistas em cada área de relevência. Ex.: Toolbox de Redes Neurais, Toolbox de Lógica Nebulosa, Toolbox de Bioinformática, Toolbox Estatística, Toolbox de Realidade Virtual, Toolbox de Algoritmo Genético e Busca Direta.

software. A Tabela 4.2 apresenta as configurações de hardware e os sistemas operacionais dos computadores usados no experimento.

**Tabela 4.2:** *Configurações de hardware e sistemas operacionais dos computadores usados nos experimentos.*

| Processador | Clock | Memória | S. Operacional |
|---|---|---|---|
| AMD Atlhon 64 | 2800 Mhz | 512 MB | Linux Ubuntu 6.06 (32 bits) |
| AMD Atlhon 64 | 2800 Mhz | 512 MB | Linux Ubuntu 6.06 (32 bits) |
| AMD Atlhon 64 | 2800 Mhz | 512 MB | Debian (64 bits) |
| AMD Atlhon 64 | 2800 Mhz | 512 MB | Windows XP (32 bits) |
| Pentium 4 | 2400 Mhz | 256 MB | Windows XP (32 bits) |
| Pentium 4 | 3000 Mhz | 1024 MB | Windows XP (32 bits) |

A pesquisa não tem como objetivo comparar ou avaliar o desempenho entre diferentes tipos de hardware ou sistemas operacionais.

### 4.3.3 Cenário de números inteiros

A idéia de ordenar números em duas dimensões foi utilizada para explorar as fronteiras do problema de interesse, auxiliando a compreender melhor a natureza do problema e definir rumos da pesquisa.

O problema da ordenação multidimensional apresenta a dificuldade conhecida como explosão combinatorial (presente em problemas como o "Caixeiro Viajante"[3]). Em outras palavras, a complexidade aumenta exponencialmente com o aumento das dimensões da matriz. O experimento foi repetido com matrizes de diferentes tamanhos: $5$ x $5$, $7$ x $7$ e $10$ x $10$, cujas complexidades em relação ao espaço de busca são respectivamente $25!$, $49!$ e $100!$, conforme Tabela 4.3.

O cálculo do espaço de busca desconsidera a possível existência de elementos repetidos no conjunto a ser ordenado, pois diminuiriam os valores, anteriormente calculados, para o tamanho espaço de busca. As matrizes foram "inicializadas" com números inteiros de $1$ a $n^2$, onde $n$ corresponde ao tamanho da matriz (uma matriz de ordem 7, por exemplo, é preenchida de 1 a 49), desta forma garante-se que não há elementos iguais no conjunto.

[3]Um caixeiro viajante deseja visitar N cidades uma única vez, com o objetivo de minimizar a distância total percorrida para visitar as N cidades.

**Tabela 4.3:** *Tamanho da matriz e Tamanho do espaço de busca*

| Tamanho da matriz | Tamanho do espaço de busca |
|---|---|
| 3 x 3 | $9! = 362880$ |
| 4 x 4 | $16! = 2,09 \text{ x } 10^{13}$ |
| 5 x 5 | $25! = 1,55 \text{ x } 10^{25}$ |
| 6 x 6 | $36! = 3,71 \text{ x } 10^{41}$ |
| 7 x 7 | $49! = 6,08 \text{ x } 10^{62}$ |
| 8 x 8 | $64! = 1,26 \text{ x } 10^{89}$ |
| 9 x 9 | $81! = 5,79 \text{ x } 10^{120}$ |
| 10 x 10 | $100! = 9,33 \text{ x } 10^{157}$ |

### 4.3.4 Cenário de cores

O experimento de ordenação topológica em matrizes com números inteiros, possibilitou o ajuste e corroboração da função de avaliação dos indivíduos e demais operadores genéticos. O presente experimento avalia o sistema proposto com dados multidimensionais, e para tal, a motivação remete aos mapas de cores de Kohonen [Kohonen 1989] (rede neural SOM), com cores representadas numericamente em três dimensões: vermelho (red), verde (green) e azul (blue). Diferente dos mapas de Kohonen, o sistema proposto executa uma busca por permutações topológicas de qualidade, enquanto os mapas de kohonen "criam" os melhores protótipos, baseado em entradas do conjunto inicial de treinamento. O presente problema aloca cada cor em uma posição na matriz, de forma a preenchê-la totalmente. No SOM tradicional, após o treinamento da rede, um único neurônio pode responder por várias entradas ou nenhuma.

O experimento de ordenação de cores possibilita avaliar o resultado visual, permitindo julgamentos imediatos a respeito da qualidade das soluções. O experimento concretiza-se com matrizes de $5^a$, $7^a$ e $10^a$ ordem, permanecendo os mesmos tamanhos do espaço de busca calculados no cenário de números inteiros. A diferença entre os dois cenários (números inteiros e cores) é a complexidade dos dados a serem ordenados, elevada a três dimensões no contexto de cores. O aumento da dimensionalidade causou considerável impacto na velocidade de avaliações de indivíduos por segundo.

## 4.4 Delineamento Experimental

O experimento de ordenação com os objetos cores e números inteiros, foi submetido às soluções plana e hierárquica com diferentes parâmetros para os diferentes tamanhos de matriz. Nas matrizes de tamanho 5x5 e 7x7, a população de soluções contou com 64 indivíduos e oito na população de controle, com um critério de parada de 300000 indivíduos avaliados. A Tabela 4.4 exibe a configuração dos parâmetros critério de parada e quantidade de indivíduos na população de soluções (na solução hierárquica existe ainda o parâmetro de indivíduos na população de controle).

**Tabela 4.4:** *Delineamento experimental para matrizes de ordem 5 e 7.*

| | Solução Plana | Solução Hierárquica |
|---|---|---|
| População de soluções | 64 indivíduos | 64 indivíduos |
| População de controle | não aplicável | 8 indivíduos |
| Critério de parada | 300000 avaliações | 300000 avaliações |

Os experimentos com a matriz de tamanho 10x10 foram efetuados com 32 indivíduos na população de soluções e 4 indivíduos na população de controle e utilizou como critério de parada o número total de indivíduos avaliados, neste caso 700000. A Tabela 4.4 exibe o delineamento para matrizes de tamanho 10x10. A complexidade da matriz de ordem 10 exigiu incremento do numero de avaliações no critério de parada, devido ao aumento exponencial no tamanho do espaço de busca (Tabela 4.3) quando comparado às matrizes de ordem 5 e 7.

**Tabela 4.5:** *Delineamento experimental para matrizes de ordem 10.*

| | Solução Plana | Solução Hierárquica |
|---|---|---|
| População de soluções | 32 indivíduos | 32 indivíduos |
| População de controle | não aplicável | 4 indivíduos |
| Critério de parada | 700000 avaliações | 700000 avaliações |

## 4.5 Resultados - cenário de números inteiros

A presente Seção descreve os resultados obtidos e compara a solução tradicional, utilizando algoritmos genéticos (matriz de proximidade, com

preenchimento de 100% [Basalaj 2001]), ao sistema proposto, aplicando hierarquia de algoritmos genéticos com meta-aprendizagem acerca do problema. Os resultados são analisados através de estatística descritiva e inferencial (comparação utilizando teste t).

### 4.5.1 Análise Descritiva

A análise descritiva evidencia características da amostra, através de medidas de tendência central (moda, média e mediana), bem como medidas de dispersão (variância, desvio padrão e amplitude total).

#### Matriz de ordem 5

A variável custo apresentou uma média ligeiramente menor para a solução hierárquica, mas apresentou maior variabilidade (49391,31), com máximo de 252481 e mínimo de 15425. A Tabela 4.6 exibe o resumo da estatística descritiva aplicada ao cenário.

**Tabela 4.6:** *Custo em matrizes 5x5 ordenando números inteiros.*

| | Solução Plana | Solução Hierárquica |
|---|---|---|
| Mínimo | 20417 | 15425 |
| Máximo | 115969 | 252481 |
| Média | 41974,33 | 40103,40 |
| Desvio Padrão | 20935,49 | 49391,31 |

A Tabela 4.7 exibe o resumo comparativo dos dados amostrais obtidos nas soluções plana e hierárquica para a variável adaptação. Ambas as soluções alcançaram soluções equivalentes apontadas pelo mínimo 264711,33 e pelo máximo 265134,50. A média das soluções também apresentou diferença pouco significativa, com menor variabilidade na solução hierárquica (110,28).

**Tabela 4.7:** *Adaptação em matrizes 5x5 ordenando números inteiros.*

| | Solução Plana | Solução Hierárquica |
|---|---|---|
| Mínimo | 264711,33 | 264711,33 |
| Máximo | 265134,50 | 265134,50 |
| Média | 264810,35 | 264820,63 |
| Desvio Padrão | 111,64 | 110,28 |

### Matriz de ordem 7

A variável custo apresentou melhores resultados em termos de média, sinalizando um custo equivalente à metade do número de avaliações da solução plana. A solução hierárquica também apresentou menor variabilidade com desvio padrão de 35854,36. A Tabela 4.8 exibe o resumo do cenário.

**Tabela 4.8:** *Custo em matrizes 7x7 ordenando números inteiros.*

| | Solução Plana | Solução Hierárquica |
|---|---|---|
| Mínimo | 137857 | 63553 |
| Máximo | 297921 | 238657 |
| Média | 201884,73 | 88180,20 |
| Desvio Padrão | 40262,72 | 35854,36 |

As soluções plana e hierárquica obtiveram o mesmo mínimo (aproximadamente 5903808,54) de adaptação, logo ambas alcançaram nível de qualidade satisfatório. A solução plana apresentou maior variabilidade com um desvio padrão de 3926,44, apesar do máximo (pior solução) ter sido encontrada pela solução hierárquica. Na média a solução plana foi um pouco melhor em termos qualitativos, conforma a análise descritiva na Tabela 4.9.

**Tabela 4.9:** *Adaptação em matrizes 7x7 ordenando números inteiros.*

| | Solução Plana | Solução Hierárquica |
|---|---|---|
| Mínimo | 5903808,53 | 5903808,53 |
| Máximo | 5917085,88 | 5920371,39 |
| Média | 5908383,32 | 5908515,01 |
| Desvio Padrão | 3926,44 | 3667,90 |

**Matriz de ordem 10**

A variável custo apresentou uma grande diferença entre médias: 593682,07 na solução plana e 156281,28 na solução hierárquica, diferença de 3,79 vezes. A Tabela 4.10 apresenta os resultados da variável custo.

**Tabela 4.10:** *Custo em matrizes 10x10 ordenando números inteiros.*

| | Solução Plana | Solução Hierárquica |
|---|---|---|
| Mínimo | 426785 | 68513 |
| Máximo | 698497 | 517025 |
| Média | 593682,07 | 154883,13 |
| Desvio Padrão | 68380,23 | 108880,71 |

A solução hierárquica, em matrizes de tamanho 10x10, não apresentou grandes diferenças na qualidade dos resultados, comparada à solução plana. O mínimo (153770729,38) foi alcançado pela solução hierárquica apresentando também pouca melhoria em termos de média. A Tabela 4.11 apresenta a estatística descritiva comparando ambas soluções. A Seção seguinte discute a variabilidade das médias através da estatística inferencial.

**Tabela 4.11:** *Adaptação em matrizes 10x10 - Números inteiros.*

| | Solução Plana | Solução Hierárquica |
|---|---|---|
| Mínimo | 153844858,12 | 153770729,38 |
| Máximo | 154171527,03 | 154152940,65 |
| Média | 153959905,32 | 153920178,10 |
| Desvio Padrão | 92656,61 | 103614,47 |

### 4.5.2 Análise Inferencial

A análise utilizando estatística inferencial, verifica diferenças significativas entre as médias e evidencia os ganhos em cada tipo de solução. O teste t-Student apresenta uma comparação entre as médias das amostras, apresentadas pelas soluções plana e hierárquica.

### Matriz de ordem 5

A análise para o custo, em matrizes de tamanho 5x5, não apresenta diferenças significativas entre os resultados. O nível de significância de 0,42 não descarta a hipótese nula, sinalizando resultados equivalentes para ambos os tipos de algoritmo (plano e hierárquico).

**Tabela 4.12:** *Custo em matrizes 5x5 ordenando números inteiros - Teste t.*

| | Solução Plana | Solução Hierárquica |
|---|---|---|
| Média | 41974,33 | 40103,40 |
| Variância | 438294904,60 | 2439501217,00 |
| Observações | 30 | 30 |
| gl | 39 | |
| Stat t | 0,19 | |
| P(T<=t) uni-caudal | 0,42 | |
| t crítico uni-caudal | 1,68 | |

O nível de significância (0,36) superior ao nível de significância pré-estabelecido (0,05) indica que a hipótese nula não pode ser rejeitada, ou seja, as soluções plana e hierárquica apresentam soluções com qualidade equivalente.

**Tabela 4.13:** *Adaptação em matrizes 5x5 ordenando números inteiros - Teste t.*

| | Solução Plana | Solução Hierárquica |
|---|---|---|
| Média | 264810,35 | 264820,63 |
| Variância | 12462,60 | 12160,81 |
| Observações | 30 | 30 |
| gl | 58 | |
| Stat t | -0,35 | |
| P(T<=t) uni-caudal | 0,36 | |
| t crítico uni-caudal | 1,67 | |

### Matriz de ordem 7

A variável custo apresenta diferença significativa entre as médias. O nível de significância observado foi de $5,61631E-17$, logo a hipótese nula deve ser rejeitada, indicando que a solução hierárquica apresenta um custo menor para alcançar soluções de qualidade equivalente a solução plana. A Tabela

4.14 apresenta o resultado do teste t-Student para matrizes de tamanho 7x7 na ordenação de números inteiros.

**Tabela 4.14:** *Custo em matrizes 7x7 ordenando números inteiros - Teste t.*

| | Solução Plana | Solução Hierárquica |
|---|---|---|
| Média | 201884,73 | 88180,20 |
| Variância | 1621086260,00 | 1247576742,00 |
| Observações | 30 | 30 |
| gl | 57 | |
| Stat t | 11,62 | |
| P(T<=t) uni-caudal | 0,00 | |
| t crítico uni-caudal | 1,67 | |

O nível de significância observado (0,44) está acima do nível de significância aceito (0,05) para a variável adaptação. A hipótese nula não pode ser descartada, indicando possível equivalência das médias.

**Tabela 4.15:** *Adaptação em matrizes 7x7 ordenando números inteiros - Teste t.*

| | Solução Plana | Solução Hierárquica |
|---|---|---|
| Média | 5908383,32 | 5908515,01 |
| Variância | 15416896,40 | 13060960,61 |
| Observações | 30 | 30 |
| gl | 58 | |
| Stat t | -0,13 | |
| P(T<=t) uni-caudal | 0,44 | |
| t crítico uni-caudal | 1,67 | |

### Matriz de ordem 10

A variável custo repete o resultado alcançado nas matrizes de tamanho 7x7. O nível de significância ($1,91237E-24$) menor que o nível de significância do teste t indica que a hipótese nula deve ser rejeitada, ou seja, a solução hierárquica possui um custo menor para produzir soluções de qualidade equivalente às soluções apresentadas pelo algoritmo genético plano. A Tabela 4.16 apresenta os resultados comparativos do teste t.

Para o quesito adaptação, o nível de significância alcançado (0,06) é superior ao nível de significância esperado (0,05), logo a hipótese nula não pode ser rejeitada, indicando que os algoritmos produziram soluções

**Tabela 4.16:** *Custo em matrizes 10x10 ordenando números inteiros - Teste t.*

| | Solução Plana | Solução Hierárquica |
|---|---|---|
| Média | 593682,06 | 154883,13 |
| Variância | 4675855800,00 | 11504859131,00 |
| Observações | 30 | 30 |
| gl | 49 | |
| Stat t | 18,89 | |
| P(T<=t) uni-caudal | 0,00 | |
| t crítico uni-caudal | 1,67 | |

equivalentes. A Tabela 4.17 apresenta os resultados do teste t-Student para matrizes de tamanho 10x10.

