UNIVERSIDADE PRESBITERIANA MACKENZIE
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

Um Novo Algoritmo Imunológico Artificial para Agrupamento de Dados

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia Elétrica da Universidade Presbiteriana Mackenzie, como requisito parcial para a obtenção do título de Mestre em Engenharia Elétrica.

Aluno: Ederson Borges
Orientador: Prof. Dr. Leandro Nunes de Castro

São Paulo
2010

B732u Borges, Ederson.
Um novo algoritmo imunológico artificial para agrupamento de dados / Ederson Borges. – 2010.
66 p. : il. ; 24 cm.

Dissertação (Mestrado em Engenharia Elétrica) – Escola de Engenharia, Universidade Presbiteriana Mackenzie, São Paulo, 2010.
Orientação: Leandro Nunes de Castro Silva.
Bibliografia: p. 63-66
Contém CD-ROM com o arquivo da versão final.
1.Agrupamento de dados. 2. Rede imunológica artificial. 3. Sistemas imunológicos artificiais. II. Título.

CDD 006.3

UNIVERSIDADE PRESBITERIANA MACKENZIE
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

Um Novo Algoritmo Imunológico Artificial para Agrupamento de Dados

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia Elétrica da Universidade Presbiteriana Mackenzie, como requisito parcial para a obtenção do título de Mestre em Engenharia Elétrica.

Aluno: Ederson Borges

Aprovada em 27/01/2010

BANCA EXAMINADORA

_____

Prof. Dr. Leandro Nunes de Castro
Universidade Presbiteriana Mackenzie

_____

Prof. Dr. Pedro Paulo Balbi de Oliveira
Universidade Presbiteriana Mackenzie

_____

Prof. Dr. Lalinka Teixeira de Campos Gomes
Centro de Pesquisas Avançadas Wernher Von Braun

AGRADECIMENTOS

Aos meus amigos, pais e colegas que participaram comigo desse trabalho, do começo ao fim, permitindo mais um momento da realização de um sonho.

Ao Pai Celestial por todos os dias permitidos de viver, conviver e trabalhar para chegar a este momento.

Ao meu orientador, Prof. Dr. Leandro Nunes de Castro, que não posso deixar de destacar, a presença incessante ao meu lado me auxiliando e incentivando nessa tarefa.

RESUMO

Agrupamento de dados é uma importante tarefa da mineração de dados e descoberta de conhecimentos em bases de dados. Existem diversos algoritmos capazes de realizar a tarefa de agrupamento de dados, sendo que os mais populares envolvem o cálculo de similaridade ou distância entre objetos da base de dados. Boa parte dos algoritmos pode agrupar os dados de forma simples e eficiente, mas possui inconvenientes como a forma de obter o número ótimo de partições e a possibilidade de ficar preso em ótimos locais. Para tentar diminuir estes inconvenientes essa dissertação propõe um novo Algoritmo Imunológico para Agrupamento de Dados baseado em Sistemas Imunológicos Artificiais. Esse algoritmo é caracterizado pela geração de múltiplas soluções simultâneas de boa qualidade no que tange o número de partições (grupos) para a base de dados e uma função de custo que avalia explicitamente a qualidade dessas partições, minimizando o inconveniente de ficar preso em ótimos locais. O algoritmo foi testado utilizando quatro bases de dados conhecidas na literatura e obteve resultados satisfatórios no que tange a diversidade das soluções encontradas, mas apresentou um custo computacional elevado em relação a outros algoritmos testados.

Palavras-chave: *Agrupamento de dados, Diversidade, K-médias, Rede Imunológica Artificial, Sistemas Imunológicos Artificiais.*

ABSTRACT

Clustering is an important data mining task from the field of Knowledge Discovery in Databases. There are several algorithms capable of performing clustering tasks, and the most popular ones involve the calculation of a similarity or distance measure among objects from the database. Many algorithms can perform clustering in a simple and efficient manner, but have drawbacks as a way to get the optimal number of partitions and the possibility of getting stuck in local optima solutions. To try and reduce these drawbacks this dissertation proposes a new clustering algorithm based on Artificial Immune Systems. This algorithm is characterized by the generation of multiple simultaneous high quality solutions in terms of the number of partitions (clusters) for the database and the use of a cost function that explicitly evaluates the quality of partitions, minimizing the inconvenience of getting stuck in local optima. The algorithm was tested using four databases known in the literature and obtained satisfactory results in terms of the diversity of solutions, but has a high computational cost compared to other algorithms tested.

Keywords: *Clustering, Diversity, K-means, Artificial Immune Network, Artificial Immune Systems.*

Lista de Ilustrações

Lista de Tabelas

SUMÁRIO

1 INTRODUÇÃO

A *computação natural* tem como um de seus objetivos o desenvolvimento de soluções para problemas reais através da observação de princípios do meio ambiente e do comportamento de espécies. Também é a área que apresenta o uso de materiais naturais para computação, como *computação quântica* e *computação molecular* (DE CASTRO, 2006).

A *computação natural* engloba técnicas que utilizam teorias antes relacionadas apenas à biologia, como, por exemplo, a teoria da evolução de Darwin. Pesquisadores utilizaram essa teoria para formular os primeiros *algoritmos evolutivos*, utilizados em tarefas computacionais como agrupamento de dados, reconhecimento de padrões e otimização de funções (DE CASTRO, 2006). Não só os *algoritmos evolutivos* podem ser citados, mas também outras técnicas podem ser mencionadas como parte integrante da *computação natural* para a solução de problemas, como as *redes neurais artificiais*, a *inteligência de enxame*, os *sistemas imunológicos artificiais* e outros.

Essa dissertação enfoca o desenvolvimento e aplicação de técnicas bio-inspiradas, particularmente *sistemas imunológicos artificiais*, ao problema de agrupamento de dados. Nesses problemas tem-se a necessidade de buscar técnicas capazes de encontrar, de forma automática, o número de partições que descreve corretamente os grupos de objetos encontrados na base.

Como citado anteriormente, os *algoritmos evolutivos* e outras diversas técnicas, já foram utilizados para se aproximar o melhor número de grupos em tarefas de agrupamento de dados (HRUSCHKA; CAMPELLO; DE CASTRO, 2006). A utilização de outras formas de se encontrar o número de grupos para esse tipo de problema é tratada nesse trabalho. E não apenas o número de grupos é foco do trabalho, mas também a geração de múltiplas soluções de boa qualidade em uma mesma simulação.

Através da utilização de técnicas encontradas na área de *Sistemas Imunológicos Artificiais* (DE CASTRO; TIMMIS, 2002) espera-se ser possível encontrar esse benefício da determinação de múltiplos agrupamentos de boa qualidade simultaneamente.

1.1 MOTIVAÇÃO

Tida como uma importante tarefa da descoberta de conhecimento em bases dados o agrupamento de dados pode ser utilizado para identificar relações de grande relevância prática

entre os dados (HRUSCHKA; DE CASTRO; CAMPELLO, 2006), como para a identificação de perfis de clientes, detecção de anomalias, segmentação de mercado, dentre outras.

Uma dificuldade para muitas técnicas de agrupamento está no fato delas necessitarem que seja fornecido, a priori, um número *k* de grupos possíveis na base apresentada. Diversas variações de técnicas, incluindo *algoritmos evolutivos*, já foram testadas para se aproximar de uma melhor solução para a descoberta automática do número *k* necessário para o agrupamento a ser gerado (HRUSCHKA; DE CASTRO; CAMPELLO, 2006).

Assim, é de grande importância o desenvolvimento de novas técnicas para se encontrar soluções capazes de melhorar a eficiência das técnicas de agrupamento, principalmente no que tange a determinação do número de grupos. Nesse caso os *sistemas imunológicos artificiais* podem ser capazes de gerar soluções não encontradas por outras técnicas. Além disso, as técnicas dos *sistemas imunológicos artificiais* possuem intrinsecamente características de manutenção de ótimos locais que faz com que os algoritmos gerem múltiplas soluções de boa qualidade com alta taxa de diversidade.

1.2 OBJETIVOS

Através de técnicas de agrupamento podem ser obtidos conhecimentos relevantes sobre os dados estudados (HRUSCHKA; DE CASTRO; CAMPELLO, 2006). Como exemplo, pode-se descrever a classificação de animais e plantas dadas suas características.

Entretanto, são necessárias técnicas capazes de determinar automaticamente o número ótimo de partições para técnicas de agrupamento (HRUSCHKA; DE CASTRO; CAMPELLO, 2006). Essas soluções podem ser obtidas através do uso de Sistemas Imunológicos Artificiais que são capazes de explorar de forma abrangente o espaço de busca podendo obter novas soluções para o agrupamento e manter múltiplas soluções de boa qualidade.

Esse trabalho tem o objetivo de propor um novo algoritmo imunológico para agrupamento de dados. Esse algoritmo tem como características principais:

1. Determinação automática do número *k* de grupos da base de dados;
2. Utilização de uma função de custo explícita no processo de busca;
3. Manutenção da diversidade através de um mecanismo explícito de identificação de soluções genotipicamente similares.

1.3 ORGANIZAÇÃO

Para a melhor apresentação dessa dissertação a organização se dará nos seguintes capítulos, descritas como:

- Capítulo 2: Mineração de dados – Nesse capítulo contempla-se a introdução da mineração de dados e a explicação detalhada do problema de se obter um agrupamento de dados, como determinar o número de partições e a forma de se obter uma medida de similaridade entre os objetos.
- Capítulo 3: Sistemas Imunológicos Artificiais – Nesse capítulo apresentam-se conceitos de Sistemas Imunológicos Artificiais (SIA) e técnicas de otimização baseadas em SIA.
- Capítulo 4: Algoritmo Proposto – Nesse capítulo é apresentado o algoritmo que será utilizado como base para o agrupamento de dados.
- Capítulo 5: Resultados e discussões – Nesse capítulo são apresentados os experimentos realizados utilizando o algoritmo proposto bem como a comparação deste com outros algoritmos existentes na literatura.

2 MINERAÇÃO DE DADOS

Os últimos anos testemunharam um forte desenvolvimento computacional que, dentre outros, cria novas formas de geração, coleta e armazenamento de grandes volumes de dados (CHEN; HAN; YU, 1996). Isto tem ocorrido em diversas áreas não só de pesquisa (por exemplo, imagens astronômicas, dados biológicos e física de moléculas) como em áreas cotidianas (por exemplo, supermercados, cartões de crédito e contas de telefone) (HAND; MANNILA; SMYTH, 2001). Dessa forma, faz-se necessária também a criação de novos métodos para descobrir informações relevantes (conhecimento) a partir destes dados armazenados. Com este intuito, pesquisadores passaram a se dedicar ao desenvolvimento de métodos para a extração de conhecimento a partir de bases de dados; estes métodos compõem hoje a chamada *mineração de dados* (FAYYAD; SHAPIRO; SMYTH, 1996). Muitos pesquisadores consideram que a extração de dados é um ponto fundamental para sistemas de bancos de dados e *aprendizado de máquina*; para o setor industrial é uma oportunidade de conseguir melhorar seus ganhos. Muitas aplicações emergentes que disponibilizam serviços, como em serviços on-line e na internet, também utilizam técnicas de mineração de dados para melhor entender o comportamento de seus usuários e, a partir desta compreensão (conhecimento), criar a oportunidade de aumentar os negócios e fornecer melhores serviços (CHEN; HAN; YU, 1996).

A mineração de dados pode ser entendida como uma área de análise de bases de dados com a finalidade de se encontrar relações e resumir os dados de forma compreensível e útil (HAND; MANNILA; SMYTH, 2001). Através da utilização das técnicas desenvolvidas em mineração de dados surge um novo processo para a *descoberta de conhecimento em bases de dados* (KDD – *Knowledge Discovery in Databases*) (FAYYAD; SHAPIRO; SMYTH, 1996). Pode-se definir KDD como sendo um processo não trivial de identificação de padrões válidos, úteis e compreensíveis em bases de dados (CHEN; HAN; YU, 1996; FAYYAD; SHAPIRO; SMYTH, 1996). Este processo inclui diversos estágios: selecionar os dados, pré-processar os dados, transformar os dados, aplicar técnicas de mineração de dados para extrair padrões e relacionamentos e, por fim, interpretar e avaliar as estruturas encontradas (HAND; MANNILA; SMYTH, 2001). O processo é considerado interativo e iterativo, pois envolve a decisão por um ou mais usuários e ocorre em diversas etapas (FAYYAD; SHAPIRO; SMYTH, 1996). A Figura 2.1 ilustra os principais estágios do processo de descoberta de conhecimento em bases de dados.

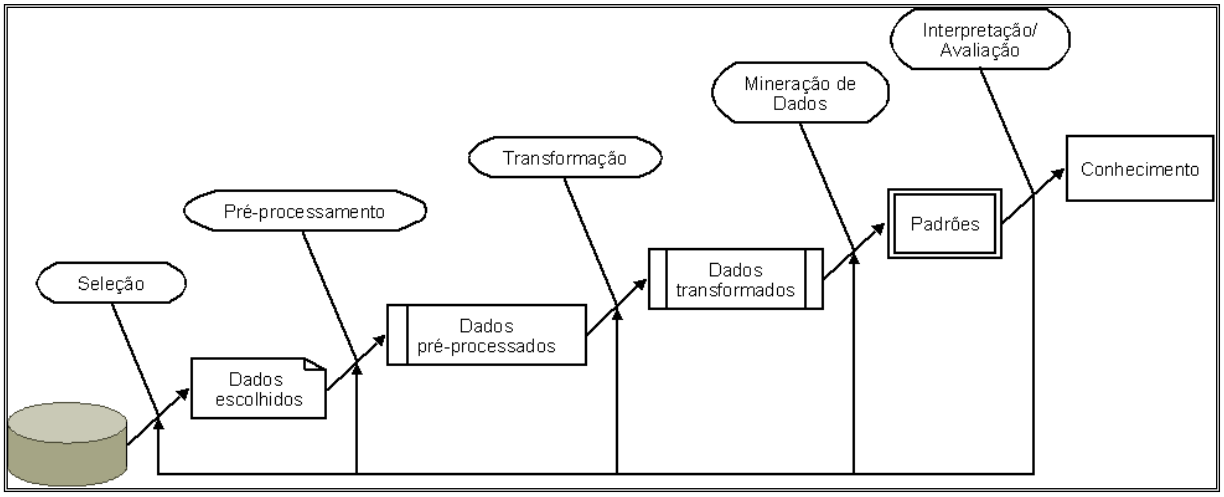


Figura 2.1: Estágios do processo de descoberta de conhecimento em banco de dados (FAYYAD; SHAPIRO; SMYTH, 1996).

Existem diversos métodos dentro da mineração de dados utilizados para a descoberta de conhecimento. A forma de se escolher o melhor método a ser utilizado envolve alguns estágios. Esses estágios compreendem a análise da estrutura e natureza de representação dos dados a ser utilizada; decidir como avaliar a melhor representação para os dados; escolher um algoritmo que otimize esta avaliação; e decidir a melhor forma de apresentar os dados para o algoritmo escolhido (HAND; MANNILA; SMYTH, 2001). Muitas vezes no processo os métodos de mineração de dados são aplicados repetidamente (FAYYAD; SHAPIRO; SMYTH, 1996).

A mineração de dados deve ser capaz de encontrar padrões em diferentes níveis de abstração para que o usuário que esteja aplicando técnicas de mineração possa obter padrões que devem ser úteis para seu tipo de aplicação (HAN; KAMBER, 2000). Para especificar o tipo de padrão a ser encontrado em tarefas de mineração de dados são utilizadas as funcionalidades de mineração de dados. As tarefas de mineração podem ser divididas em dois tipos (HAN; KAMBER, 2000):

- Preditivas: fazem inferência com os dados com o objetivo de prever algo, os dados contêm os valores de saída "desejados", correspondente para cada amostra. Como exemplo, tem-se:
    - Classificação: possui saídas discretas que representam rótulos de classe;
    - Regressão: possui saídas contínuas que representam valores de variáveis dependentes;

- Descritivas: consiste em analisar os dados do domínio e sugerir uma partição deste domínio, de acordo com similaridades observadas nos dados. Como exemplo, tem-se:
    - Agrupamento de dados: possui saídas que representam particionamentos para a base apresentada.