**Tabela 4.17:** *Matrizes 10x10 ordenando números inteiros - Teste t adaptação*

| | Solução Plana | Solução Hierárquica |
|---|---|---|
| Média | 153959905,30 | 153920178,10 |
| Variância | 8585247170,00 | 10544530083,00 |
| Observações | 30 | 30 |
| gl | 57 | |
| Stat t | 1,57 | |
| P(T<=t) uni-caudal | 0,06 | |
| t crítico uni-caudal | 1,67 | |

## 4.6 Resultados - cenário de cores

Para o cenário de cores, aplicamos a Estatística Descritiva na demonstração das características amostrais dos dados, coletados através das simulações computacionais. A Estatística Inferencial é aplicada na análise de possíveis generalizações para a população.

### 4.6.1 Análise Descritiva

A Estatística Descritiva utiliza técnicas para resumir e descrever um conjunto de dados, através de medidas de tendência central como média, moda e mediana, aliadas a medidas de dispersão como a variância, desvio padrão e amplitude total.

### Matriz de ordem 5

O custo da solução hierárquica foi em média (40615,40) bastante próximo do custo apresentado pela solução plana (49460,20), uma diferença de apenas 10%. A solução hierárquica apresentou menor variabilidade representada pelo desvio padrão de 44831,22.

**Tabela 4.18:** *Custo em matrizes 5x5 ordenando cores.*

| | Solução Plana | Solução Hierárquica |
|---|---|---|
| Mínimo | 17025 | 16449 |
| Máximo | 211521 | 261697 |
| Média | 49460,20 | 40615,40 |
| Desvio Padrão | 50614,43 | 44831,22 |

As matrizes de ordem 5 apresentaram os resultados menos significativos, onde observam-se poucas variações entre as soluções plana e hierárquica. A solução hierárquica apresentou a melhor (5293,42) e a pior (5438,72) adaptação, sendo em média (5343,06) ligeiramente melhor que a solução plana.

**Tabela 4.19:** *Adaptação em matrizes 5x5 ordenando cores.*

| | Solução Plana | Solução Hierárquica |
|---|---|---|
| Mínimo | 5297,50 | 5293,42 |
| Máximo | 5422,56 | 5438,72 |
| Média | 5347,16 | 5343,06 |
| Desvio Padrão | 33,19 | 47,51 |

### Matriz de ordem 7

A solução hierárquica apresenta uma média bem menor (57%) com relação ao custo para alcançar a adaptação final. A solução plana apresenta menor variabilidade (57455,89), visto que manteve valores altos (mínimo: 114113) para o custo, diferentemente da solução hierárquica (mínimo: 54337).

Nas matrizes de tamanho 7x7, a variável adaptação indica qualidades de soluções bastante próximas. A média obtida na solução plana (107745,39) foi ligeiramente menor que a média da solução hierárquica (107822,34). Novamente a solução plana obteve a melhor (106623,01) e a pior (109747,41)

**Tabela 4.20:** *Custo em matrizes 7x7 ordenando cores.*

| | Solução Plana | Solução Hierárquica |
|---|---|---|
| Mínimo | 114113 | 54337 |
| Máximo | 293505 | 292929 |
| Média | 191026,07 | 121153,00 |
| Desvio Padrão | 57455,89 | 66446,20 |

adaptação. A menor variabilidade foi apresentada pela solução plana com um desvio padrão de 760,45.

**Tabela 4.21:** *Adaptação em matrizes 7x7 ordenando cores.*

| | Solução Plana | Solução Hierárquica |
|---|---|---|
| Mínimo | 106623,01 | 106665,00 |
| Máximo | 109747,41 | 109728,94 |
| Média | 107745,39 | 107822,34 |
| Desvio Padrão | 760,45 | 856,77 |

### Matriz de ordem 10

A variável custo indica o número de indivíduos avaliados até alcançar o valor obtido na adaptação final. A solução hierárquica teve uma média muito inferior comparada à solução plana, praticamente a metade do custo apresentado na solução plana. A solução plana apresentou o pior indivíduo dentre os 60 casos com um máximo de 691512 avaliações. O menor custo foi apresentado pela solução hierárquica, mas não representa necessariamente uma solução ótima, pois o algoritmo pode estagnar num mínimo local. Desta vez a solução plana apresentou menor variabilidade com o desvio padrão de 101629,24.

**Tabela 4.22:** *Custo em matrizes 10x10 ordenando cores.*

| | Solução Plana | Solução Hierárquica |
|---|---|---|
| Mínimo | 335969 | 139553 |
| Máximo | 691521 | 669729 |
| Média | 547796,20 | 339096,47 |
| Desvio Padrão | 101629,24 | 147334,40 |

No presente problema, tratando-se de uma minimização, a variável adaptação terá maior qualidade quando apresentar um valor menor ao com-

parar ambas as soluções. Dentro deste contexto, a solução plana apresenta o melhor (1963917,98) e pior (1972076,11) indivíduo, respectivamente o mínimo e o máximo, dentre os 60 casos da amostra. Na média, a solução hierárquica apresenta uma média melhor e o desvio padrão demonstra menor variabilidade.

**Tabela 4.23:** *Adaptação em matrizes 10x10 ordenando cores.*

| | Solução Plana | Solução Hierárquica |
|---|---|---|
| Mínimo | 1963917,98 | 1964325,56 |
| Máximo | 1988766,81 | 1973189,69 |
| Média | 1972076,11 | 1969332,60 |
| Desvio Padrão | 7082,56 | 2228,79 |

### 4.6.2 Análise Inferencial

Após verificar diferenças importantes através da análise descritiva, apresentamos um estudo da significância das diferenças evidenciadas anteriormente. A Estatística Inferencial apresenta técnicas que permitem verificar generalizações a partir da amostra, então torna-se possível afirmar se existe ganho no uso da Solução Hierárquica (Sistema Proposto) em relação à Solução Plana.

Através do teste t-Student avaliaram-se os experimentos nos diferentes tamanhos de matriz, com nível de significância de 5%, valor tipicamente aceito na comunidade científica. A análise verifica as variáveis custo e adaptação na ordenação de cores com matrizes de ordem 5, 7 e 10, presumindo variâncias diferentes.

#### Matriz de ordem 5

A variável custo não apresentou resultados significativos para matrizes de tamanho 5x5. O nível de significância efetiva alcançada foi de 0,23, indicando que a hipótese nula não pode ser rejeitada. A Tabela 4.24 apresenta um resumo do teste t.

A adaptação em matrizes 5x5 alcançou o nível de significância efetivo de 0,35, ou seja, a hipótese nula não foi rejeitada pois não existe diferença significativa entre a diferença de médias. O resumo do teste t pode ser observado na Tabela 4.25.

**Tabela 4.24:** *Custo em matrizes 5x5 - Teste t.*

| | Solução Plana | Solução Hierárquica |
|---|---|---|
| Média | 49460,20 | 40615,40 |
| Variância | 2561820149,00 | 2009838161,00 |
| Observações | 30 | 30 |
| gl | 57 | |
| Stat t | 0,71 | |
| P(T<=t) uni-caudal | 0,23 | |
| t crítico uni-caudal | 1,67 | |

**Tabela 4.25:** *Adaptação em matrizes 5x5 - Teste t.*

| | Solução Plana | Solução Hierárquica |
|---|---|---|
| Média | 5347,16 | 5343,06 |
| Variância | 1101,75 | 2257,16 |
| Observações | 30 | 30 |
| gl | 52 | |
| Stat t | 0,38 | |
| P(T<=t) uni-caudal | 0,35 | |
| t crítico uni-caudal | 1,67 | |

**Matriz de ordem 7**

A Tabela 4.26 apresenta o teste comparativo para a variável custo. O nível de significância efetivo (0,00002) é menor que nível de significância limite (0,05), logo a hipótese nula foi rejeitada, ou seja, as soluções não são equivalentes em termos de custo, pois o sistema proposto possui um custo menor para ordenar topologicamente objetos do tipo cor, em matrizes de tamanho 7x7.

O teste comparativo para a adaptação em matrizes de tamanho 7x7 apresentou um valor de nível de significância efetivo unicaudal (0,35) maior que 0,05, logo a hipótese nula não foi rejeitada, ou seja, em matrizes de tamanho 7x7 não há diferenças significativas de médias. O resumo do teste t pode ser observado na Tabela 4.27.

**Matriz de ordem 10**

O teste t utilizado na análise da variável custo, contemplando as soluções plana e hierárquica, é apresentado na Tabela 4.28. O nível de significância efetivo unicaudal (0,000000002) é menor que alfa (0,05), logo a hipótese

**Tabela 4.26:** *Custo em matrizes 7x7 - Teste t.*

| | Solução Plana | Solução Hierárquica |
|---|---|---|
| Média | 191026,06 | 121153,00 |
| Variância | 3301179868,00 | 4415097044,00 |
| Casos | 30 | 30 |
| gl | 57 | |
| Stat t | 4,35 | |
| P(T<=t) uni-caudal | 0,00 | |
| t crítico uni-caudal | 1,67 | |

**Tabela 4.27:** *Adaptação em matrizes 7x7 - Teste t.*

| | Solução Plana | Solução Hierárquica |
|---|---|---|
| Média | 107745,39 | 107822,34 |
| Variância | 578282,30 | 734054,35 |
| Observações | 30 | 30 |
| gl | 57 | |
| Stat t | -0,36 | |
| P(T<=t) uni-caudal | 0,35 | |
| t crítico uni-caudal | 1,67 | |

nula de que as soluções são equivalentes no custo foi rejeitada, ou seja, a solução hierárquica possui um custo menor para ordenar objetos topologicamente em mapas de tamanho 10x10, com diferença significativa entre as médias. O valor t observado, igual a 6,38, está na região crítica (t crítico = 1,67).

O teste comparativo dos dados das amostras das soluções plana e hierárquica para a variável adaptação apresentaram um valor efetivo unicaudal (0,03) menor que o nível de signifcância (0,05), logo a hipótese nula (solução são equivalentes) foi rejeitada, ou seja, em matrizes de tamanho 10x10, a solução hierárquica produz valores com melhor adaptação em relação à solução plana, pois a diferença de médias é significativa (veja a Tabela 4.29).

## 4.7 Discussão

O Sistema Proposto (solução hierárquica) apresentou resultados satisfatórios, corroborando parcialmente a hipótese principal, pois o ganho da solução hierárquica não é observado para matrizes pequenas (tamanho 5x5). No entanto apresenta ganhos superiores a 100% em matrizes maiores (tamanhos 7x7 e 10x10). O espaço de busca das matrizes de tamanho 7x7 é apro-

**Tabela 4.28:** *Custo em matrizes 10x10 - Teste t.*

| | Solução Plana | Solução Hierárquica |
|---|---|---|
| Média | 547796,20 | 339096,46 |
| Variância | 10328501961,00 | 21707424799,00 |
| Observações | 30 | 30 |
| gl | 52 | |
| Stat t | 6,38 | |
| P(T<=t) uni-caudal | 0,00 | |
| t crítico uni-caudal | 1,67 | |

**Tabela 4.29:** *Adaptação em matrizes 10x10 - Teste t.*

| | Solução Plana | Solução Hierárquica |
|---|---|---|
| Média | 1972076,11 | 1969332,60 |
| Variância | 50162629,93 | 4967498,94 |
| Observações | 30 | 30 |
| gl | 35 | |
| Stat t | 2,02 | |
| P(T<=t) uni-caudal | 0,02 | |
| t crítico uni-caudal | 1,68 | |

ximadamente $10^{37}$ vezes maior que o espaço de busca da matriz de tamanho 5x5 (veja Tabela 4.3) e $10^{132}$ vezes mais possibilidades na matriz de tamanho 10x10, onde justifica-se então a necessidade da solução hierárquica quando a complexidade aumenta exponencialmente.

O efeito na redução do custo foi observada tanto no experimento com números inteiros quanto no experimento com cores, ou seja, aumentar a complexidade dos objetos (unidimensional e tridimensional, respectivamente, não aumenta a complexidade da busca). A Tabela 4.30 apresenta uma síntese dos resultados comparativos apresentados anteriormente em relação à hipótese nula ($H_0$).

A hipótese secundária sugere melhorias qualitativas ao utilizar hierarquia de algoritmos genéticos. No entanto, tal hipótese foi corroborada apenas para as matrizes de tamanho 10x10 no experimento de cores, ou seja, neste caso específico, o aumento da complexidade dos objetos (de unidimensional para tridimensional) a serem arranjados topologicamente foi significativo.

A seguir, apresentaremos os melhores resultados obtidos em cada cenário, exibindo e discutindo o melhor caso coletado a partir do conjunto de 30 amostras em cada configuração de tamanho de matriz.

**Tabela 4.30:** *Síntese da análise estatística inferencial para $H_0$.*

| | Números Inteiros | | Cores | |
|---|---|---|---|---|
| | Adaptação | Custo | Adaptação | Custo |
| 5x5 | não rejeitada | não rejeitada | não rejeitada | não rejeitada |
| 7x7 | não rejeitada | rejeitada | não rejeitada | rejeitada |
| 10x10 | não rejeitada | rejeitada | rejeitada | rejeitada |

A qualidade da solução produzida é subjetiva, analisamos os resultados encontrados no cenário de números inteiros para matrizes de tamanho 5x5: a Tabela 4.31 apresenta a solução encontrada pelo algoritmo genético plano e a Tabela 4.32 apresenta a solução hierárquica.

**Tabela 4.31:** *Resultado 5x5 encontrado na solução plana para números inteiros.*

| | | | | |
|---|---|---|---|---|
| 8 | 6 | 4 | 2 | 1 |
| 10 | 9 | 7 | 5 | 3 |
| 12 | 11 | 15 | 14 | 13 |
| 16 | 17 | 19 | 21 | 23 |
| 18 | 20 | 22 | 24 | 25 |

**Tabela 4.32:** *Resultado 5x5 encontrado na solução hierárquica para números inteiros.*

| | | | | |
|---|---|---|---|---|
| 8 | 6 | 4 | 3 | 1 |
| 10 | 9 | 7 | 5 | 2 |
| 13 | 12 | 11 | 15 | 14 |
| 23 | 21 | 19 | 17 | 16 |
| 25 | 24 | 22 | 20 | 18 |

As duas soluções apresentam o mesmo valor de adaptação (264711,33), ou seja, as distâncias e dissimilaridades entre os objetos são as mesmas. Porém ao observar a Tabela 4.31 notam-se os extremos (1 e 25) na mesma lateral, ao passo que na Tabela 4.32 os extremos (1 e 25) encontram-se em cantos opostos.

As Tabelas 4.33 e 4.34 apresentam respectivamente a aptidão final das soluções plana e hierárquica para matrizes 7x7. Ambas as soluções obtiveram a mesma aptidão final (5903808,53) e novamente, a solução plana aloca os cantos de uma das laterais para os extremos (1 e 49), enquanto a solução hierárquica posicionou os extremos em cantos opostos, de forma a generalizar a organização topológica de forma mais intuitiva.

**Tabela 4.33:** *Resultado 7x7 encontrado na solução plana para números inteiros.*

| | | | | | | |
|---|---|---|---|---|---|---|
| 35 | 37 | 39 | 43 | 46 | 47 | 49 |
| 32 | 33 | 36 | 40 | 42 | 44 | 48 |
| 29 | 30 | 31 | 34 | 38 | 41 | 45 |
| 26 | 27 | 28 | 22 | 23 | 24 | 25 |
| 21 | 20 | 19 | 16 | 12 | 9 | 4 |
| 18 | 17 | 14 | 10 | 8 | 6 | 3 |
| 15 | 13 | 11 | 7 | 5 | 2 | 1 |

**Tabela 4.34:** *Resultado 7x7 encontrado na solução hierárquica para números inteiros.*

| | | | | | | |
|---|---|---|---|---|---|---|
| 49 | 48 | 45 | 43 | 39 | 37 | 35 |
| 47 | 44 | 42 | 40 | 36 | 33 | 32 |
| 46 | 41 | 38 | 34 | 31 | 30 | 29 |
| 25 | 26 | 27 | 28 | 22 | 23 | 24 |
| 21 | 20 | 19 | 16 | 12 | 9 | 4 |
| 18 | 17 | 14 | 10 | 8 | 6 | 3 |
| 15 | 13 | 11 | 7 | 5 | 2 | 1 |

Nas matrizes de tamanho 10x10, a solução hierárquica encontra um mínimo global de melhor qualidade. A solução plana obteve uma aptidão final (153844858,12) inferior a aptidão apresentada pela solução hierárquica (153770729,38). As Tabelas 4.35 e 4.36 apresentam, respectivamente, os resultados para as soluções plana e hierárquica.