2.1 AGRUPAMENTO DE DADOS

Uma tarefa específica de mineração que será o foco desse trabalho é o *agrupamento de dados* (EVERITT, 1993; JAIN; MURTY; FLYNN, 1999). Diferentemente de outras tarefas da mineração de dados, como classificação e estimação, o agrupamento não utiliza as classes previamente conhecidas dos dados, chamadas de rótulos. Seu principal objetivo é encontrar padrões e relacionamentos em conjuntos de dados sem se conhecer rótulos a priori (HAN; KAMBER, 2000; WITTEN, FRANK, 2005). Por isso, essas técnicas também são conhecidas como métodos não-supervisionados, ou seja, métodos através dos quais não se conhecem previamente as classes existentes (CHEN; HAN; YU, 1996; JAIN; MURTY; FLYNN, 1999; NALDI, 2008).

Pode-se definir o agrupamento de dados como sendo o processo que visa agrupar ou segmentar fisicamente ou abstratamente objetos em grupos (ou clusters) de objetos similares (CHEN; HAN; YU, 1996; WITTEN; FRANK, 2005; HRUSCHKA et al., 2009). Dito de outra forma, um agrupamento (cluster ou grupo) de objetos é um conjunto de objetos cuja similaridade com objetos do mesmo grupo é maior que a similaridade com objetos de grupos distintos.

O agrupamento de dados é uma importante tarefa para a descoberta de conhecimento em bases de dados. Esta técnica é útil em diversas análises de padrões, tomada de decisão, aprendizado de máquina e para identificar relações entre os dados (JAIN; MURTY; FLYNN, 1999; HRUSCHKA; DE CASTRO; CAMPELLO, 2006). Em muitos problemas existem poucas informações iniciais disponíveis sobre os dados e muitas vezes o usuário deve fazer suposições conforme possível (JAIN; MURTY; FLYNN, 1999). Nesses casos, as técnicas de agrupamento de dados visam auxiliar o usuário apresentando relações entre os dados de forma a facilitar a interpretação e maximizar a possibilidade de uma melhor escolha.

Têm-se como exemplos de utilização de agrupamentos de dados a descoberta de conhecimento que pode ser utilizada como um início para exploração de relações entre

objetos de uma base. Esta técnica dá suporte ao desenvolvimento de modelos de segmentação de populações, como uma segmentação demográfica de consumidores. Também se tem como exemplo a utilização para classificação de animais e plantas dadas suas características. Utilizações na atualidade podem ser destacadas na área de segurança da computação, gerenciamento do relacionamento com os clientes, detecção de anomalias e muitas outras (MALOOF, 2006).

Na área de bioinformática cresce a necessidade de se encontrar novas técnicas de reconhecimento de padrões que possam descobrir e apresentar conhecimento significante para bases de dados. O agrupamento de dados pode ser capaz de identificar grupos de genes similares. Genes que apresentam padrões similares em um grande número de experimentos podem ser alocados a um mesmo grupo. Genes alocados a um mesmo grupo são co-regulados e envolvidos em funções relacionadas. Outra possibilidade envolve o agrupamento de amostras por relacionamento de parentesco nos padrões de expressão que representam fenótipos complexos. Como exemplo, essa identificação de amostras com fenótipos similares pode representar diferentes tipos de câncer. (HRUSCHKA; CAMPELLO; DE CASTRO, 2006).

A Figura 2.2 ilustra de forma simplificada os principais estágios envolvidos na tarefa de agrupamento.

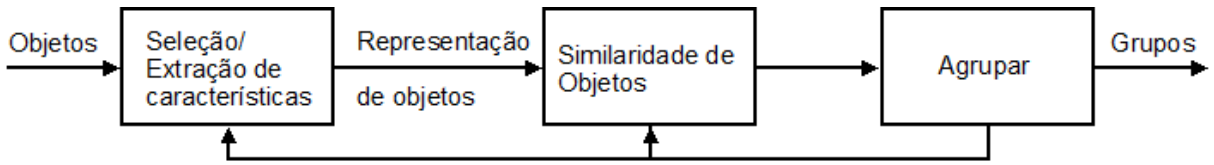


Figura 2.2: Estágios do agrupamento (JAIN et al., 1999).

A r*epresentação de objetos* se refere às informações disponíveis para o algoritmo de agrupamento; s*eleção/extração de características* é o processo que seleciona e transforma atributos originais que melhor representem o conjunto de dados; e a *similaridade entre os objetos* avalia a similaridade (ou dissimilaridade) entre os objetos da base através de funções de similaridade (ou distância) (JAIN; MURTY; FLYNN, 1999).

2.2 CONJUNTOS DE DADOS

Para iniciar o processo de agrupamento de dados deve-se analisar o conjunto de dados a ser agrupado. Um conjunto de dados é composto de objetos, que podem ser representações

físicas, como plantas ou animais, ou abstratas, como interações entre pessoas (NALDI, 2008). Cada objeto presente neste conjunto de dados possui características, também conhecidas como atributos, que podem ser abstratas ou físicas.

Informações sobre os atributos normalmente se fazem necessárias para então efetuar transformações através de normalizações, conversões de tipo e redução do número de atributos, dentre outras. Os atributos podem ser subdivididos em dois grupos: *quantitativos* e *qualitativos*. Os atributos *quantitativos* representam números (finitos ou infinitos) que podem ser enumeráveis ou não e podem ser apresentados em valores *contínuos*, *discretos* e em *intervalos*. Os atributos *qualitativos* representam características que podem ou não formar uma ordem e podem ser *nominais* ou *não-ordenados* e *ordinais* (JAIN; MURTY; FLYNN, 1999; NALDI, 2008).

Existem diversos conjuntos de dados para os quais podem ser aplicadas técnicas de agrupamento. Um exemplo ilustrativo de conjunto de dados tem a seguinte configuração de objetos, apresentada na Figura 2.3.

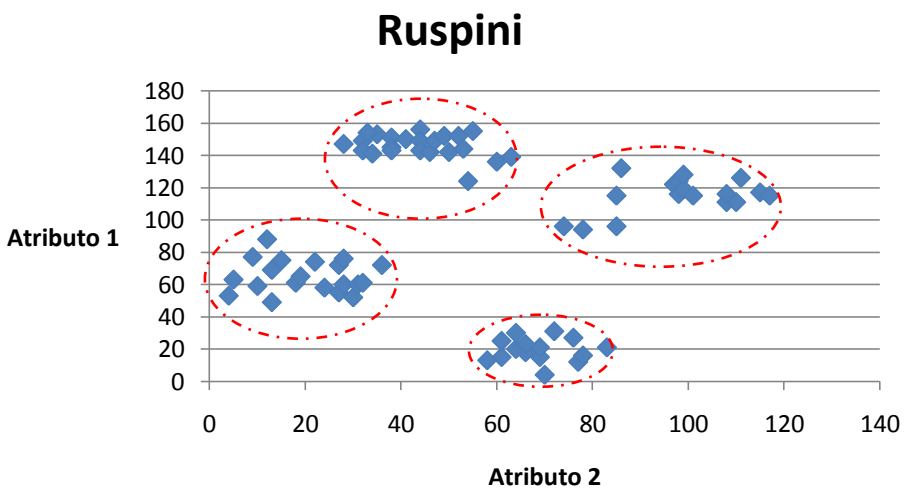


Figura 2.3: Objetos da base de dados Ruspini.

A base *Ruspini* (KAUFMAN; ROUSSEEUW, 1990) é uma base sintética e utilizada em diversos trabalhos como um *benchmark* para avaliação de métodos de agrupamento de dados (HRUSCHKA; CAMPELLO; DE CASTRO, 2004). Pode-se observar que este conjunto possui grupos bem definidos de objetos em um espaço bi-dimensional. Nesse caso, um algoritmo de agrupamento de dados aplicado à base Ruspini tem uma maior tendência de apresentar a separação dos objetos destes grupos em quatro agrupamentos distintos, conforme visualizado na Figura 2.3.

Considere agora uma base contendo animais como seus objetos (HAYKIN, 1999). Esses animais possuem diversas características apresentadas na Tabela 2.1 sendo que a base possui apenas atributos binários indicando a presença ou ausência de uma característica.

Essa base poderia ser agrupada de várias formas distintas, porém coerentes. Através das características físicas (É) verifica-se que o agrupamento terá como grupos divididos em PEQUENO, MÉDIO e GRANDE, ou então, através de características de ações dos animais (GOSTA DE) terá como grupos os animais que gostam de CAÇAR, CORRER, VOAR e NADAR sendo que com estas características existem animais que possam pertencer a diferentes grupos possuindo mais de uma característica presente. Pode ser observada então a necessidade de se encontrar múltiplas soluções que apresentem estes diferentes padrões encontrados em bases de dados.