**Tabela 4.35:** *Resultado 10x10 encontrado na solução plana para números inteiros.*

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 23 | 27 | 37 | 40 | 50 | 89 | 93 | 96 | 99 | 100 |
| 22 | 26 | 35 | 39 | 48 | 81 | 87 | 91 | 95 | 98 |
| 19 | 24 | 33 | 38 | 46 | 62 | 82 | 86 | 92 | 97 |
| 17 | 21 | 28 | 36 | 43 | 61 | 77 | 83 | 88 | 94 |
| 14 | 18 | 25 | 34 | 41 | 60 | 73 | 80 | 84 | 90 |
| 11 | 15 | 20 | 32 | 42 | 57 | 67 | 76 | 79 | 85 |
| 8 | 12 | 16 | 31 | 44 | 55 | 63 | 70 | 75 | 78 |
| 4 | 10 | 13 | 30 | 45 | 53 | 59 | 66 | 71 | 74 |
| 3 | 6 | 9 | 29 | 47 | 52 | 56 | 64 | 68 | 72 |
| 1 | 2 | 5 | 7 | 49 | 51 | 54 | 58 | 65 | 69 |

No contexto de matrizes com tamanho 10x10, a solução hierárquica encontrou uma permutação favorável ao mapa através dos números posicionados nos cantos, contribuindo com a organização topológica do restante da matriz.

**Tabela 4.36:** *Resultado 10x10 encontrado na solução hierárquica para números inteiros.*

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 3 | 4 | 7 | 12 | 16 | 23 | 29 | 30 | 34 |
| 2 | 6 | 10 | 13 | 17 | 22 | 28 | 31 | 35 | 37 |
| 5 | 9 | 15 | 18 | 21 | 26 | 32 | 36 | 38 | 43 |
| 8 | 14 | 19 | 24 | 27 | 33 | 39 | 42 | 45 | 47 |
| 11 | 20 | 25 | 40 | 41 | 44 | 46 | 48 | 49 | 50 |
| 51 | 52 | 53 | 55 | 57 | 60 | 61 | 76 | 81 | 90 |
| 54 | 56 | 59 | 62 | 68 | 74 | 77 | 82 | 87 | 94 |
| 58 | 63 | 65 | 69 | 75 | 80 | 83 | 86 | 91 | 96 |
| 64 | 66 | 70 | 73 | 79 | 84 | 88 | 92 | 95 | 99 |
| 67 | 71 | 72 | 78 | 85 | 89 | 93 | 97 | 98 | 100 |

No cenário de cores a solução plana alcançou a aptidão (qualidade) de 5297,50 e solução hierárquica 5293,42. A Figura 4.1 exibe a aptidão final do melhor caso obtido na solução plana e a Figura 4.2 exibe a aptidão final do melhor caso encontrado pela solução hierárquica.

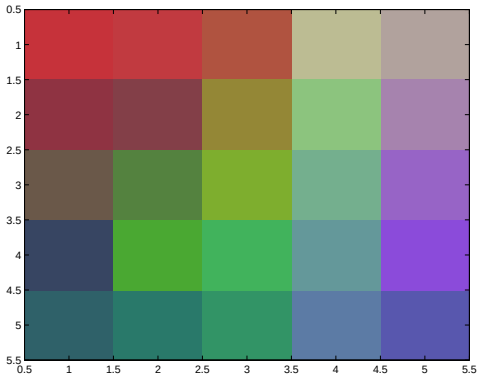


**Figura 4.1:** *Resultado 5x5 para a solução plana.*

Apesar da solução hierárquica ter obtido o menor custo de ajuste das cores, visualmente a solução plana organizou melhor os tons avermelhados, enquanto a solução hierárquica ajustou de forma otimizada os tons esverdeados, imprimindo um julgamento subjetivo de qual solução exibe a melhor organização topológica.

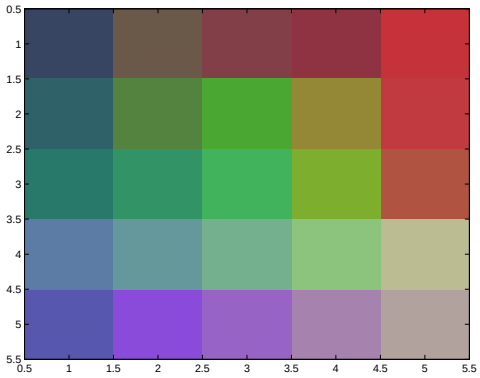


**Figura 4.2:** *Resultado 5x5 para a solução hierárquica.*

Para matrizes de tamanho 7x7, o melhor caso obtido na solução plana alcançou aptidão de 106623,01, enquanto o melhor caso da solução hieráquica obteve 106665,00. As Figuras 4.3 e 4.4 exibem os melhores casos de cada tipo de solução.

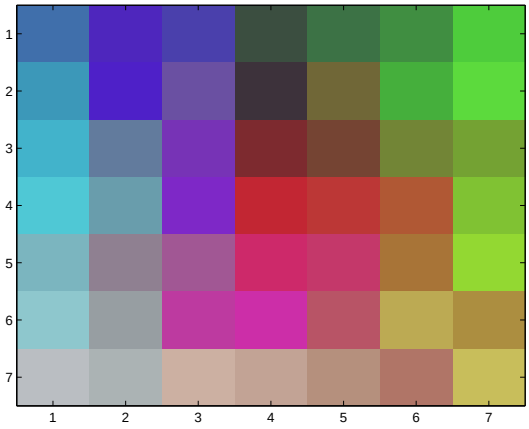


**Figura 4.3:** *Resultado 7x7 para a solução plana.*

A solução plana tratou melhor os tons esverdeados enquanto a solução hierárquica arranjou de forma otimizada os tons azuis. Apesar da solução plana ter sido matematicamente superior, analisando subjetivamente, o resultado da solução hierárquica parece mais organizado que a solução plana pois as células centrais da solução plana causam certo desconforto.

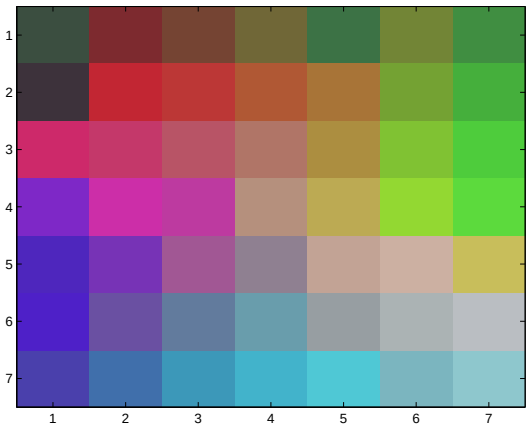


**Figura 4.4:** *Resultado 7x7 para a solução hierárquica.*

No cenário de cores, as matrizes de tamanho 10x10 foram submetidas às soluções plana e hierárquica. O valor mínimo alcançado pela solução plana em 30 casos foi de 1963917,98. A Figura 4.5 exibe o mapa topológico de tal caso. Dentre os 30 casos executados pela solução hierárquica, o valor mínimo foi 1964325,56. A Figura 4.6 apresenta o mapa topológico produzido pela solução hierárquica.

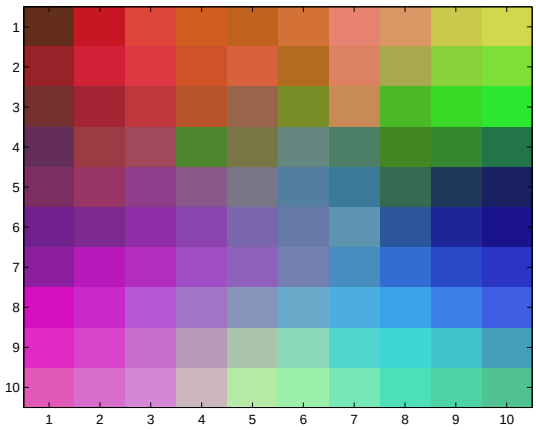


**Figura 4.5:** *Resultado 10x10 para a solução plana.*

Ambas as soluções tiveram dificuldade para tratar os tons esverdeados, mas analisando as cores mais avermelhadas, a solução hierárquica apresenta uma ordenação global mais harmoniosa quando comparada a solução plana.

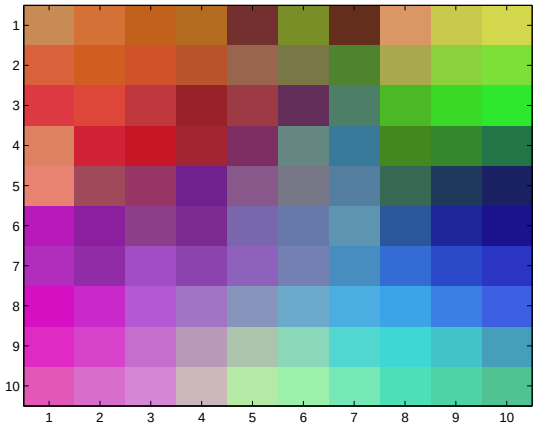


**Figura 4.6:** *Resultado 10x10 para a solução hierárquica.*

CAPÍTULO 5

# Conclusão

## 5.1 Considerações gerais

As meta-heurísticas formam uma classe importante de técnicas aplicáveis na resolução de problemas modernos, cada vez mais complexos, onde inexiste vasta informação a respeito. O sucesso da Computação Evolucionária deve ser atribuído à robustez da técnica, passível de modificações na heurística para adequá-la ao problema em questão. Neste sentido, o Sistema Proposto modificou o operador tradicional de cruzamento (*crossover*), a fim de acelerar a busca computacional por ótimos globais, no contexto de permutações topológicas. A variação, recombinação gênica, utilizou três novos operadores baseado em cruzamento interobjetos.

A hieraquia de Algoritmos Genéticos, aplicada ao Sistema Proposto, constitui-se de um algoritmo para domínio de parâmetros, situado no topo da hierarquia, e outro para o domínio de soluções, posicionado na base. O algoritmo genético do topo é capaz de adaptar-se ao ambiente e controlar, de forma eficiente, os parâmetros ajustáveis na população de soluções. Podemos definir essa capacidade como meta-adaptação, onde a observação da evolução dos indivíduos no ambiente guia o processo de otimização.

A validação empírica do modelo hierárquico proporcionou a corroboração parcial da hipótese principal, isto é, o modelo acelerou a busca de mapas topológicos apenas em grandes espaços de busca (iguais ou superiores a $10^{62}$). De forma geral a hipótese secundária foi rejeitada, pois a melhoria qualitativa dos mapas foi observada apenas no cenário de cores com matrizes 10X10. Sendo que em todos os outros casos, não houve melhoria. A qualidade apresentada foi equivalente a obtida no algoritmo sem hierarquia.

## 5.2 Principais contribuições

A pesquisa demonstrou aplicabilidade de hierarquia de algoritmos genéticos na organização de mapas topológicos. Os resultados validaram empiricamente o modelo proposto e confirmaram sua eficiência diante do problema de explosão combinatória. A aplicação da meta-heurística foi satisfatória, ao levar em consideração a inexistência de um método determinístico para busca de mapas topológicos de objetos multidimensionais.

As principais contribuições podem ser resumidas na redução do custo computacional, através da hierarquia de algoritmos genéticos, e na métrica da função de avaliação, descritas a seguir:

- A hierarquia de Algoritmos Genéticos acelera o Algoritmo Genético tradicional em problemas de explosão combinatória, diminuindo o custo computacional na busca de um ótimo global. Os resultados significativos foram observados em espaços de busca iguais ou superiores a $10^{62}$ possibilidades, no domínio de soluções.
- A métrica da função de avaliação, com custo modelado pelo parabolóide hiperbólico, generaliza a comparação qualitativa entre objetos descritos por $n$-dimensões, pertencentes a mesma classe. A métrica é aplicável a qualquer conjunto de objetos passível de ser representado no algoritmo genético; podendo, ainda, ser utilizado em outras heurísticas como colônia de formigas ou *particle swarm*.

## 5.3 Trabalhos futuros

Os experimentos não forneceram o ponto exato em que o uso de hierarquia em Algoritmos Genéticos torna-se eficiente, pois apenas algumas configurações foram utilizadas (matrizes 5x5, 7x7 e 10x10). Sabemos, contudo, que este ponto encontra-se no intervalo de matrizes 5x5 (aproximadamente $10^{25}$ possibilidades) até matrizes 7x7 (aproximadamente $10^{62}$ possibilidades). Para determinação de tal ponto, em trabalhos futuros, sugerimos a utilização de outras formas geométricas para os mapas topológicos, bem como o preenchimento parcial da matriz de resultados, ou seja, densidade inferior a 100%.

O Sistema Proposto utilizou esquemas de variação derivados da recombinação entre objetos do mapa, denominado cruzamento interobjetos (um único pai gera um novo descendente). Para um estudo futuro, sugere-se comparação direta com a pesquisa de [Basalaj 2001], onde utilizou-se o operador de variação do tipo “crossover cíclico”.

As contribuições do presente trabalho permitem aplicações diretas em catálogos eletrônicos, utilizados em *e-commerce*. A busca de produtos, numa determinada categoria, é acelerada pela visualização topológica dos objetos, portanto há redução no tempo de escolha do produto desejado.

Acreditamos que a técnica, utilizada no Sistema Proposto, é aplicável a problemas de segmentação de dados. Após encontrar classes, por meio da segmentação, é possível otimizar a visualização dos dados, sem o eventual efeito da sobreposição de resultados.

Numa visão futurista, a limitação bidimensional na visualização dos mapas pode ser substituída por uma esfera tridimensional. A organização topológica estaria distribuída por toda a superfície da esfera e ao rotacioná-la seria possível observar todos os objetos.

## 5.4 Mensagem Final

Através das técnicas de metodologia científica, o trabalho possibilitou a verificação de suma importância de todas as etapas de uma pesquisa científica, desde a delimitação do tema até a avaliação final de resultados, obtendo significativas conclusões. Tal trabalho contribuirá imensamente em minha vida acadêmica, tanto na docência quanto em novas pesquisas. Agradeço, principalmente, a dedicação do orientador da pesquisa e amigo, Prof. PhD Weber Martins, por todo o empenho, atenção e dicas fundamentais para a conclusão do trabalho.

# Referências Bibliográficas


(Basalaj 2001) BASALAJ, W. *Proximity Visualisation of Abstract Data*. Tese (Doutorado), 2001. Disponível em: <http://wwwpavis.org>.

(Bayer 1971) BAYER, R. Binary b-trees for virtual memory. In: CODD, E. F.; DEAN, A. L. (Ed.). *Proceedings of 1971 ACM-SIGFIDET Workshop on Data Description, Access and Control, San Diego, California, November 11-12, 1971*. (S.l.): ACM, 1971. p. 219–235.

(Bergeron, Mixtacki e Stoye 2005) BERGERON, A.; MIXTACKI, J.; STOYE, J. On sorting by translocations. *Research in Computational Molecular Biology*, v. 3500/2005, p. 615–629, 2005.

(Braga, Ludermir e Carvalho 2000) BRAGA, A.; LUDERMIR, T.; CARVALHO, A. *Redes Neurais Artificiais: Teoria e Aplicações*. (S.l.): LTC, 2000.

(Chen 1996) CHEN, J. Proportion split sort. *Nordic Journal of Computing*, v. 3, n. 3, p. 271–279, 1996.

(Cole 1988) COLE, R. Parallel merge sort. *SIAM Journal on Computing*, v. 17, n. 4, p. 770–785, 1988.

(Cook 2007) COOK, T. E. ACM Press, New York, USA, p. 2647–2650, 2007.

(Darwin 1906) DARWIN, C. A. *On the Origin of Species by Means of Natural Selection*. (S.l.): Sixth London Edition, 1906.

(Eiben e Smith 2003) EIBEN, A. E.; SMITH, J. E. *Introduction to Evolutionary Computing Genetic Algorithms*. (S.l.): Springer, 2003.

(Euler 1736) EULER, L. Geometriam situs. 1736.

(Everett 2001) EVERETT, J. E. Algorithms for multidimensional scaling. In: CHAMBERS, L. (Ed.). *The Practical Handbook of Genetic Algorithms*. (S.l.): Chapman & Hall, 2001. v. 1.

(Fogel 1995) FOGEL, D. B. *Evolutionary Computation: Toward a New Philosophy of Machine Intelligence*. (S.l.): IEEE Press, 1995.

(Fogel e Ghozeil 1997) FOGEL, D. B.; GHOZEIL, A. A note on representations and variation operators. *IEEE Transactions on Evolutionary Computation*, v. 1, n. 2, p. 159–161, 1997.

(Fogel e Michalewics 1999) FOGEL, D. B.; MICHALEWICS, Z. *How to Solve It: Modern Heuristics*. (S.l.): Springer, 1999.

(Goldberg 1989) GOLDBERG, D. E. *Genetic Algorithms in Search, Optimization and Machine Learning*. Massachusetts: Addison-Wesley, 1989.

(Haykin 2001) HAYKIN, S. S. *Redes Neurais Artificiais - princípios e prática*. Porto Alegre: Bookman, 2001.

(Hoare 1962) HOARE, C. A. R. Quicksort. *Computer Journal*, v. 5, n. 1, p. 10–15, 1962.

(Holland 1975) HOLLAND, J. *Adaption in Natural and Artificial Systems*. (S.l.): University of Michigan Press, 1975.

(Kelley 1975) KELLEY, J. L. *General Topology*. (S.l.): Springer-Verlag, 1975.

(Kim e André 2004) KIM, S.; ANDRé, E. A generate and sense approach to automated music composition. ACM Press, New York, USA, p. 268–270, 2004.

(Kohonen 1989) KOHONEN, T. *Self-organizing Maps*. Berlin, Germany: Springer-Verlag, 1989.

(Lakatos e Marconi 1991) LAKATOS, E. M.; MARCONI, M. A. *Fundamentos de Metodologia Científica*. São Paulo: Editora Atlas, 1991. 212 p.

(Melanie 1998) MELANIE, M. *An Introduction to Genetic Algorithms*. (S.l.): The MIT Press, 1998.

(Moraes 2006) MORAES, L. L. *Catálogos Inteligentes com Preservação Topológica para Comércio Eletrônico*. Goiânia: Universidade Federal de Goiás, Escola de Engenharia Elétrica e de Computação, Programa de Pós-Graduação em Engenharia Elétrica e de Computação, Grupo Pireneus, 2006.

(Munkres 1999) MUNKRES, J. *Topology*. (S.l.): Prentice Hall, 1999.

(Randolph 2002) RANDOLPH, E. B. *Choice and the Internet: From clickstream to Research Stream*. (S.l.): Spring, 2002.

(Rawlins 1991) RAWLINS, G. *Foundations of Genetic Algorithms*. (S.l.): Morgan Kaufmann, 1991.

(Rechenberg 1994) RECHENBERG, I. *Evolutionsstrategie*. Stuttgart: Frommann-Holzboog, 1994.

(Reeves 1995) REEVES, C. R. *Genetic Algortihms*. Berkshire: McGraw-Hill Book Company, 1995.

(Schwefel 1995) SCHWEFEL, H.-P. *Evolution and Optimum Seeking*. New York: Wiley & Sons, 1995.

(Shell 1959) SHELL, D. L. A high-speed sorting procedure. *Communications of the ACM*, v. 2, n. 7, p. 30–32, 1959.

(Silva e Liblik 2004) SILVA, F. S. S.; LIBLIK, A. M. P. *O desenho das crianças de 6 a 8 anos: os aspectos cognitivos das primeiras noções topológicas e suas representações*. Curitiba: Universidade Federal do Paraná, Setor de Educação, Programa de Pós-Graduação em Educação., 2004.

(Werneck e Damiani 2005) WERNECK, N.; DAMIANI, F. Espaços homogêneos de parâmetros para controle de efeitos musicais. II Seminário de Música, Ciência e Tecnologia, São Paulo, 2005.

(Whitley 1994) WHITLEY, D. A genetic algorithm tutorial. *Statistics and Computing*, v. 4, p. 65–85, 1994.

(Williams 1964) WILLIAMS, J. W. J. Algorithm 232: Heapsort. *Communications of the ACM*, v. 7, n. 6, p. 347–348, 1964.

APÊNDICE A

# Mapas Auto-Organizáveis

## A.1 Mapa de cores de Kohonen para Internet

A presente seção apresenta uma implementação dos Mapas Auto-Organizáveis de Kohonen (escrita em linguagem script PHP) para Internet. Os mapas de cores gerados por tal implementação foram aplicados ao Algoritmo Genético do Sistema Proposto, no cenário de cores, tendo suas posições embaralhadas pelo AG para gerar a população inicial. O mapa de Kohonen também serviu como parâmetro para avaliar a qualidade da solução encontrada pelo sistema proposto.

A linguagem PHP roda no servidor, possibilitando o acesso à aplicação a partir de qualquer navegador (*browser*) de Internet a partir do protocolo HTTP, desde que a aplicação esteja corretamente configurada para ser executada num servidor WWW.

A Figura A.1 exibe a tela de opções iniciais do Mapa de Cores for Web.

A primeira opção permita a escolha do tamanho do mapa, com valores pré-definidos de tamanhos 3x3, 5x5, 7x7, 10x10 e 15x15. O campo quantidade de cores, indica o número de entradas a serem utilizadas durante o treinamento, contemplando as opções de 15, 50, 100, 150 e 200 cores, escolhidas aleatoriamente. As taxas de aprendizado inicial e raio da vizinhaça são ajustáveis na inicialização do SOM, assim como a opção de inicializar o mapa com pesos randômicos nas sinapses. As duas últimas opções referem-se a exibição dos resultados. A rede neural pode exibir um resultado parcial do treinamento a cada $n$ ciclos conforme exemplo na Figura A.2. A última opção permite o ajuste do tamanho do neurônio em *pixels*.

Ao final do treinamento (cujo critério de parada é a quantidade de ciclos), a rede neural apresenta as taxas finais de aprendizado e ajuste fino da vizinhança, exibindo os vetores de peso de cada sinapse.

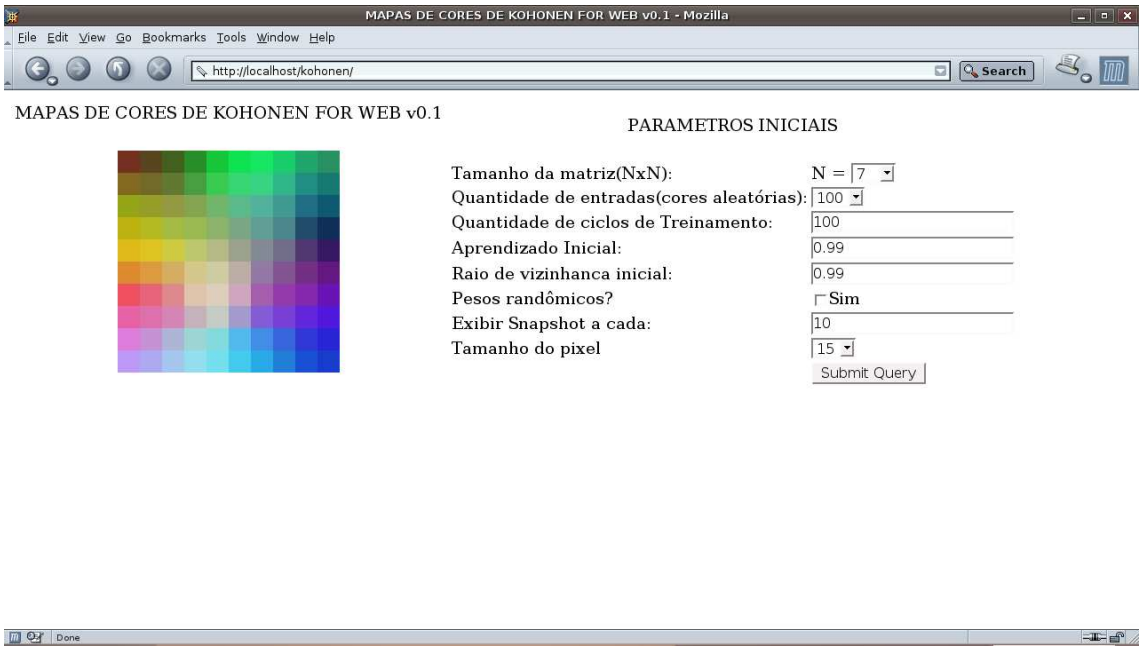


**Figura A.1:** *Opções iniciais para o Mapa de Kohonen for Web v0.1.*

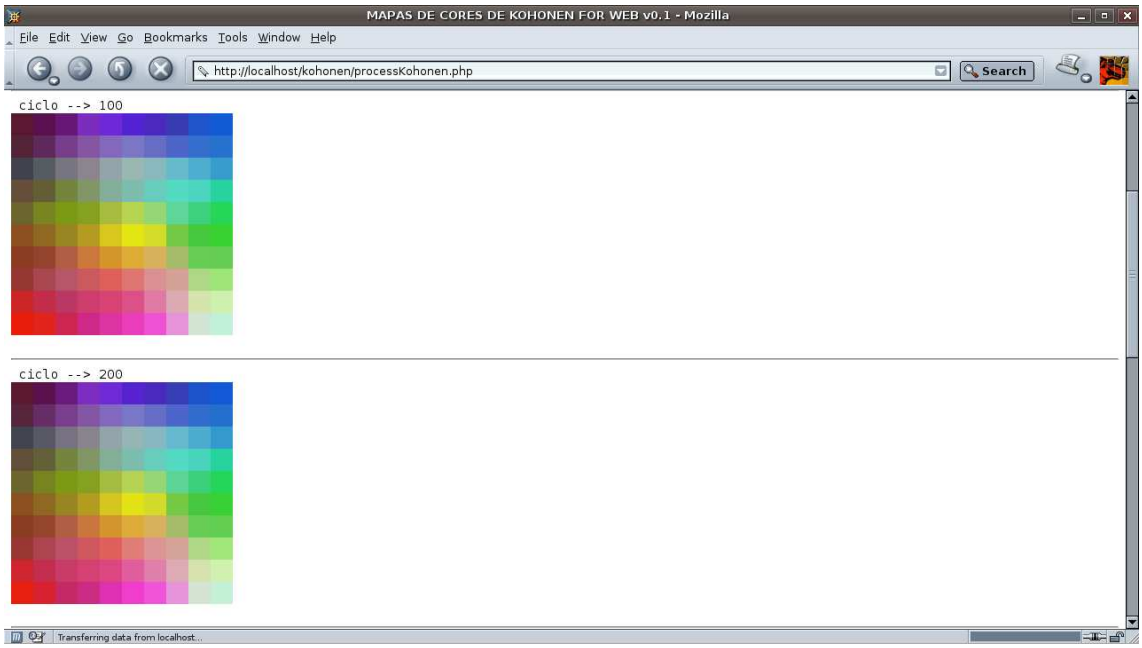


**Figura A.2:** *Evolução a cada 100 ciclos de treinamento.*

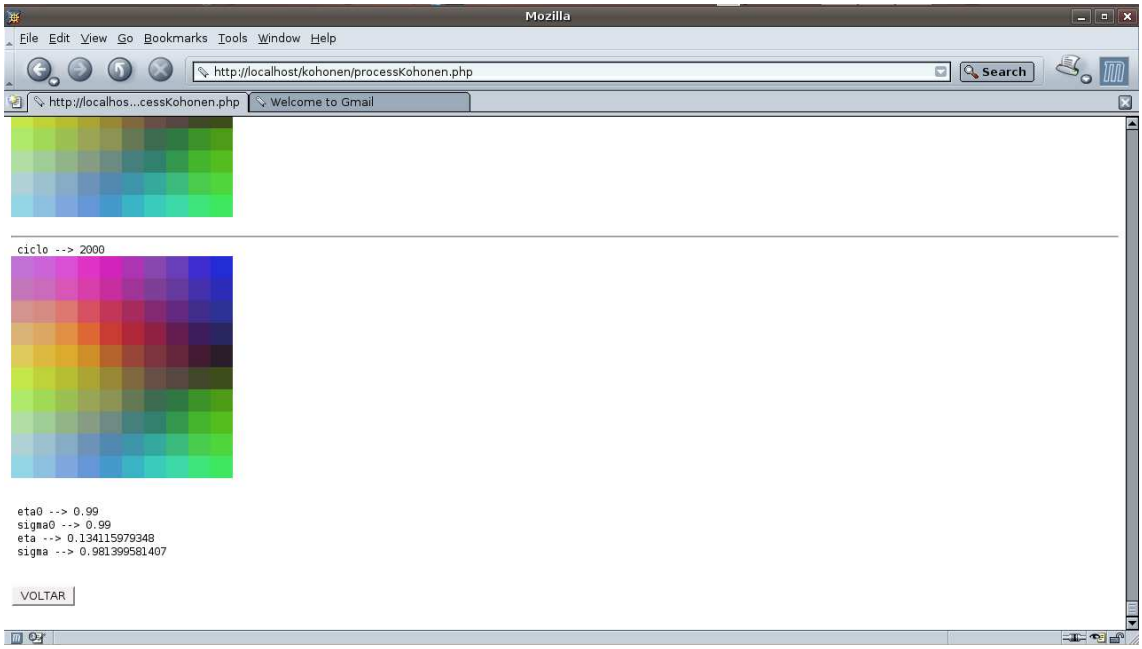


**Figura A.3:** *Mapa resultante após 2000 ciclos de treinamento.*

APÊNDICE B

# Topological Ordering of Objects Based On Hierarchies of Genetic Algorithms

## B.1 Abstract

Objects in real world are usually complex entities with many attributes, being represented as instances of n-dimensional data. When human beings (and machines) need to search huge multidimensional datasets, sorting (and its maintenance) ease the search process. Despite the fact that traditional sorting is just a case of topological n-dimensional ordering, Computer Science has been historically interested only on one-dimensional sorting, that is, on sorting based on a single key attribute. On the other hand, Soft Computing (non traditional Artificial Intelligence) typical approach to deal with topological ordering, Kohonen's Self-Organizing Maps (SOM), does not work properly for this task either. Many real-world applications (such as virtual libraries, e-commerce catalogues, web search engines etc), however, must present many specific n-dimensional objects in two-dimensional displays where users should find desired objects with efficiency. In the context of Multidimensional Data Scaling methods, this problem has been introduced as "proximity grids" with 100% density, and Genetic Algorithms have emerged as the best solution among several methods. In this work, it is proposed an alternative based on hierarchies of Genetic Algorithm with meta adaptation. Empirical results have shown that the proposed system is faster to converge when compared to the non-hierarchical version.

## B.2 Introduction

This work is concerned with ordering algorithms and data visualization of predefined sets of objects. We present a system based on hierarchies of Genetic Algorithms to organize these specific objects (instances of multidi-

mensional data) topologically in order to ease the search process. This situation is found at electronic catalogues of virtual shops (e-commerce) [12] and at some other scenarios where specific object searching (virtual libraries, results of web search engines etc) is needed. It is important to highlight the unique features of the problem from the most important viewpoints: Soft Computing and Computer Science.

In the context of Soft Computing, Kohonen's Self-Organizing Maps [10] are the typical approach when topological ordering is required. However, the topological organization of specific objects is not properly addressed by them (since their prototypes do not match the training data, but they model the training data sources statistically). For example, if a Kohonen map is employed to organize some specific colours (which are usually represented by three-dimensional data), it will frequently end up with different colours as prototypes, not the ones used at the training process. The issue of the presented work is the topological organization of specific predefined objects. In terms of Kohonen maps, prototypes must be the specific objects used in training. Therefore, Kohonen maps have been only a great motivation to develop a proper alternative to approach the problem of interest.