		Pombo	Galinha	Pato	Ganso	Coruja	Gavião	Águia	Raposa	Cão	Lobo	Gato	Tigre	Leão	Cavalo	Zebra	Vaca
É	Pequeno	1	1	1	1	1	1	0	0	0	0	1	0	0	0	0	0
	Médio	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0
	Grande	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1
Tem	Duas patas	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
	Quatro patas	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1
	Pêlos	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1
	Casco	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1
	Crina	0	0	0	0	0	0	0	0	0	1	0	0	1	1	1	0
	Penas	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
Gosta de	Caçar	0	0	0	0	1	1	1	1	0	1	1	1	1	0	0	0
	Correr	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	0
	Voar	1	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0
	Nadar	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0

Tabela 2.1: Exemplo de base de dados de animais.

2.3 TÉCNICAS DE AGRUPAMENTO

É possível dividir as técnicas de agrupamento de dados em dois grandes grupos: agrupamento *hierárquico* e *particional*. As técnicas baseadas no modelo *hierárquico* produzem uma série de partições aninhadas com base em um critério de fusão ou quebra dos grupos baseados em similaridade. As técnicas baseadas no modelo *particional* identificam a partição que otimiza (normalmente localmente) algum critério do agrupamento (ESTER et al., 1996; JAIN; MURTY; FLYNN, 1999; ZHAO, 2005).

As técnicas *hierárquicas* decompõem os objetos da base utilizada, sendo que a representação da decomposição é normalmente feita por um *dendrograma* (ESTER et al., 1996; JAIN; MURTY; FLYNN, 1999; ZHAO, 2005). O dendrograma pode ser quebrado em diferentes níveis que gera diferentes agrupamentos dos dados (JAIN; MURTY; FLYNN, 1999). A Figura 2.4 ilustra um exemplo de dendrograma que possui os itens A, B, C, D, E, F e G.

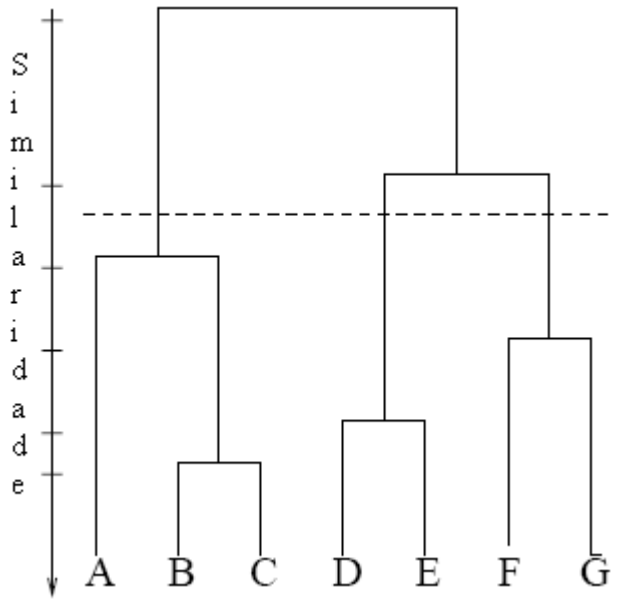


Figura 2.4: Dendrograma: representação em alguns algoritmos de agrupamento hierárquicos (JAIN; MURTY; FLYNN, 1999).

Pelo dendrograma apresentado na Figura 2.4 verifica-se a possibilidade de escolher diferentes níveis para o agrupamento. Pelo nível de similaridade escolhido nesse exemplo (linha pontilhada) tem-se três grupos, esses grupos são [A, B, C], [D, E] e [F, G]. Esse método de agrupamento assume previamente uma estrutura para os dados. Caso os dados não apresentem estrutura hierárquica esta técnica pode não ser adequada.

No modelo *particional* os algoritmos constroem uma partição a partir dos objetos de uma base de dados e formam *k* grupos com estes objetos (ESTER et al., 1996; JAIN; MURTY; FLYNN, 1999; ZHAO, 2005). Muitos destes algoritmos requerem a entrada pelo usuário do valor *k*, de forma que o usuário deve possuir um conhecimento prévio do domínio de sua base de dados, fato que muitas vezes não ocorre, ou seja, o usuário pode não dispor deste conhecimento de um número *k* adequado (ESTER et al., 1996). Os algoritmos de agrupamento particionais também podem ser utilizados para obter soluções hierárquicas por uma sequência de repetidas bi-seções (ZHAO, 2005).

Uma técnica clássica de agrupamento particional que está descrita neste trabalho, é conhecida como *k-médias* (JAIN; MURTY; FLYNN, 1999; ZHAO, 2005) essa técnica é uma das mais utilizadas na literatura. O *k-médias* se baseia no cálculo da distância entre os objetos da base e alguns objetos fictícios, comumente chamados de *centróides* ou *protótipos*, criados aleatoriamente. Os objetos então são agrupados pela proximidade com estes centróides. É feita a média das distâncias dos objetos agrupados junto a um mesmo centróide, gerando os novos centróides que são utilizados em uma próxima iteração (ESTER et al., 1996; JAIN; MURTY; FLYNN, 1999; ZHAO, 2005). O algoritmo para assim que estes novos centróides gerados atingirem um limiar de diferença pré-definido em relação aos centróides gerados na iteração anterior (WITTEN; FRANK, 2005), ou então no momento em que o algoritmo atingir um número pré-determinado de iterações. Na Figura 2.5 é apresentado o fluxograma do algoritmo *k-médias*.

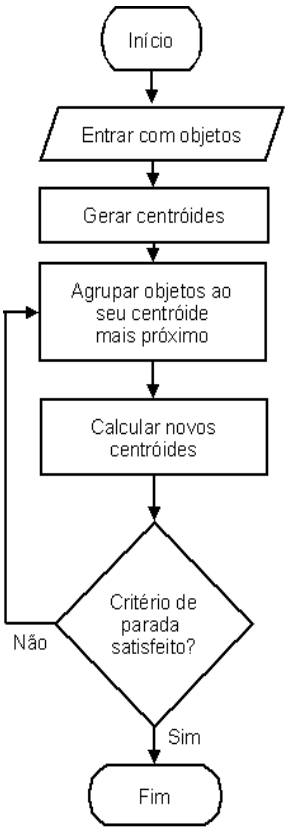


Figura 2.5: Fluxograma do algoritmo k-médias.

Esta técnica, *k-médias*, é considerada simples e eficiente, mas possui alguns inconvenientes. Um deles é que o número *k* de grupos a ser gerado deve ser especificado antes de se iniciar as iterações do algoritmo; sendo assim, um conhecimento prévio dos objetos e sua organização poderão ser necessários. Caso não seja possível este conhecimento prévio, o

que muitas vezes pode ocorrer dado que o objetivo é gerar os grupos pelo algoritmo automaticamente, o algoritmo *k-médias* acaba sendo executado diversas vezes com diferentes valores de *k* até que se encontre um valor adequado aos propósitos desejados a partir do agrupamento. Outro inconveniente é a geração aleatória dos primeiros centróides, que pode fazer com que o algoritmo encontre um ótimo local, ou seja, o algoritmo para sua execução mesmo sem encontrar a melhor solução de agrupamento. Novamente a proposta seria executar o algoritmo um número de vezes para então selecionar o melhor agrupamento possível dos objetos da base (WITTEN; FRANK, 2005). Muitas variações do algoritmo já foram apresentadas na literatura tentando minimizar estes problemas (JAIN; MURTY; FLYNN, 1999).

Através de técnicas encontradas na Computação Natural podem-se minimizar estes inconvenientes encontrados nos algoritmos clássicos de agrupamento (DE CASTRO, 2007; HRUSCHKA; DE CASTRO; CAMPELLO, 2006).

Nesse trabalho é apresentada uma área recente de estudos da Computação Natural (DE CASTRO, 2007) que se inspira no Sistema Imunológico Biológico para solucionar problemas reais. Essa bioinspiração, conhecida como Sistemas Imunológicos Artificiais (DE CASTRO; TIMMIS, 2002, 2003), é utilizada aqui para a construção de um novo algoritmo para agrupamento de dados objetivando encontrar múltiplas soluções de boa qualidade de agrupamento em uma mesma iteração. Além da utilização dos conceitos dos Sistemas Imunológicos Artificiais o algoritmo também utiliza conceitos dos Algoritmos Evolutivos que também serão brevemente revisados neste trabalho.