Historically, Computer Science is interested on one-dimensional sorting, that is, on sorting based on a single key attribute (quick sort [8], heapsort [17] and shellsort [15]). Even when k key attributes are employed, a priority order is predefined and each key attribute is considered one at a time. Sorting is needed (and justified) by future recalling of stored data. The maintenance of a sorted data structure is capital to search efficiency and many different structures have been proposed to this functionality (symmetric binary B-trees [2]). This work has dealt with a generalized version of the traditional one-dimensional sorting, since a sorted vector is also the most topological permutation of members for any proper metric (where the distances among neighboring items are minimized). Figure B.1 shows an example of this argument.

## B.3 Background

### B.3.1 Proximity Grid

The original problem of topological ordering of objects was defined in [1] as "Proximity Grids" by Basalaj. The problem is described in the context of Multidimensional Scaling (MDS) arrangements for browsing image

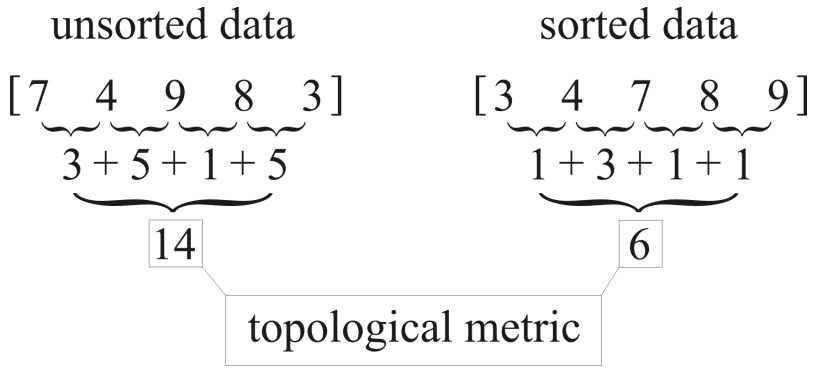


**Figura B.1:** *Topological metric on unsorted and sorted vectors.*

collections. The overlap of image thumbnails [14] in MDS arrangements has been reported as a problem when a target image should be found. “Proximity Grids” bring the grid regularity with the topological arrangement provided by the MDS algorithms. In his experiment, many results (based on 53 data collections: two dissimilarity matrices, two image collections, six graphs, and 43 data tables) are presented for different optimization algorithms applied in several densities. According to this study, based on Inferential Statistical Analysis, Genetic Algorithms have achieved the best results on 100% density grids when compared to Greedy, Improved Greedy and “Squeaky Wheel” optimization which have performed better on lower density grids [13]. While Basalaj has faced the problem as an MDS arrangement on a regular grid, this work has treated it as a permutation problem.

### B.3.2 Genetic Algorithms

Genetic Algorithms [16] are the most traditional member of the Evolutionary Algorithms family, which is inspired at the Evolutionary Theory [3], famous for its parallel nature [11]. The idea behind this family is quite simple: an initial population of candidate solutions is submitted to selection, and variation operators (crossover and mutation) are employed on each new offspring. The environment models the problem situation in such a way that the quality of candidate solutions is directly related to the environmental adaptation of the individuals. A fitness function measures the degree of adaptation and, often, it is the most time-consuming part of the code. In other words, each individual in the population represents a candidate solution, so every individual is evaluated by a fitness function that defines how good this specific individual is as a solution in the problem domain. As in Biology, the best candidate solutions (population individuals) receive the highest fitness values. To model the

individual complexity, genes codify n-dimensional individual characteristics, sometimes in m-to-1 relations, depending on the nature of the specific feature.

Variation is responsible for discovering new solutions, and selection points to which individuals survive and build the basis for further exploration [7]. Better individuals have higher chances for being selected by the "crossover" operator. Crossover is the main variation operator and produces a new individual that resembles their parents but its relative importance depends on the problem nature. Descendants are, then, submitted to the "mutation" operator. The mutation operator is applied over the "genes" according to a mutation ratio probability. This form of random noise attempts to prevent the Genetic Algorithm to be stuck in local minima (or maxima, depending on the problem) [9].

Permutation defines a special class of problems for Genetic Algorithms. Slightly different from the famous Traveling Salesman Problem [6], another permutation problem, the proposed system does not intend to find the lowest cost salesman's path but its target is the best objects permutation in a grid with respect to the topological ordering. However, GA default operators perform poorly on permutation problems since they usually generate a lot of unfeasible solutions after the crossover operator [5]. Therefore, the success of Genetic Algorithms at the permutation scenarios depends on clever variation schemes based on individual codification and on the problem features. This strategy is at the core of the proposed system.

## B.4 Research Developments

### B.4.1 Reference Model - Flat Genetic Solution

Despite the fact that Basalaj [1] has found that Flat Genetic Solution was the best method for the 100% density proximity grid problem, his development has employed MDS concepts like energy, stress/misfit functions and so on. To provide a fair comparison in our research, it was modelled another flat genetic solution from scratch. This solution is the basis for (a component to) the hierarchical genetic solution, the proposed system. Therefore, the developed flat genetic solution will be described first.

As for the cost function, one of the main contribution of this work, the hyperbolic paraboloid has emerged as the natural solution for the problem requirements. In mathematical terms, the cost function (to be minimized by

the optimization process), $f(P)$, quality of the whole grid permutation, $P$, is calculated by Equation B-1.

$$f(P) = \sum_{i=1,j=1}^{n} cost_{ij} \tag{B-1}$$

where $P$ is an $n$ x $n$ permutation, and (see Equation B-2) is the particular cost contribution of cell in the $i$-th line and $j$-th column, ($i$, $j$).

$$cost_{ij} = -\sum_{l=1,c=1}^{n} \left\{ \left( X_{ij,lc} - \frac{X_{max}}{2} \right) \left( Y_{ij,lc} - \frac{Y_{max}}{2} \right) + \frac{X_{max} Y_{max}}{4} \right\} \tag{B-2}$$

where $X_{ij,lc}$ is the Euclidean distance between the grid positions of cells ($i$, $j$) and ($l$, $c$), and $Y_{ij,lc}$ is the Euclidean distance between the contents of cells ($i$, $j$) and ($l$, $c$). $X_{max}$ is the maximum distance between any two cells in the grid, that is, the diagonal length. $Y_{max}$ is the maximum dissimilarity between any two cell contents which depends, obviously, on the specific set of $n$-dimensional contents (objects). Therefore, while grid positions are two-dimensional data, specified by row and column, grid contents can be an $n$-dimensional data.

To understand the real meaning of Equation B-2, it should be remembered that close cells with similar contents are rewarded as well as distant cells with dissimilar contents. On the other hand, some punishment is needed to deal with close cells with dissimilar contents and distant cells with similar contents. These requirements are shown in Figure B.2.

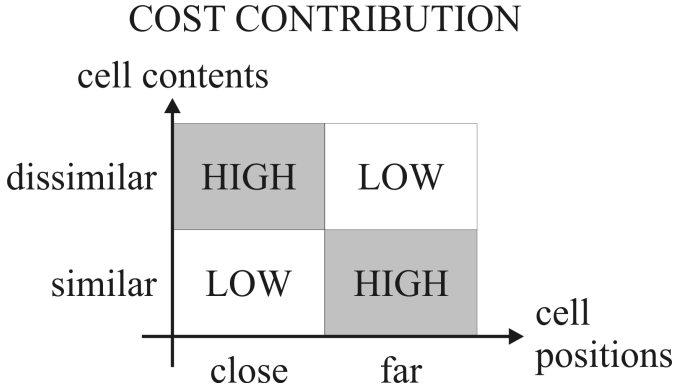


**Figura B.2:** *Cost contribution from comparing a pair of cells.*

In a three-dimensional visualization, the particular cost contribution of any two cells defines a Hyperbolic Paraboloid (see Figure B.3), that is, a surface basically defined by $z = x.y$ in its pure form. Once cost minimization is

sought, the surface is inverted by the introduction of a minus sign. Moreover, it is also moved from its typical position in the distance-dissimilarity plane from the origin (0,0) to ($\frac{X_{max}}{2}$,$\frac{Y_{max}}{2}$) and a constant term is introduced ($\frac{X_{max}Y_{max}}{4}$) to guarantee the minimal cost equals to zero.

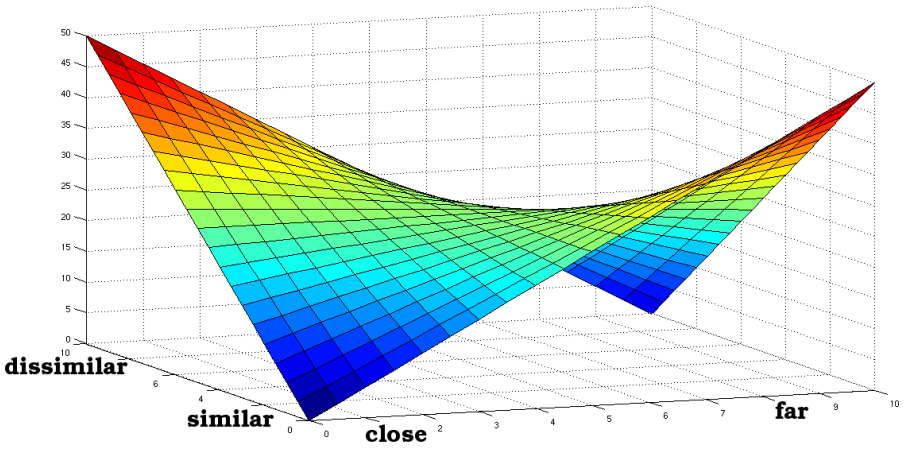


**Figura B.3:** *Three-dimensional visualization of cost contribution from a pair of cells.*

In this flat genetic population (which will be called “solution domain” in the hierarchical solution), individuals are instances of grid permutations (solution candidates). Each solution candidate evolves by changing its content (object) positions. In order to define variation and selection operators to be applied to the population, three aspects must be addressed:

- The variation operator (which variation operator should be applied),
- The specific locus (where, the precise cell position, this variation operator should be applied),
- The individual selection operator (to define which individuals/grids should be modified).

As for the variation operator, there are three possible values which are chosen at random:

- Swap permutation (where two colors/genes exchange their positions),
- Three-gene swap permutation (where three colors exchange their positions),
- Area permutation (where colors randomly exchange their positions within a randomly chosen rectangular area),

Only the swap permutation, frequently associated with a mutation strategy, has been found in literature [4].

The specific locus (application place), central position, for the variation operator is selected by three different strategies with the same probability:

- At random,
- Deterministically (by calculating the cell with the highest cost contribution to the whole permutation cost), and
- Probabilistically (by using the individual cost contribution of each cell as its probability of being selected).

Finally, the selection of individuals to produce offsprings (new permutations) is conducted in two different modes which are also selected at random:

- Probabilistically (based on the topological quality of each solution candidate), and
- By contrast (where the individual cell cost is normalized to the range from zero to one).

It should be noted that an offspring is generated from only one permutation (parent) to avoid unfeasibility and poor convergence due to local minima. In other words, there is no traditional crossover (genetic exchange between any two individuals). Elitism is realized by saving the best half of the parents and the best half of the offsprings. This strategy aims to balance the effect of newer and older generations.

### B.4.2 Proposed System - Hierarchical Genetic Solution

The proposed system expands the flat solution by controlling its parameters with another Genetic Algorithm. It is based, therefore, on hierarchies of Genetic Algorithms with meta adaptation. As there are many choices to be made with respect to how the solution should be pursued (and they are chosen at random in the flat solution), a hierarchy of two levels of genetic optimization is defined: solution and parameter optimization levels. While a GA is working in the solution domain (as it has been described at the previous section), another GA looks for proper parameters in order to accelerate the search process and to improve the quality of problem solutions. This dynamics is shown in Figure B.4.

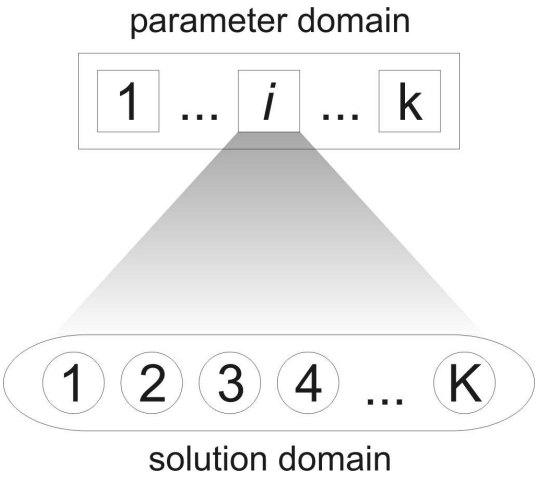


**Figura B.4:** *Dynamics of the Proposed System based on Hierarchies of Genetic Algorithms.*

The population in the parameter domain (higher population) represents instances of variation and selection operators that are applied on the solution population (lower population). The proposed system uses meta adaptation since it looks for parameters that facilitates the search process in another level. The most successful individuals in the higher population will reach the best fitness degree when the most adapted individuals in the lower population are produced by following its parameter values.

To clarify the main ideas and set up the path for empirical experiments, without any loss of generality, the proposed system could be presented in the scenario of the color sorting problem: given a set of colors (three-dimensional data) and a grid with the same number of elements, what is the most topological ordered permutation?

In the parameter domain, in order to define variation and selection operators to be applied in the solution domain, three genes are employed to represent, as stated before, the variation operator, the specific application locus, and the individual selection operator. The main difference between reference and proposed models is that decisions were taken at random and by genetic optimization respectively.

In contrast with the solution domain, traditional (uniform random) crossover takes place at the parameter domain. Offsprings inherit each particular gene from one of their parents that has been chosen at random.

At the parameter domain, mutation is empirically derived as 5%, typically higher than usual choice. This choice is justified by two arguments: chromosomes (gene sequences) are not long (only three genes), and the need to keep sufficient diversity at the solution domain to avoid premature convergence.

There is no elitism in the parameter domain, but the new population at the solution domain is built by equal contributions of the work that each set of parameters (individual in the parameter domain) has accomplished. Theses contributions are composed by the best offsprings. For example, if the parameter domain is composed by four individuals and the solution domain population has 32 individuals, each of the four (parameter) individuals generates 32 offsprings and selects its best eight ones to integrate the future 32-individual solution population.

## B.5 Experiments

In order to validate empirically the proposed system, 30 computational simulations were undertaken in each experimental condition (problem domain, resolution and solution alternative). In other words, samples have been originated from 30 computational simulations for each one of the problem domains (integer numbers, one-dimensional data, and colors, three-dimensional data) in each resolution (5x5, 7x7, and 10x10). The main goal is to compare the solution alternatives: the traditional (flat) genetic solution, TS, and proposed (hierarchical) genetic solution, HS.

Combinatorial explosions - frequent phenomena in permutation problems - are related directly to the increasing of resolution (grid dimensions). Just to be explicit, the search space for 5x5, 7x7 and 10x10 resolutions are 25!, 49! and 100! respectively (see Table B.1).

**Tabela B.1:** *Resolution (grid dimensions) and Search space size.*

| Resolution | Search space size |
|---|---|
| 3 x 3 | $9! > 3.5 \text{ x } 10^{5}$ |
| 4 x 4 | $16! > 2.0 \text{ x } 10^{13}$ |
| 5 x 5 | $25! > 1.5 \text{ x } 10^{25}$ |
| 6 x 6 | $36! > 3.7 \text{ x } 10^{41}$ |
| 7 x 7 | $49! > 6.0 \text{ x } 10^{62}$ |
| 8 x 8 | $64! > 1.2 \text{ x } 10^{89}$ |
| 9 x 9 | $81! > 5.7 \text{ x } 10^{120}$ |
| 10 x 10 | $100! > 9.3 \text{ x } 10^{157}$ |

In 5x5 and 7x7 resolutions, there have been 64 individuals in the solution level (population) and eight individuals in the parameter level (population). The stop criterion (derived empirically) is 300,000 offsprings/births,

since each new permutation has its associated cost calculated. In the resolution of 10x10, simulations is finished after 700,000 offsprings/birth, there are 32 individuals in the solution level (population), and the parameter level (population) has four individuals.