3 SISTEMAS IMUNOLÓGICOS ARTIFICIAIS

Muitas inspirações para o desenvolvimento de modelos computacionais e métodos de resolução de problemas podem ser encontradas na biologia. Os Sistemas Imunológicos Artificiais (SIA) utilizam inspiração biológica nos sistemas imunológicos dos vertebrados para o desenvolvimento de ferramentas computacionais de solução de problemas (DE CASTRO; TIMMIS, 2002; HUNT; COOKE, 1996; DASGUPTA; JI; GONZALEZ, 2003).

Os SIA podem ser definidos como sistemas computacionais inspirados em imunologia teórica, observações de funções imunológicas, seus princípios e mecanismos com o objetivo de resolver problemas (DE CASTRO; TIMMIS, 2002). Eles têm ganhado popularidade e surgem como um novo ramo da Inteligência Artificial (DASGUPTA; JI; GONZALEZ, 2003) e Computação Flexível (DE CASTRO; TIMMIS, 2003). Comparados a outros paradigmas da Computação Natural, como as redes neurais artificiais e os algoritmos evolutivos os SIA ainda estão no início de seu desenvolvimento (DE CASTRO; TIMMIS, 2002).

Antes de serem apresentadas as técnicas utilizadas pelos SIA serão brevemente discutidos alguns conceitos biológicos utilizados para o desenvolvimento dessas técnicas.

3.1 PRINCÍPIOS DE SISTEMAS IMUNOLÓGICOS BIOLÓGICOS

Todo ser vivo apresenta uma resistência frente a agentes causadores de doenças, conhecidos como patógenos, que incluem vírus, bactérias e outros parasitas. O tipo de resistência para cada patógeno varia de uma espécie para outra de ser vivo. Os seres humanos, por exemplo, possuem um sistema imunológico de alta complexidade que age juntamente com outros sistemas para proteger nosso corpo contra patógenos e manter o equilíbrio do organismo (DE CASTRO, 2006).

Os sistemas imunológicos consistem de múltiplas células, moléculas e órgãos que interagem de diversas maneiras para detectar e eliminar patógenos (HOFMEYR; FORREST, 1999). Diversas teorias encontradas na literatura tentam explicar a forma com que este complexo sistema trabalha. Através dessas teorias algumas características dos sistemas imunológicos podem ser destacadas como importantes fontes de inspiração para o desenvolvimento dos SIA (HUNT; COOKE, 1996):

- *Diversificação*: As células e moléculas que compõem o sistema imunológico possuem ampla diversidade, sendo capazes de reconhecer uma vasta quantidade de patógenos;
- *Distributividade*: O sistema imunológico não possui um sistema de controle central, ele está distribuído por diversas células e moléculas que se espalham por todo o organismo;
- *Memória*: Os sistemas imunológicos possuem uma memória auto-organizada e dinâmica que permite armazenar informação sobre os patógenos e combatê-los mais eficientemente em casos de futuras invasões por patógenos já conhecidos.

Com essas características as teorias sobre os sistemas imunológicos biológicos estão sendo utilizadas para o desenvolvimento de algoritmos computacionais para a solução de problemas complexos. Algumas idéias centrais têm sido comumente exploradas, como, por exemplo, a *teoria da seleção clonal e maturação de afinidade* e as *redes imunológicas*, essas idéias fazem parte do *sistema imunológico adaptativo*. Ultimamente novas propostas baseadas nos conceitos de *imunidade inata* também têm sido exploradas (DE CASTRO, 2006).

3.1.1 Seleção Clonal e Maturação da Afinidade

De uma forma simplificada, a *teoria da seleção clonal* formalizada por Burnet (1959) explica que assim que o sistema imunológico é exposto a um patógeno, algumas células imunológicas do corpo são estimuladas. Essas células possuem receptores capazes de reagir com partes desse patógeno, chamadas de *antígenos*. Com essa estimulação as células imunológicas começarão a se proliferar (expansão clonal). Algumas células irão liberar *anticorpos* (moléculas receptoras de antígenos) em altas taxas, que, por sua vez, podem se combinar aos antígenos. Esse complexo antígeno-anticorpo será posteriormente eliminado (destruído) por outras células imunes (FORSDYKE, 1995). Além das células produtoras de anticorpos algumas células tornar-se-ão *células de memória*, ou seja, células que possuem períodos prolongados de vida que faz com que nos tornemos imunes aos patógenos que já desafiaram nosso sistema imune.

No momento da expansão clonal alguns *erros* (*mutação*) podem ocorrer. Uma característica importante que deve ser destacada com a ocorrência da mutação é que ela ocorrerá a uma taxa inversamente proporcional ao *grau de reconhecimento* (*afinidade*) do

anticorpo com o antígeno. Dessa forma, caso a afinidade entre o antígeno e um anticorpo seja alta, a mutação ocorrerá a uma menor taxa, enquanto uma afinidade menor entre antígeno e anticorpo deverá proporcionar uma maior taxa de mutação. Isto sugere que a mutação é regulada no sistema imunológico pela afinidade entre antígeno e anticorpo (DE CASTRO; VON ZUBEN, 2001). Este processo de mutação controlada seguida pela seleção e manutenção em memória das células com maior afinidade ao antígeno é conhecido como *maturação da afinidade*.

Considerando que o sistema imunológico é um sistema distribuído por todo o corpo, apenas as células que se encontram próximas ao antígeno serão estimuladas, outras células podem começar a interação com o antígeno, mas não serão todas as células que sofrerão o estímulo (DE CASTRO, 2006).

A teoria da seleção clonal e a maturação da afinidade formam um importante núcleo da *resposta imunológica adaptativa* e vem sendo muito estudada na literatura dos SIA para o desenvolvimento de sistemas adaptativos para solução de problemas (DE CASTRO, 2006).

3.1.2 Redes Imunológicas

A teoria das redes imunológicas foi proposta por Jerne em meados dos anos 70. Ao contrário da seleção clonal, que propõe que um sistema imunológico de células e moléculas discretas é estimulado por antígenos externos, a teoria das redes imunológicas apresenta um sistema imunológico dinâmico (DE CASTRO, 2006).

Jerne sugere que o sistema imunológico é uma rede regulada composto de células e moléculas que podem se reconhecer mesmo na ausência de antígenos. A dinâmica do sistema imunológico está na regulação (geração de novas células estimuladas e morte de células não estimuladas) e no reconhecimento entre as células do sistema imunológico e no reconhecimento de outras células, moléculas e antígenos (DE CASTRO; VON ZUBEN, 2000; DASGUPTA; JI; GONZALEZ, 2003).

No reconhecimento entre as células pode ocorrer uma resposta *positiva* ou *negativa*. No caso de uma resposta *positiva* o reconhecimento fará com que as células iniciem um processo de proliferação, ativação e liberação de anticorpos. Para uma resposta *negativa* o processo que se dará é um processo de tolerância e supressão (eliminação de células) (DE CASTRO; VON ZUBEN, 2000).

No modelo de rede imunológica proposto por Varela e Coutinho (1991) é descrita a atividade do sistema imunológico. São introduzidos três conceitos na rede imunológica:

estrutura, *dinâmica* e *metadinâmica* (DE CASTRO; VON ZUBEN, 2001; VARELA; COUTINHO, 1991; DE CASTRO, 2006).

A *estrutura* corresponde aos padrões encontrados nas células que descrevem a forma que estas se conectam entre si. A *dinâmica* apresenta como as novas células e moléculas variam ao longo do tempo. Essa característica apresenta como o sistema imunológico pode ser adaptável ao ambiente. A *metadinâmica* corresponde à característica do sistema imunológico constantemente gerar novas células e também de ocorrer a eliminação (morte) das células que não foram estimuladas por um período de tempo (DE CASTRO; VON ZUBEN, 2000, 2001).

3.2 ALGORITMOS DOS SIA

Diversos exemplos de algoritmos podem ser citados que utilizam inspiração nos sistemas imunológicos naturais.

De Castro e Timmis (2003) propuseram uma separação em dois tipos de algoritmos dos SIA: modelos baseados em *população* e os modelos baseados em *rede*.

Nos modelos baseados em população temos os algoritmos *clonais* e os algoritmos de *seleção negativa*. Nos modelos baseados em rede temos as *redes contínuas* e as *redes discretas*.

Os modelos clonais são aqueles que se inspiram nas funções do sistema imunológico de geração de novos clones e maturação de afinidade. Um algoritmo que utiliza esta inspiração é o CLONALG (DE CASTRO; VON ZUBEN, 2000), desenvolvido inicialmente para tarefas de aprendizado de máquina e reconhecimento de padrões, mas posteriormente adaptado a solução de problemas de otimização, com ênfase em otimização multimodal e combinatória (DE CASTRO; VON ZUBEN, 2002).

Os principais aspectos imunológicos utilizados para o desenvolvimento do algoritmo CLONALG foram: (1) seleção e clonagem das células estimuladas, proporcionalmente a sua afinidade a um antígeno; (2) morte das células pouco ou não estimuladas; (3) maturação de afinidade e seleção de células proporcionalmente à afinidade; e (4) geração e manutenção de diversidade (DE CASTRO; TIMMIS, 2003).

Um modelo baseado em *rede*, mais especificamente *redes discretas*, é o algoritmo *aiNet* (DE CASTRO; VON ZUBEN, 2001). Este algoritmo foi aplicado com sucesso em diversos problemas de compressão de dados e agrupamento.

O algoritmo *aiNet* tem sua inicialização com a geração de um pequeno número de células geradas aleatoriamente. Após a geração das células iniciais são introduzidos os padrões antigênicos e as afinidades destes padrões são calculadas em relação a cada célula (anticorpo) da rede. Anticorpos com alta afinidade são selecionados e reproduzidos (proliferação) de acordo com sua afinidade; anticorpos com maior afinidade tem um número maior de clones. E estes clones sofrem uma taxa de mutação inversamente proporcional a sua afinidade, de forma que clones com maior afinidade tem uma taxa de mutação menor. São selecionados os clones que apresentarem maior afinidade para serem mantidos na rede (memória clonal). A afinidade entre os anticorpos restantes é calculada, aqueles anticorpos em que a afinidade for menor que um determinado valor são retirados da rede (supressão). A metadinâmica do algoritmo corresponde à introdução e morte de anticorpos da rede (DE CASTRO; TIMMIS, 2003).

Com esta metadinâmica pode-se destacar a presença de características de diversidade e maior exploração do espaço de busca no algoritmo *aiNet* (DE CASTRO; VON ZUBEN, 2001). Além disso, outras características estão presentes no algoritmo, como manutenção de soluções com ótimos locais e a população com tamanho dinâmico.

Para o algoritmo *aiNet* foi desenvolvida uma versão para otimização, chamada de *opt-aiNet* (Artificial Immune Network for Optimization), detalhada a seguir (DE CASTRO; TIMMIS, 2002).

3.2.1 opt-aiNet: Um algoritmo de rede imunológica para otimização

O algoritmo *opt-aiNet*, derivado do algoritmo *aiNet*, tem a capacidade de desempenhar buscas unimodais e multimodais (DE CASTRO; TIMMIS, 2002b; TIMMIS; EDMONDS, 2004).

Na *opt-aiNet* cada célula da rede representa uma solução do problema a ser tratado. Dessa forma, como o algoritmo é utilizado para encontrar soluções de otimização em ambientes contínuos, cada célula é um vetor de valor real no espaço Euclidiano. O algoritmo *opt-aiNet* utiliza uma função de avaliação que se deseja otimizar e que dá o valor do *fitness* desta célula. O *fitness* quantifica a qualidade da solução representada por uma célula: valores altos de *fitness* indicam uma soluções de boa qualidade e valores baixos indicam soluções de menor qualidade. Existe ainda o processo de geração de clones e mutação. Na geração existe apenas a proliferação das células em um determinado valor pré-definido pelo usuário. Na mutação tem-se a variação dos clones gerados, que ocorrerá baseada em um critério

específico. Para a *opt-aiNet* esse critério é uma função que irá alterar o clone com um valor inversamente proporcional ao *fitness*: com um *fitness* maior a alteração sofrida será menor, com *fitness* menor a variação será maior. Além disso, existem os processos de afinidade, supressão e entrada de novas células na rede. Para a afinidade será feita a avaliação baseada na distância Euclidiana entre as células. Com uma afinidade acima de um limiar pré-especificado a célula com menor valor de *fitness* sofrerá supressão, avaliação de células com valores abaixo desse limiar serão mantidas como *células de memória* que representam as melhores soluções candidatas para a função. Após este processo de supressão novas células geradas aleatoriamente serão incluídas na rede (DE CASTRO; TIMMIS, 2002; TIMMIS; EDMONDS, 2004).

O algoritmo da *opt-aiNet* está resumido pelo Algoritmo 3.1 (DE CASTRO; TIMMIS, 2002).

Algoritmo 3.1: Algoritmo opt-aiNet.

1. Inicializar população de células aleatoriamente (o número de células iniciais não é relevante).
2. **Enquanto** o critério de parada não é encontrado, **faça**
   2.1. Determinar o *fitness* de cada célula da rede e normalizar o vetor de *fitness*.
   2.2. Gerar um número *Nc* de clones para cada célula da rede.
   2.3. Aplicar mutação em cada clone proporcionalmente ao *fitness* de sua célula pai, mas manter a célula pai.
   2.4. Determinar o *fitness* de todos os indivíduos da população.
   2.5. Para cada clone, selecionar a célula com mais alto *fitness* e calcular o *fitness* médio da população selecionada.
   2.6. Se o erro médio da população não é significantemente diferente da iteração anterior, então continua. Senão, volta ao Passo 2.1
   2.7. Determinar a afinidade de todas as células na rede. Suprimir células com maior aptidão entre si cujas afinidades são inferiores ao limiar de supressão σs e determinar o número de células da rede (células de memória) após a supressão.
   2.8. Introduzir uma porcentagem *d*% de células geradas aleatoriamente e retornar ao Passo 2.
3. **Fim Enquanto.**

Tomando como base o algoritmo *opt-aiNet* pode-se utilizar sua metodologia para o desenvolvimento de um novo algoritmo para resolução de problemas de agrupamento de dados. Essa dissertação apresenta uma adaptação do algoritmo *opt-aiNet* para utilização em tarefas de mineração de dados, especificamente a tarefa de agrupamento de dados.

4 OCOPT-AINET: ALGORITMO IMUNOLÓGICO PARA AGRUPAMENTO DE DADOS

O algoritmo imunológico proposto nesse trabalho, intitulado *ocopt-aiNet* (Optimal *Clustering opt-aiNet*), toma como base dois algoritmos existentes na literatura: 1) o algoritmo imunológico *opt-aiNet* (DE CASTRO; TIMMIS, 2002b); e 2) o algoritmo evolutivo para agrupamento (*Evolutionary Algorithm for Clustering - EAC*), proposto para realizar agrupamento ótimo de dados (HRUSCHKA; CAMPELLO; DE CASTRO, 2006). O algoritmo *opt-aiNet* foi utilizado para definir toda a dinâmica de adaptação do algoritmo *ocopt-aiNet*, enquanto o algoritmo *EAC* foi usado para definir a representação dos indivíduos da população e os operadores de variação genética do algoritmo *ocopt-aiNet* (HRUSCHKA; CAMPELLO; DE CASTRO, 2006).

4.1 EAC: ALGORITMO EVOLUTIVO PARA AGRUPAMENTO DE DADOS

Os algoritmos evolutivos possuem sua inspiração na Teoria da Evolução das espécies de Charles Darwin. Esses algoritmos são capazes de encontrar boas soluções em tempo computacional razoável e, por este motivo, eles são utilizados como opção para a solução de problemas complexos.

O princípio da evolução é o conceito primário que liga todos os seres vivos em uma corrente singular de eventos. Cada espécie presente nesta corrente é produto de um conjunto de eventos específicos sobre uma pressão seletiva aplicada pelo ambiente nos organismos. Ao longo de gerações, seleção natural e variações formam-se os indivíduos melhor adaptados ao ambiente (FOGEL, 2000).

A *Seleção Natural* foi o termo utilizado para definir como novas características que surgem pelas variações são mantidas entre gerações. Essas pequenas diferenças (*variações*) nos organismos ao longo de gerações devem resultar na aparição de novas e mais adaptadas espécies para seu ambiente (DE CASTRO, 2006).