Among many aspects that could be studied, besides the solution alternatives (flat or hierarquical), the focus lies on the following variables:

- Solution cost (already defined by Equation B-1): negatively correlated with solution quality, that is, as the solution cost decreases, the solution quality increases and vice-versa.
- Convergence time: number of individual evaluations that is needed to reach the global solution in the genetic optimization process.

Back to the hypotheses, when the hierarquical solution is compared to the flat solution, it is expected that solution cost is lower, and convergence time is lower too.

## B.6 Results

This section presents empirical results to compare the reference model (flat genetic solution) to the proposed system (hierarchy of Genetic Algorithms) in both problem domains (integer numbers and colors). Statistical (descriptive and inferential) analysis has been employed to organize and generalize these results. In particular, statistical parametric t-Student test is chosen to compare these two alternative solutions.

### B.6.1 Descriptive Statistical Analysis

The integer numbers problem has been adopted as the initial point since it represents the topological ordering issue in the simplest form. Objects are one-dimensional data, and Euclidian distances are two-number differences. Table B.2 shows the main descriptive statistics of solution cost while Table B.3 is related to the convergence time.

In the integer numbers problem, as the solution cost indicates the quality of the optimization process and is negatively correlated with the solution quality, in descriptive terms, the reference model has produced, in average, better solutions than the proposed system in 5x5 and 7x7 resolutions. On the other hand, the proposed system has performed better in the highest resolution, 10x10, when the search space is also increased. Moreover, relative

mean differences have reduced from 5x5 to 7x7 resolutions and have reached its maximum at 10x10 grids. This fact justifies the tendency of the proposed system to find better solutions as the search space increases. In terms of convergence time, the superiority of the proposed system over the reference model is clear even for the lower resolution, 5x5, and it has gained momentum as to higher resolutions.

**Tabela B.2:** *Descriptive Statistics of Solution Cost in the Integer Numbers Problem*

| Resolution | 5 x 5 | | 7 x 7 | | 10 x 10 | |
|---|---|---|---|---|---|---|
| Statistic \ Solution | Reference | Proposed | Reference | Proposed | Reference | Proposed |
| Mean | 264810.35 | 264820.63 | 5908383.32 | 5908515.02 | 153959905.32 | 153920178.11 |
| Standard Error | 20.38 | 20.13 | 716.87 | 659.82 | 16916.70 | 18747.92 |
| Median | 264802.56 | 264802.56 | 5907131.24 | 5907100.36 | 153935595.75 | 153899879.52 |
| Standard Deviation | 111.64 | 110.28 | 3926.44 | 3614.00 | 92656.61 | 102686.56 |
| Kurtosis | 1.12 | 0.83 | 0.68 | 2.59 | (0.19) | 0.01 |
| Asymmetry | 1.19 | 1.04 | 1.29 | 1.49 | 0.89 | 0.53 |
| Range | 423.18 | 423.18 | 13277.35 | 16562.86 | 326668.91 | 382211.27 |
| Minimum | 264711.33 | 264711.33 | 5903808.54 | 5903808.54 | 153844858.13 | 153770729.38 |
| Maximum | 265134.51 | 265134.51 | 5917085.88 | 5920371.40 | 154171527.03 | 154152940.65 |
| Cases | 30 | 30 | 30 | 30 | 30 | 30 |

In the colors problem, objects are three-dimensional data and, therefore, they specify a more complex situation. Table B.4 summarizes descriptive statistics of solution cost while Table B.5 deals with the convergence time. In the colors problem, the solution cost has shown an unexpected behavior since the proposed system has lower values for 5x5 and 10x10 grids, and, therefore, clearly suggests the use of inferential techniques. As for the convergence time, the proposed system has performed better than the reference model although it has lost some strength when compared to the context of integer numbers problem. This last observation can be clearly seen from comparison of Figures B.5 and B.6. Note that each bar is the mean of 30 simulations.

In graphical format, it is interesting to register the evolution of usage of variation and individual selection operators along with the specific locus modes to discuss it later. In the integer numbers context, Figure B.7 presents the evolution of the variation operator usage in the process of convergence for all resolutions, 5x5, 7x7, and 10x10. Figure B.8 has the same structure but has its focus in the colors problem.

First, it is remarkable the importance of the simple variation operator in all six situations. It has never been the least used variation operator and

**Tabela B.3:** *Descriptive Statistics of Convergence Time in the Integer Numbers Problem*

| Resolution | 5 x 5 | | 7 x 7 | | 10 x 10 | |
|---|---|---|---|---|---|---|
| Statistic \ Solution | Reference | Proposed | Reference | Proposed | Reference | Proposed |
| Mean | 41974.33 | 40103.40 | 201884.73 | 88180.20 | 593682.07 | 154883.13 |
| Standard Error | 3822.28 | 9017.58 | 7350.93 | 6448.71 | 12484.46 | 19583.04 |
| Median | 36033.00 | 26689.00 | 200385.00 | 76609.00 | 593233.00 | 113377.00 |
| Standard Deviation | 20935.49 | 49391.31 | 40262.72 | 35321.05 | 68380.23 | 107260.71 |
| Kurtosis | 5.63 | 13.93 | 0.38 | 12.02 | -0.42 | 7.14 |
| Asymmetry | 2.23 | 3.75 | 0.65 | 3.28 | -0.35 | 2.68 |
| Range | 95552 | 237056 | 160064 | 175104 | 271712 | 448512 |
| Minimum | 20417 | 15425 | 137857 | 63553 | 426785 | 68513 |
| Maximum | 115969 | 252481 | 297921 | 238657 | 698497 | 517025 |
| Cases | 30 | 30 | 30 | 30 | 30 | 30 |

has finished as the most employed variation operator in four times (66%). The three-gene operator, on the other hand, is the overall loser but it has ended on the top two positions in the highest resolution on both problems. One could also observe there is high diversity when both problems are compared. This fact reassures the value of the hierarchical optimization because it is more flexible than the flat process.

The evolution of application locus operator in the integer numbers context can be seen in Figure B.9 while Figure B.10 focuses in the colors problem.

First, it is remarkable the importance of the random application in all six situations. It finished as the most employed variation operator in five out of six situations. The deterministic application locus, on the other hand, deserves the runner-up mention. In particular, it has a strong contribution on the toughest situation (10x10 in the colors problem). It must be highlighted the strong relevance of the deterministic application operator in the early stages of the genetic optimization in all situations, and the progressively increase of the random application.

Finally, the evolution of usage of selection operator is presented in Figure 9 (integer numbers problem) and Figure 10 (colors problem) in the process of convergence for all resolutions, 5x5, 7x7, and 10x10. The impressive result is the usage similar rates for both forms of selection: probabilistically and by contrast.

As for the best results, they can be seen in Figures B.13 and B.14 for the integer numbers and colors problems respectively. It is not easy to

**Tabela B.4:** *Descriptive Statistics of Solution Cost in the Colors Problem*

| Resolution | 5 x 5 | | 7 x 7 | | 10 x 10 | |
|---|---|---|---|---|---|---|
| Statistic \ Solution | Reference | Proposed | Reference | Proposed | Reference | Proposed |
| Mean | 5347.16 | 5343.07 | 107745.39 | 107822.34 | 1972076.11 | 1969332.61 |
| Standard Error | 6.06 | 8.67 | 138.84 | 156.42 | 1293.09 | 406.92 |
| Median | 5344.76 | 5338.84 | 107740.64 | 107511.95 | 1969925.70 | 1969375.38 |
| Standard Deviation | 33.19 | 47.51 | 760.45 | 856.77 | 7082.56 | 2228.79 |
| Kurtosis | -0.03 | -1.35 | 0.68 | -0.29 | 1.49 | -0.38 |
| Asymmetry | 0.40 | 0.36 | 0.92 | 0.93 | 1.52 | -0.03 |
| Range | 125.05 | 145.30 | 3124.40 | 3063.94 | 24848.83 | 8864.13 |
| Minimum | 5297.51 | 5293.43 | 106623.02 | 106665.01 | 1963917.98 | 1964325.57 |
| Maximum | 5422.56 | 5438.73 | 109747.41 | 109728.95 | 1988766.82 | 1973189.70 |
| Cases | 30 | 30 | 30 | 30 | 30 | 30 |

compare all cells at once but one could focus on the corners to conclude that the proposed system has consistently placed the minimum and maximum values on opposite corners in the integer numbers context. On the other hand, the striking resemblance between solutions has to be considered along with the fact that the proposed system has found their solutions much quicker than the reference model.

**Tabela B.5:** *Descriptive Statistics of Convergence Time in the Colors Problem*

| Resolution | 5 x 5 | | 7 x 7 | | 10 x 10 | |
|---|---|---|---|---|---|---|
| Statistic \ Solution | Reference | Proposed | Reference | Proposed | Reference | Proposed |
| Mean | 49460.20 | 40615.40 | 191026.07 | 121153.00 | 547796.20 | 339096.47 |
| Standard Error | 9240.89 | 8185.02 | 10489.96 | 12131.36 | 18554.88 | 26899.46 |
| Median | 29057.00 | 29249.00 | 194049.00 | 89921.00 | 573121.00 | 298209.00 |
| Standard Deviation | 50614.43 | 44831.22 | 57455.89 | 66446.20 | 101629.24 | 147334.40 |
| Kurtosis | 5.05 | 21.86 | -1.32 | 0.42 | -1.19 | -0.37 |
| Asymmetry | 2.41 | 4.46 | 0.24 | 1.23 | -0.25 | 0.62 |
| Range | 194496.00 | 245248.00 | 179392.00 | 238592.00 | 355552.00 | 530176.00 |
| Minimum | 17025.00 | 16449.00 | 114113.00 | 54337.00 | 335969.00 | 139553.00 |
| Maximum | 211521.00 | 261697.00 | 293505.00 | 292929.00 | 691521.00 | 669729.00 |
| Cases | 30 | 30 | 30 | 30 | 30 | 30 |

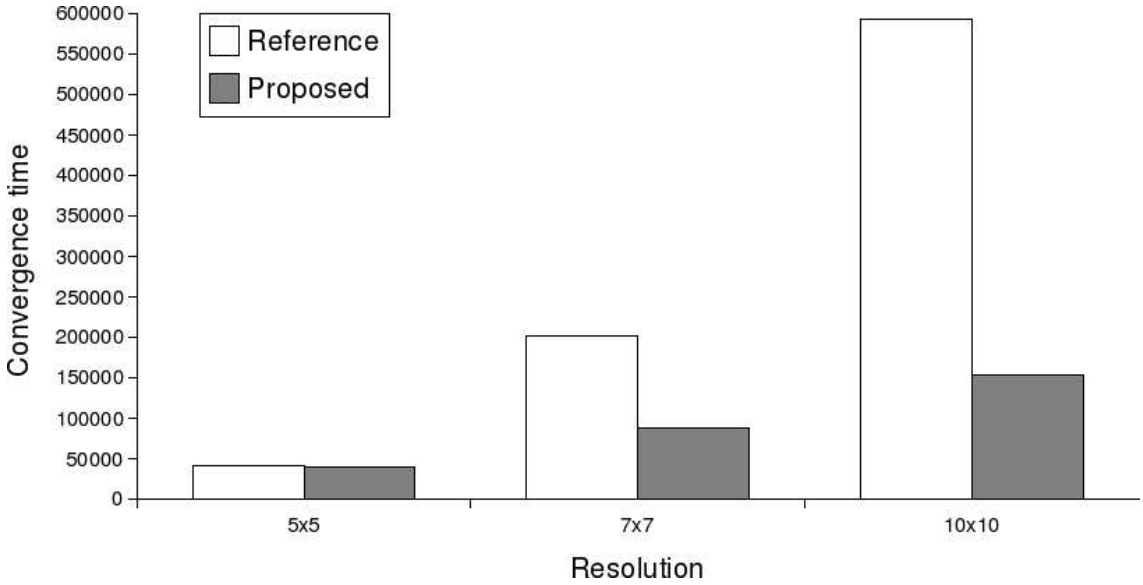


**Figura B.5:** *Mean convergence time in integer numbers problem.*

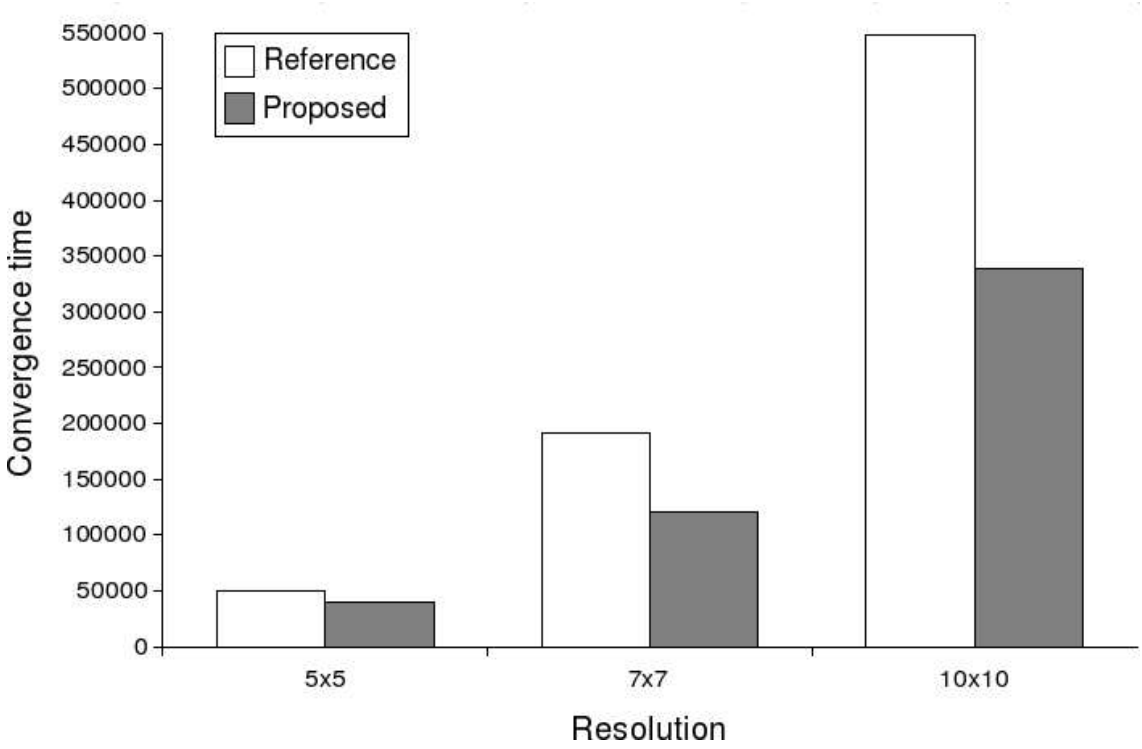


**Figura B.6:** *Mean convergence time in color problem.*

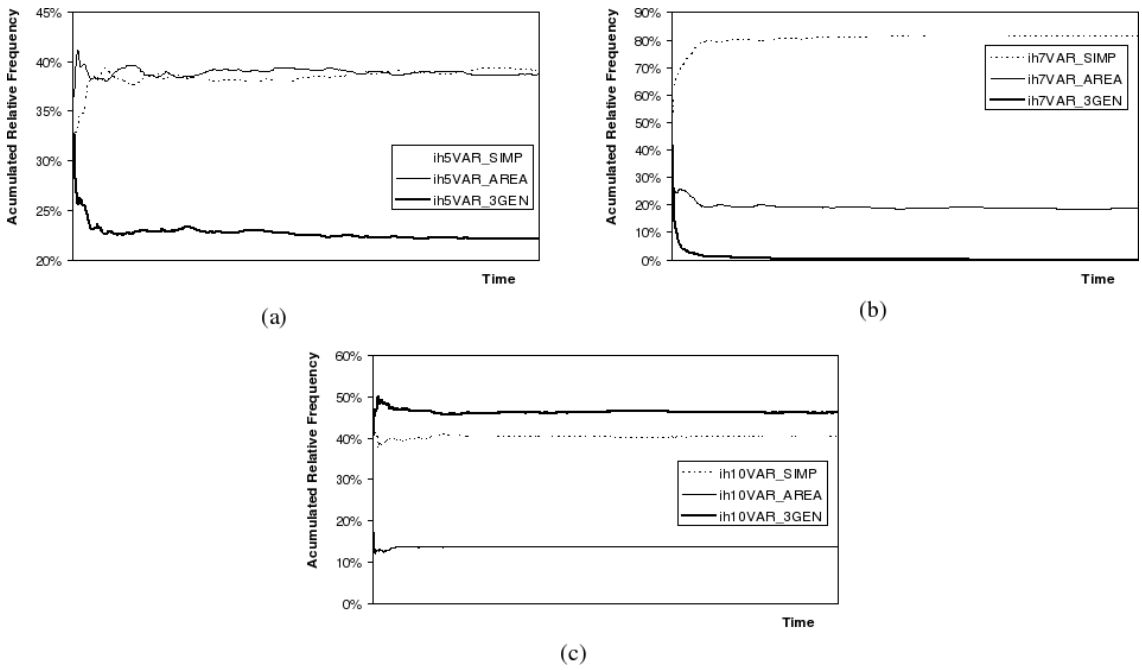


**Figura B.7:** *Evolution of variation operator usage in the integer numbers problem to: (a) 5x5, (b) 7x7, and (c) 10x10.*