O EAC é iniciado por uma população de indivíduos que representam soluções candidatas ao problema que se quer tratar. Para esta geração inicial utilizam-se indivíduos gerados aleatoriamente.

Esta geração inicial então é utilizada para gerar *descendentes* através de variações aleatórias pré-selecionadas. As soluções candidatas resultantes serão avaliadas pela sua eficácia em solucionar o problema tratado – valor do *fitness* de cada indivíduo da população. Assim como ocorre a pressão seletiva pelo ambiente aos indivíduos, no algoritmo evolutivo também ocorre um processo em que apenas os indivíduos mais adaptados (melhor *fitness*) são mantidos para a próxima geração e este processo é repetido inúmeras vezes (FOGEL, 2000). Para o *EAC* deve-se parametrizar o número máximo de iterações (*num_it*) como critério de parada. Este processo geralmente está presente nos algoritmos evolutivos; o EAC possui ainda um passo para executar uma busca local; utilizando, para isso, o algoritmo *k-médias* apresentado anteriormente. Essa busca local tem o objetivo de aproximar as soluções candidatas de um agrupamento de boa qualidade antes que seja feita a avaliação de cada indivíduo da população. Dessa forma ocorre uma melhora na busca pelo melhor agrupamento.

O resumo do algoritmo evolutivo para agrupamento (EAC) está apresentado no Algoritmo 4.1.

Algoritmo 4.1: Algoritmo evolutivo para agrupamento de dados.

1. Iniciar a população com indivíduos aleatórios.
2. **Enquanto** critério de parada não satisfeito, *i* < *num_it*, **faça**
   2.1. Aplicar busca local (k-médias) para cada indivíduo.
   2.2. Avaliar cada indivíduo (solução candidata) segundo uma função de avaliação.
   2.3. Aplicar normalização linear (transformação para intervalo [0,1]).
   2.4. Selecionar indivíduos através de seleção elitista e seleção por roleta.
   2.5. Aplicar os operadores evolutivos pré-selecionados (variação).
   2.6. Substituir os indivíduos antigos pelos formados no Passo 2.5 para a próxima geração.
3. **Fim Enquanto.**

Vale ressaltar alguns passos presentes no EAC: o primeiro passo é a avaliação de cada indivíduo (Passo 2.2) que é feita seguindo uma função de avaliação, que também está presente no algoritmo proposto nessa dissertação e que será apresentado na sequência (dessa forma tem-se um mesmo critério para avaliar as soluções candidatas em ambos os algoritmos); e o segundo passo é a seleção dos indivíduos (Passo 2.4) que, além de utilizar um método de seleção proporcional, a *seleção por roleta*, também apresenta um método elitista, *seleção elitista.* Por esse último método de seleção o indivíduo com melhor *fitness* será mantido

diretamente para a próxima geração da população sem sofrer mutação. No método de *seleção por roleta* é feita a seleção através de uma escolha aleatória onde indivíduos com maior *fitness* possuem maior probabilidade de serem selecionados para serem mantidos na próxima geração e, da mesma forma, indivíduos com menor valor de *fitness* possuem uma menor probabilidade de se manterem na próxima geração.

Os operadores de evolutivos (ou operadores de mutação como são chamados na *ocopt-aiNet*) encontrados no algoritmo *EAC* estão descritos a seguir.

4.1.1 Operadores Evolutivos

O primeiro operador (Operador de Exclusão) é aplicado apenas nas células que possuírem mais de dois grupos em sua codificação, promovendo a eliminação, através de uma escolha aleatória, de um dos grupos da solução. Como exemplo pode-se utilizar a célula com a codificação ilustrada na Figura 4.1.

Célula 1 – [113413422134433]

Figura 4.1: Célula representando possível solução.

Supondo que o grupo 2 seja escolhido aleatoriamente para ser excluído desta célula, os objetos nas posições 8 e 9 terão que ser movidos para outro grupo. O(s) novo(s) grupo(s) ao(s) qual(is) estes objetos pertencerão será(ão) escolhido(s) através da proximidade de cada um destes objetos ao *centróide* (objeto fictício que representa a média do grupo) de cada grupo remanescente representado por essa célula. A Figura 4.2 ilustra um caso hipotético de objetos, com dois atributos cada, codificados pela Célula 1 ilustrada na Figura 4.1 e indica os novos grupos dos objetos 8 e 9.

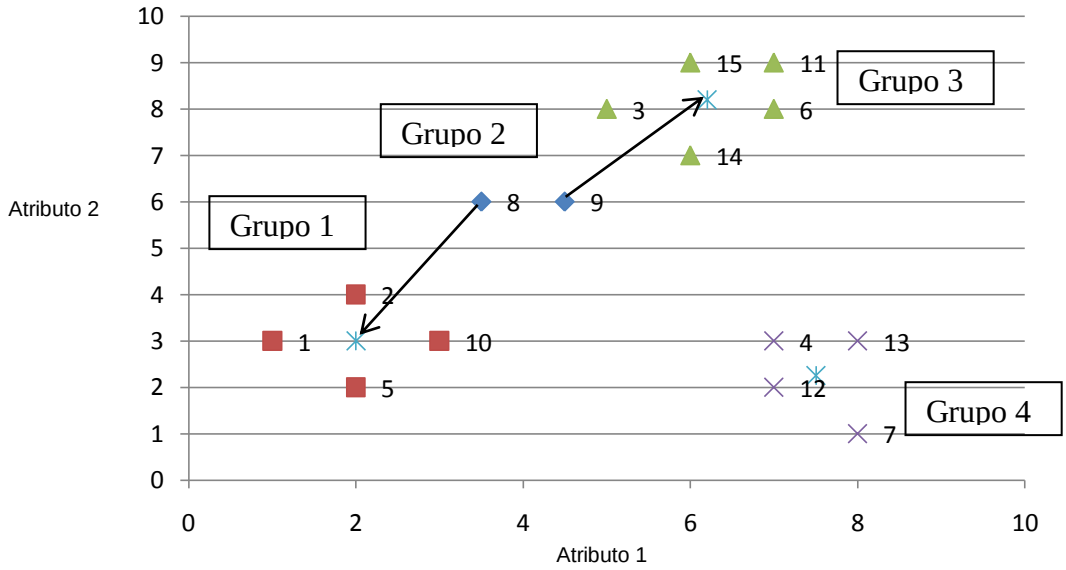


Figura 4.2: Agrupamento codificado pela Célula 1, no qual os objetos (8 e 9) pertencentes originalmente ao grupo 2 são realocados para os grupos remanescentes mais próximos.

No Algoritmo 4.2 é apresentado o pseudocódigo do Operador de Exclusão implementado pelo EAC.

Algoritmo 4.2: Operador de Exclusão.

1. Verificar se célula possui mais de dois grupos.
2. Escolher aleatoriamente um grupo presente na célula.
3. Excluir rótulo do grupo escolhido no Passo 2 da célula.
4. Realocar objetos rotulados pelo grupo escolhido a centróides dos outros grupos existentes na célula.

O segundo operador (Operador de Divisão) pode ser aplicado em qualquer grupo de uma célula que possua mais que dois objetos. Este operador divide um grupo, escolhido aleatoriamente, em dois novos grupos sendo que os objetos mais próximos do centróide do grupo original dão origem a um novo grupo e os objetos mais próximos do objeto mais distante do centróide dão origem a outro novo grupo. Como exemplo considere a célula ilustrada pela Figura 4.3.

Célula 2 – [113413413134433]

Figura 4.3: Célula utilizada para operador 2.

Foi escolhido aleatoriamente o grupo 4 ao qual será aplicado o operador de mutação 2. Na Célula 2 ilustrada na Figura 4.3 o grupo 4 vai ser dividido em dois grupos, um grupo

formado pelos objetos 4, 12 e 13 (Grupo 5) e outro grupo formado apenas pelo objeto 7 (Grupo 6). Esses grupos com suas novas formações são ilustrados na Figura 4.4.