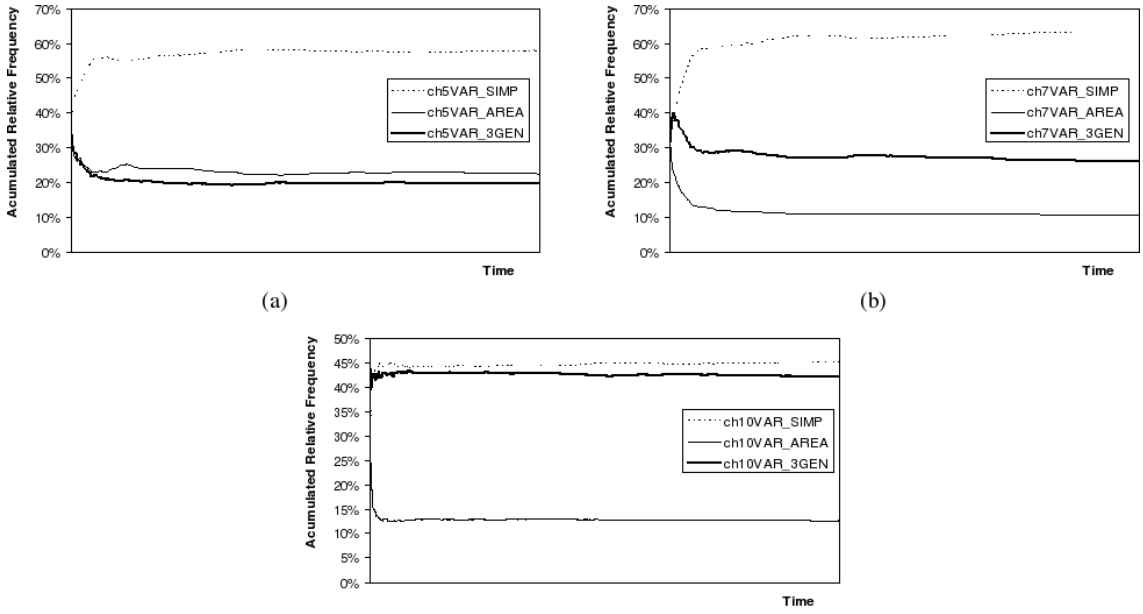


**Figura B.8:** *Evolution of variation operator usage in the colors problem to: (a) 5x5, (b) 7x7, and (c) 10x10.*

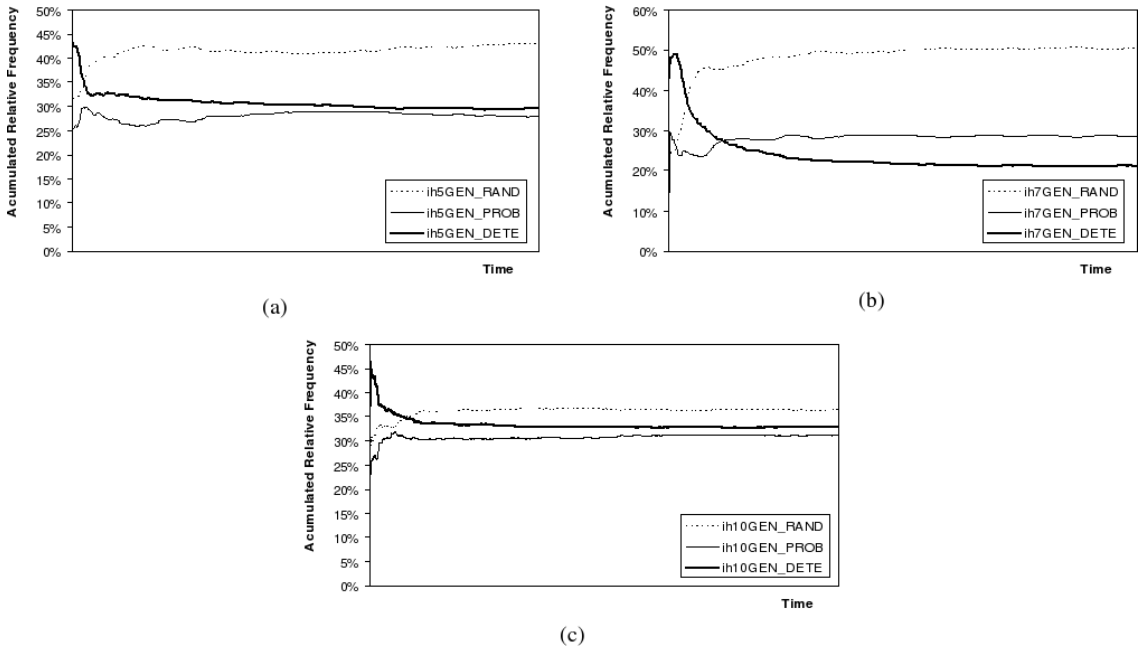


**Figura B.9:** *Evolution of application locus usage in the integer numbers problem to: (a) 5x5, (b) 7x7, and (c) 10x10.*

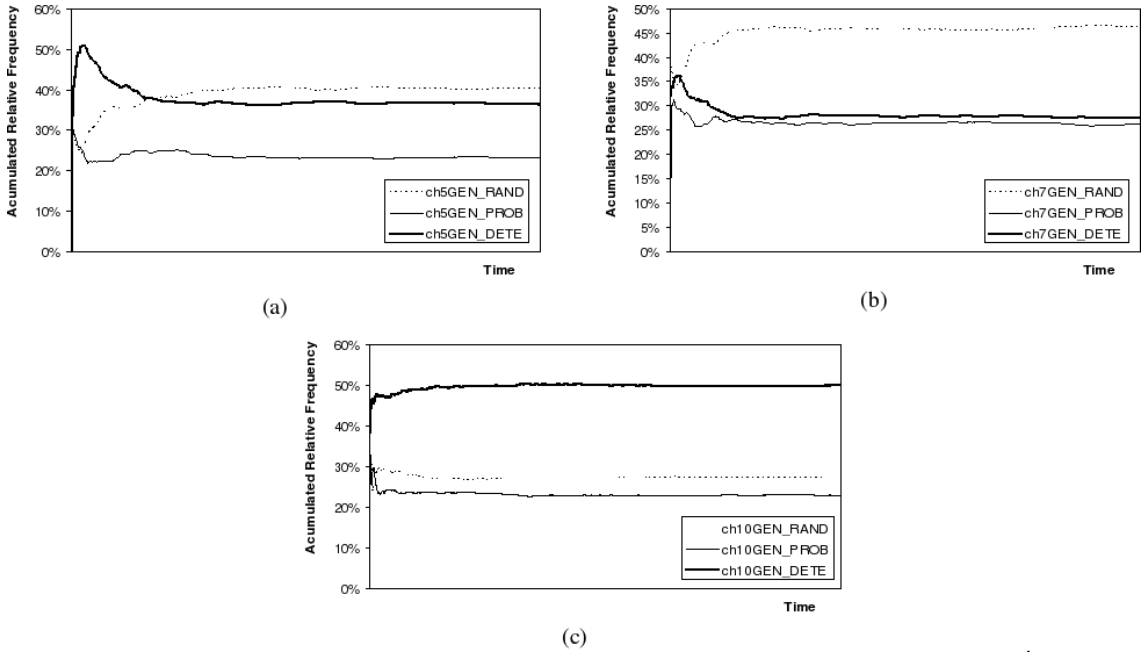


**Figura B.10:** *Evolution of application locus usage in the colors problem to: (a) 5x5, (b) 7x7, and (c) 10x10.*

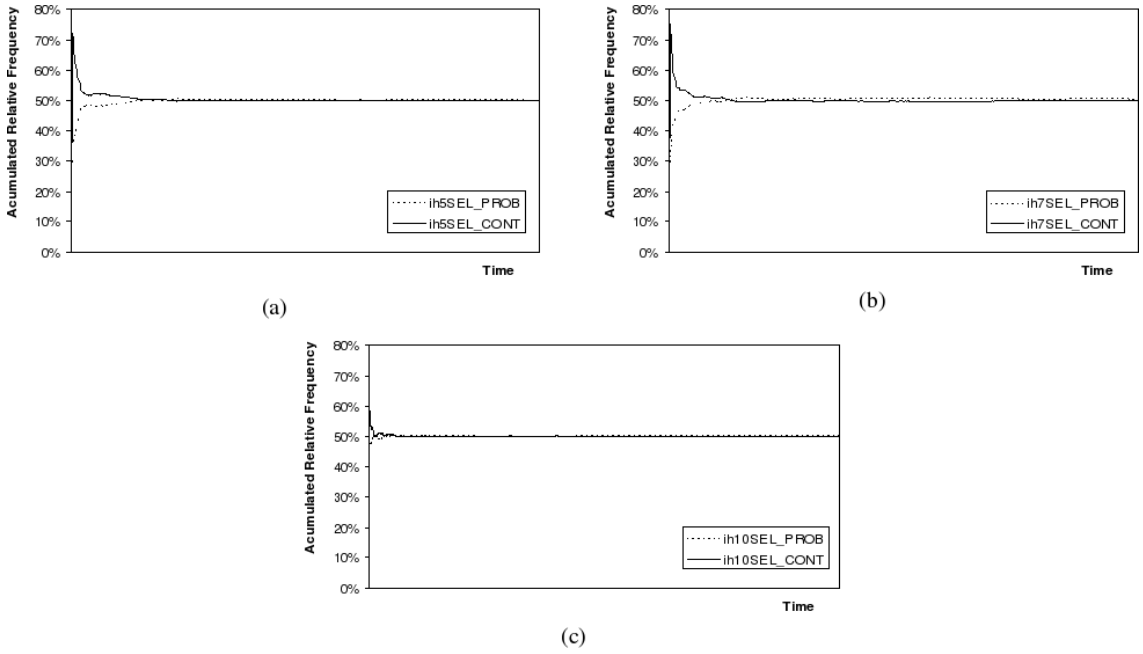


**Figura B.11:** *Evolution of selection usage in the integer numbers problem to: (a) 5x5, (b) 7x7, and (c) 10x10.*

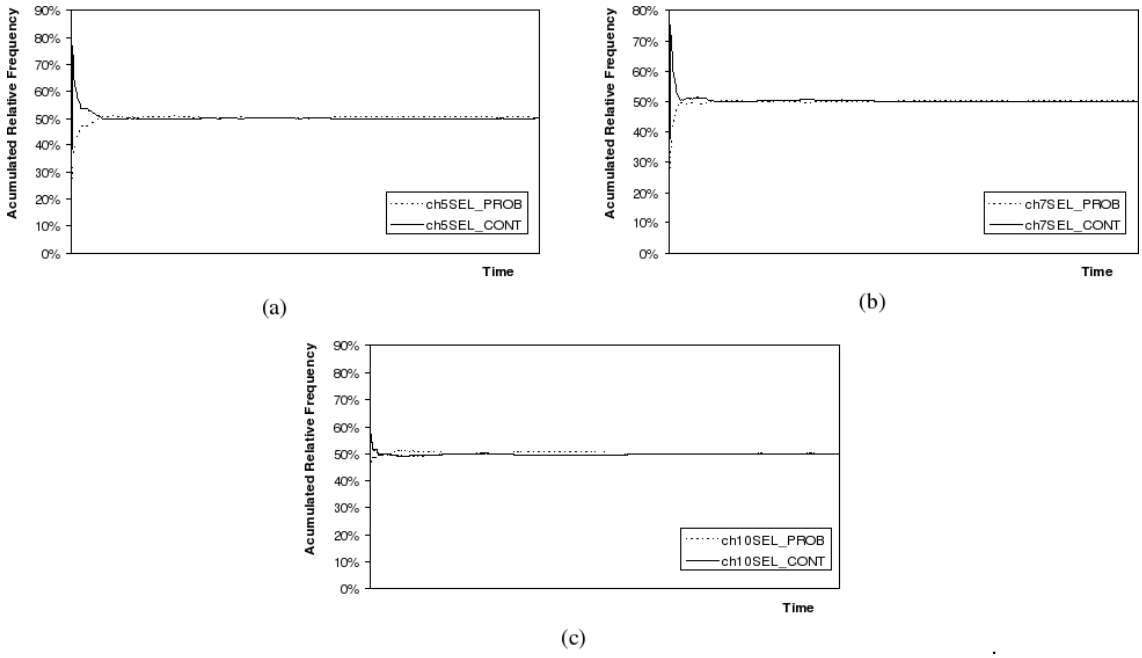


**Figura B.12:** *Evolution of selection usage in the colors problem to: (a) 5x5, (b) 7x7, and (c) 10x10.*

Reference

| | | | | |
|---|---|---|---|---|
| 1 | 3 | 13 | 23 | 25 |
| 2 | 5 | 14 | 21 | 24 |
| 4 | 7 | 15 | 19 | 22 |
| 6 | 9 | 11 | 17 | 20 |
| 8 | 10 | 12 | 16 | 18 |

Proposed

| | | | | |
|---|---|---|---|---|
| 1 | 2 | 14 | 16 | 18 |
| 3 | 5 | 15 | 17 | 20 |
| 4 | 7 | 11 | 19 | 22 |
| 6 | 9 | 12 | 21 | 24 |
| 8 | 10 | 13 | 23 | 25 |

(a)

Reference

| | | | | | | |
|---|---|---|---|---|---|---|
| 1 | 3 | 4 | 25 | 45 | 48 | 49 |
| 2 | 6 | 9 | 24 | 41 | 44 | 47 |
| 5 | 8 | 12 | 23 | 38 | 42 | 46 |
| 7 | 10 | 16 | 22 | 34 | 40 | 43 |
| 11 | 14 | 19 | 28 | 31 | 36 | 39 |
| 13 | 17 | 20 | 27 | 30 | 33 | 37 |
| 15 | 18 | 21 | 26 | 29 | 32 | 35 |

Proposed

| | | | | | | |
|---|---|---|---|---|---|---|
| 1 | 3 | 4 | 24 | 29 | 32 | 35 |
| 2 | 6 | 9 | 23 | 30 | 33 | 37 |
| 5 | 8 | 12 | 22 | 31 | 36 | 39 |
| 7 | 10 | 16 | 28 | 34 | 40 | 43 |
| 11 | 14 | 19 | 27 | 38 | 42 | 45 |
| 13 | 17 | 20 | 26 | 41 | 44 | 48 |
| 15 | 18 | 21 | 25 | 46 | 47 | 49 |

(b)

Reference

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 5 | 7 | 49 | 51 | 54 | 58 | 65 | 69 |
| 3 | 6 | 9 | 29 | 47 | 52 | 56 | 64 | 68 | 72 |
| 4 | 10 | 13 | 30 | 45 | 53 | 59 | 66 | 71 | 74 |
| 8 | 12 | 16 | 31 | 44 | 55 | 63 | 70 | 75 | 78 |
| 11 | 15 | 20 | 32 | 42 | 57 | 67 | 76 | 79 | 85 |
| 14 | 18 | 25 | 34 | 41 | 60 | 73 | 80 | 84 | 90 |
| 17 | 21 | 28 | 36 | 43 | 61 | 77 | 83 | 88 | 94 |
| 19 | 24 | 33 | 38 | 46 | 62 | 82 | 86 | 92 | 97 |
| 22 | 26 | 35 | 39 | 48 | 81 | 87 | 91 | 95 | 98 |
| 23 | 27 | 37 | 40 | 50 | 89 | 93 | 96 | 99 | 100 |

Proposed

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 5 | 8 | 11 | 51 | 54 | 58 | 64 | 67 |
| 3 | 6 | 9 | 14 | 20 | 52 | 56 | 63 | 66 | 71 |
| 4 | 10 | 15 | 19 | 25 | 53 | 59 | 65 | 70 | 72 |
| 7 | 13 | 18 | 24 | 40 | 55 | 62 | 69 | 73 | 78 |
| 12 | 17 | 21 | 27 | 41 | 57 | 68 | 75 | 79 | 85 |
| 16 | 22 | 26 | 33 | 44 | 60 | 74 | 80 | 84 | 89 |
| 23 | 28 | 32 | 39 | 46 | 61 | 77 | 83 | 88 | 93 |
| 29 | 31 | 36 | 42 | 48 | 76 | 82 | 86 | 92 | 97 |
| 30 | 35 | 38 | 45 | 49 | 81 | 87 | 91 | 95 | 98 |
| 34 | 37 | 43 | 47 | 50 | 90 | 94 | 96 | 99 | 100 |

(c)

**Figura B.13:** *Best results in the integer numbers problem to: (a) 5x5, (b) 7x7, and (c) 10x10.*

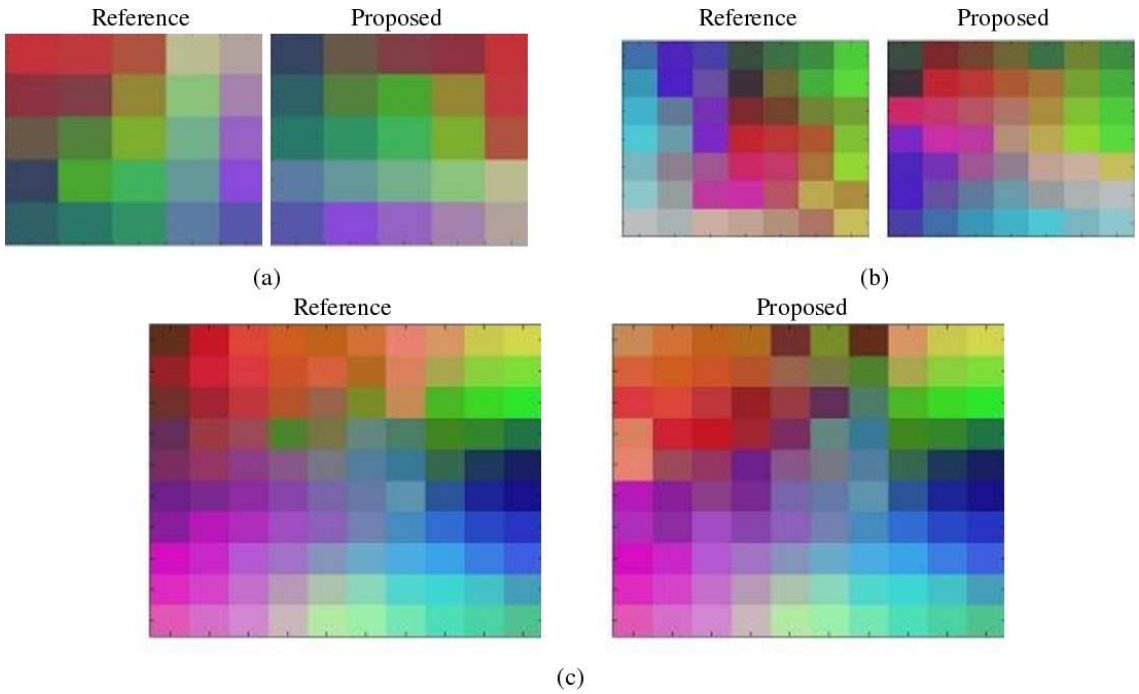


**Figura B.14:** *Best results in the colors problem to: (a) 5x5, (b) 7x7, and (c) 10x10.*

### B.6.2 Inferential Statistical Analysis

To generalize results to the information sources of the collected data samples, Inferential Statistical techniques have been employed, in particular, hypothesis test (t-Student test) that leads to the p-value (probability that the observed t value or even more extreme t values happen under the null hypothesis which states there is no significant difference between the reference (flat) and the proposed (hierarchical) solutions. As a 5% significance level has been adopted, Table B.6 synthesizes these results by presenting the p-values and by pointing their significance (when p-value is lower than the predefined significance level). The tendency of the proposed system to reach better solutions with lower convergence time is clear again since the number of significant sample differences increases as the problem and search space increase.

**Tabela B.6:** *Inferential Statistics of Results*

| | Solution Cost | | | Convergence Time | | |
|---|---|---|---|---|---|---|
| Problem \ Resolution | 5 x 5 | 7 x 7 | 10 x 10 | 5 x 5 | 7 x 7 | 10 x 10 |
| Integer Numbers | 0.3605 | 0.4465 | 0.0606 | 0.4247 | **$5.61 \times 10^{-17}$** | **$1.91 \times 10^{-24}$** |
| Colors | 0.3501 | 0.3571 | **0.0253** | 0.2383 | **$2.78 \times 10^{-5}$** | **$2.34 \times 10^{-8}$** |

### B.6.3 Discussion

Based on the statistical analysis above, it can be argued that:

- If the dimensionality of the problem increases progressively, the relation between convergence time of flat and hierarchical solutions does increase progressively too. This fact indicates that , in more complex situations where search spaces are higher, the hierarchical approach tends to overcome the flat solution more clearly. The hierarchical solution imposes greater diversity to populations related to the parameter space than the flat one due to the application of distict variations operators along the subprocesses of evolution. Therefore, the probability of premature convergence is reduced, essential feature in conceiving an evolutionary algorithm.
- In terms of solution cost and if the stopping criteria is taken into consideration, it could be realized that both algorithms have shown similar results. However, it should be noted, based on the inferential

statistical analysis (see Table B.6), that the hierarchical algorithm starts to perform significantly better than the flat alternative to problems with higher dimensionality.

- In this work, it is highlighted that the hierarchical approach presents itself as the most efficient than the flat approach as the search space increases. It could be enphasized that real problems, most of the time, have huge search spaces and, additionally, demand quick solution. By following this rationale, the proposed hierarchical algorithm has situated as a potential tool to solve problems with these features.

## B.7 Conclusions

This article has presented a novel approach to the problem of 100% - density proximity grid based on hierarchical Genetic Algorithms. The process of designing and implementing were revised as well as an experiment where a topological colour map (obtained by Kohonen self-organizing algorithm) serves as the best solution. Comparing to the flat Genetic Algorithm, the hierarchical model is better since less individual evaluations were needed in order to find the best solution. Moreover, the best fitness of the hierarchical model is closer to the Kohonen (overall best) fitness than the one from the flat solution.

Future work is aimed at using crossover operators suited to permutation problems at the lower level population. Other types of data, specially from products in e-commerce sites, will be used to experiment with the consumer satisfaction in front of topological ordered products in the sense of facilitate his decision process.

## B.8 References

[1] W. Basalaj. Proximity Visualisation of Abstract Data. PhD thesis, Trinity College Cambridge, 2001.

[2] Rudolf Bayer. Binary b-trees for virtual memory. In E. F. Codd and A. L. Dean, editors, Proceedings of 1971 ACM-SIGFIDET Workshop on Data Description, Access and Control, San Diego, California, November 11-12, 1971, pages 219-235. ACM, 1971.

[3] Charles A. Darwin. On the Origin of Species by Means of Natural Selection. 6th London Edition, 1906.

[4] A. E. Eiben and J. E. Smith. Introduction to Evolutionary Computing Genetic Algorithms. Springer, 2003.

[5] D. B. Fogel and A. Ghozeil. A note on representations and variation operators. IEEE Transactions on Evolutionary Computation, 1(2):159-161, 1997.

[6] D. B. Fogel and Z. Michalewics. How to Solve It: Modern Heuristics. Springer, 1999.

[7] D. E. Goldberg. Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley, Massachusetts, 1989.

[8] C. A. R. Hoare. Quicksort. Computer Journal, 5(1):10-15, 1962.

[9] J. Holland. Adaption in Natural and Artificial Systems. University of Michigan Press, 1975.

[10] T. Kohonen. Self-organizing Maps. Springer-Verlag, Berlin, Germany, 1989.

[11] Z. Michalewicz. Genetic Algorithms + Data Structures = Evolution Programs. Springer-Verlag, 1999.

[12] E. B. Randolph. Choice and the Internet: From clickstream to Research Stream. Spring, 2002.

[13] K. Rodden, W. Basalaj, D. Sinclair, and K. Wood. Evaluating a visualisation of image similarity as a tool for image browsing. IEEE Information Visualisation, pages 36-43, 1999.

[14] K. Rodden, W. Basalaj, D. Sinclair, and K. Wood. Does organisation by similarity assist image browsing? CHI '01: Proceedings of the SIGCHI conference on Human factors in computing systems, 2001.

[15] D. L. Shell. A high-speed sorting procedure. Communications of the ACM, 2(7):30-32, 1959.

[16] D. Whitley. A Genetic Algorithm tutorial. Statistics and Computing, 4:65-85, 1994.

[17] J. W. J. Williams. Algorithm 232: Heapsort. Communications of the ACM, 7(6):347-348, 1964.

APÊNDICE C

# Cálculo do custo distância x dissimilaridade

## C.1 Exemplo do cálculo do custo distância x dissimilaridade

O presente apêndice destina-se a exemplificar o trecho do Código 3.3, exibindo a evolução das iterações para o primeiro item da matriz $ind$.

```
Variando ia de 1 a 5 e ib de 1 a 5 teremos:
adaptacao = 0
ia = 1
ib = 1
ind(1,1) = 17

Variando ic de 1 a 5 e id de 1 a 5 teremos:
Para ic = 1 e id = 1:
ind(1,1) = 17
dist = 0
md = 0
valorMDmax = 5.6569
valorDDmax = 16
custo = 0
totalGene = 0

Para ic = 1 e id = 2:
ind(1,2) =  24
dist = 7
md = 1
valorMDmax = 5.6569
```

```
valorDDmax = 16
custo = 20.7990
total = 20.7990

Para ic = 1 e id = 3:
ind(1,3) = 1
dist = 16
md = 2
valorMDmax = 5.6569
valorDDmax = 16
custo =  29.2548
total = 50.0538

Para ic = 1 e id = 4:
ind(1,4) = 8
dist = 9
md = 3
valorMDmax = 5.6569
valorDDmax = 16
custo = 22.4558
total = 94.9655

Para ic = 1 e id = 5:
ind(1,5) = 15
dist = 2
md = 4
valorMDmax = 5.6569
valorDDmax = 16
custo = 29.6569
total = 131.8234

Para ic = 2 e id = 1:
ind(2,1) = 23
dist =  6
md = 1
valorMDmax = 5.6569
valorDDmax = 16
custo = 18.9706
```

```
total = 140.1076

Para ic = 2 e id = 2:
ind(2,2) = 5
dist = 12
md = 1.4142
valorMDmax = 5.6569
valorDDmax = 16
custo = 28.2843
total = 177.7056

Para ic = 2 e id = 3:
ind(2,3) = 7
dist = 10
md = 2.2361
valorMDmax = 5.6569
valorDDmax = 16
custo = 23.8121
total = 197.0456

Para ic = 2 e id = 4:
ind(2,4) = 14
dist = 3
md = 3.1623
valorMDmax = 5.6569
valorDDmax = 16
custo = 24.2967
total = 221.8268

Para ic = 2 e id = 5:
ind(2,5) = 16
dist =  1
md = 4.1231
valorMDmax =  5.6569
valorDDmax = 16
custo = 31.6902
total = 260.9105
```

```
Para ic = 3 e id = 1:
ind(3,1) = 4
dist = 13
md = 2
valorMDmax = 5.6569
valorDDmax = 16
custo = 26.7696
total = 282.7594

Para ic = 3 e id = 2:
ind(3,2) = 6
dist = 11
md = 2.2361
valorMDmax = 5.6569
valorDDmax = 16
custo = 24.4045
total = 304.7989

Para ic = 3 e id = 3:
ind(3,3) = 13
dist = 4
md = 2.8284
valorMDmax = 5.6569
valorDDmax = 16
custo = 22.6274
total = 325.6492

Para ic = 3 e id = 4:
ind(3,4) = 20
dist = 3
md = 3.6056
valorMDmax = 5.6569
valorDDmax = 16
custo = 26.5130
total = 356.0479

Para ic = 3 e id = 5:
ind(3,5) = 22
```

```
dist = 5
md = 4.4721
valorMDmax = 5.6569
valorDDmax = 16
custo = 27.5585
total = 384.6519

Para ic = 4 e id = 1:
ind(4,1) = 10
dist = 7
md = 3
valorMDmax = 5.6569
valorDDmax = 16
custo = 22.7990
total = 402.6914

Para ic = 4 e id = 2:
ind(4,2) = 12
dist = 5
md = 3.1623
valorMDmax = 5.6569
valorDDmax = 16
custo = 23.6290
total = 427.1503

Para ic = 4 e id = 3:
ind(4,3) = 19
dist = 2
md = 3.6056
valorMDmax = 5.6569
valorDDmax = 16
custo = 27.2902
total = 458.1017

Para ic = 4 e id = 4:
ind(4,4) = 21
dist = 4
md = 4.2426
```

```
valorMDmax = 5.6569
valorDDmax = 16
custo = 28.2843
total = 487.3800

Para ic = 4 e id = 5:
ind(4,5) = 3
dist = 14
md = 5
valorMDmax = 5.6569
valorDDmax = 16
custo = 9.5980
total = 478.2917

Para ic = 5 e id = 1:
ind(5,1) = 11
dist = 6
md = 4
valorMDmax = 5.6569
valorDDmax = 16
custo = 24.9706
total = 518.6349

Para ic = 5 e id = 2:
ind(5,2) = 18
dist = 1
md = 4.1231
valorMDmax = 5.6569
valorDDmax = 16
custo = 31.6902
total = 557.0446

Para ic = 5 e id = 3:
ind(5,3) = 25
dist = 8
md = 4.4721
valorMDmax = 5.6569
valorDDmax = 16
```

```
custo = 22.6274
total = 570.6093

Para ic = 5 e id = 4:
ind(5,4) = 2
dist = 15
md = 5
valorMDmax = 5.6569
valorDDmax = 16
custo = 7.4264
total = 562.8347

Para ic = 5 e id = 5:
ind(5,5) = 9
dist = 8
md = 5.6569
valorMDmax = 5.6569
valorDDmax = 16
custo = 22.6274
total = 600.6631

adaptacao = adaptacao + 600.6631
```

Até este ponto, foram calculados o custo das distâncias e dissimilaridades do item $ind(1,1)$ em relação a todos os outros itens da matriz, onde a variável cumulativa $adaptacao$ alcançou o valor $600,6631$.

# Livros Grátis

( http://www.livrosgratis.com.br )

Milhares de Livros para Download:

Baixar livros de Literatura
Baixar livros de Literatura de Cordel
Baixar livros de Literatura Infantil
Baixar livros de Matemática
Baixar livros de Medicina
Baixar livros de Medicina Veterinária
Baixar livros de Meio Ambiente
Baixar livros de Meteorologia
Baixar Monografias e TCC
Baixar livros Multidisciplinar
Baixar livros de Música
Baixar livros de Psicologia
Baixar livros de Química
Baixar livros de Saúde Coletiva
Baixar livros de Serviço Social
Baixar livros de Sociologia
Baixar livros de Teologia
Baixar livros de Trabalho
Baixar livros de Turismo