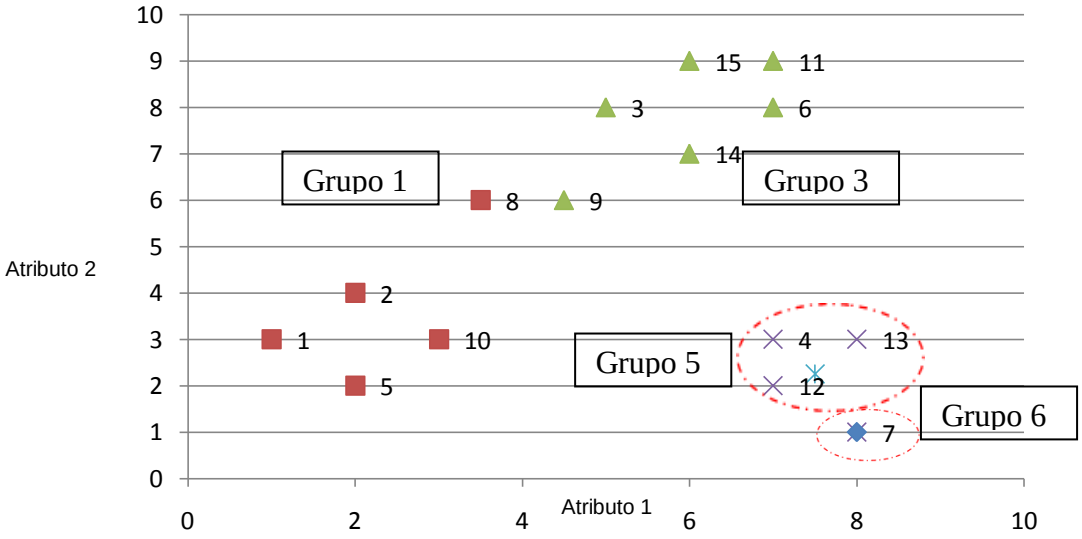


Figura 4.4: Agrupamento codificado pela Célula 2 através do qual o grupo 4 forma dois novos grupos (4, 12 e 13; 7).

No Algoritmo 4.3 é apresentado o pseudocódigo do Operador de Divisão implementado pelo EAC.

Algoritmo 4.3: Operador de Divisão.

1. Escolher aleatoriamente um grupo da célula.
2. Verificar se grupo possui mais de dois objetos.
3. Calcular centróide do grupo escolhido.
4. Calcular similaridade entre objetos e centróide e armazenar objeto mais distante.
5. Realocar objetos pela distância entre centróide e objeto mais distante; objetos mais próximos do centróide formarão um novo grupo enquanto objetos mais próximos do objeto armazenado (mais distante do centróide) formarão outro grupo.

4.2 DESCRIÇÃO DO ALGORITMO IMUNOLÓGICO

Duas versões para o algoritmo *ocopt-aiNet* são propostas nesta dissertação. A primeira versão (Versão 1), inspirada no EAC, possui uma busca local realizada por meio do algoritmo *k-médias*, e a segunda versão (Versão 2) é idêntica a primeira, porém sem a busca local pelo

k-médias. Espera-se encontrar uma maior diversidade com esta retirada da busca local, pois o algoritmo *k-médias* pode ter a tendência de encontrar soluções candidatas (células da rede) semelhantes entre si.

A *ocopt-aiNet* foi desenvolvida com o objetivo de minimizar alguns dos inconvenientes encontrados nos algoritmos clássicos de agrupamento, a saber, a proposição de uma única solução de agrupamento a cada execução, a necessidade de especificação a priori do número de grupos da base, e a busca local em torno de uma região específica do espaço. Sendo assim, o algoritmo *ocopt-aiNet* apresenta as seguintes características:

- Determinação automática do número de grupos da base;
- Ampla exploração do espaço de busca;
- Geração de múltiplos particionamentos simultâneos para a base de dados, permitindo uma análise mais exploratória e multimodal da base.

Para descrever o algoritmo *ocopt-aiNet* a terminologia a ser utilizada é a mesma da *opt-aiNet*:

- *Célula*: representa um indivíduo da população que codifica uma solução candidata de agrupamento;
- *Clones*: células de novas gerações copiadas a partir das células já existentes e sujeitas à mutação;
- *Afinidade*: grau de similaridade entre as células, avaliado através de uma análise de sua codificação para o agrupamento;
- *Supressão*: eliminação de células da população;
- *Fitness*: valor de cada célula em relação a uma função objetivo.

4.3 CODIFICAÇÃO

Na *opt-aiNet* cada célula da rede é representada por um vetor de valores reais no espaço Euclidiano. A codificação do algoritmo *ocopt-aiNet* é inspirada no EAC, sendo assim, substancialmente diferente da *opt-aiNet*, incorporando o rótulo de cada grupo da base de dados na codificação.

Considerando uma base contendo N objetos, cada célula representa uma solução candidata para o problema de agrupamento proposto e sua estrutura é um vetor de valores inteiros de N posições que representam os objetos da base. Cada posição conterá um rótulo (valor inteiro) que indica o grupo ao qual o objeto pertence. Se a solução possui k grupos

então a possibilidade de rótulos para um objeto será {1,...,*k*}. Como exemplo pode-se considerar a célula apresentada na Figura 4.5.

Célula 1- [111122221133333]

Figura 4.5: Exemplo de codificação do algoritmo ocopt-aiNet.

A célula ilustrada na Figura 4.5 possui quinze posições, sendo assim codifica uma base que contém quinze objetos. Nesse caso seis objetos {1,2,3,4,9,10} formam o grupo 1, o grupo de rótulo 2 é formado por quatro objetos {5,6,7,8} e o grupo de rótulo 3 é formado por cinco objetos {11,12,13,14,15}.

4.4 GERAÇÃO INICIAL

Assim como a *opt-aiNet*, no algoritmo *ocopt-aiNet*, tem-se como população inicial células geradas aleatoriamente, mas para o algoritmo *ocopt-aiNet* assume-se uma quantidade máxima de grupos para as posições do vetor que representa cada célula, processo também presente no EAC. Faz-se uma escolha aleatória para cada posição da célula indicando a qual grupo o objeto representado por aquela posição estará alocado inicialmente. Assumindo um valor qualquer *k* de grupos iniciais, para cada posição poderá ser escolhido um valor variando de 1,..,*k*, $k > 1$. O número de células iniciais também poderá ser definido preliminarmente pelo usuário ou então escolhido aleatoriamente.

Um exemplo de um repertório (população) de cinco células é apresentado na Figura 4.6:

Célula 1 - [111525453111123]
Célula 2 - [324451235555322]
Célula 3 - [111225424133314]
Célula 4 - [352541452411132]
Célula 5 - [152342251133455]

Figura 4.6: Exemplo de população da ocopt-aiNet.

4.5 REFINAMENTO DOS AGRUPAMENTOS

Após a geração da população inicial é utilizado o algoritmo *k-médias* para o refinamento (busca local) dos grupos codificados pelas células (NALDI, 2008). Assim como no EAC, o objetivo desse refinamento é o de executar uma aproximação do objetivo do algoritmo de encontrar agrupamentos ótimos de dados, manipulando cada célula da rede para uma melhor organização.

Como apresentado anteriormente, o algoritmo *k-médias* é um algoritmo do tipo particional: a partir de uma base contendo *N* objetos o *k-médias* particiona esta base em *k* grupos. O valor *k* representa a quantidade de partições geradas e no algoritmo *ocopt-aiNet* o valor *k* que será utilizado está representado em cada célula a ser feita a busca local. Dois parâmetros podem ser utilizados como critério de parada para o algoritmo: o número máximo de iterações e a diferença absoluta máxima entre os centróides de duas iterações consecutivas (HRUSCHKA; DE CASTRO; CAMPELLO, 2006). No caso de sua aplicação dentro da *ocopt-aiNet* apenas o parâmetro de máximo de iterações (*max_it*) está sendo utilizado. O *k-médias* empregado no algoritmo *ocopt-aiNet* está resumido no Algoritmo 4.4.

Algoritmo 4.4: Algoritmo k-médias utilizado no algoritmo *ocopt-aiNet*.

---

1. Assumir número *k* de grupos gerado a partir do número de grupos de cada célula;
2. **Enquanto** número de iterações $N < max\_it$, **faça**
   2.1. Calcular o centróide dos *k* grupos das partições existentes;
   2.2. Codificar cada objeto da base de dados com o centróide mais próximo e retorna ao Passo 2.
3. **Fim Enquanto.**

4.6 MUTAÇÃO

São utilizados três operadores de mutação no algoritmo *ocopt-aiNet*. Os dois primeiros operadores (Operadores de Exclusão e Divisão) foram inspirados na implementação dos operados evolutivos encontrados no EAC e foram apresentados anteriormente.

O terceiro operador de mutação (Operador de Transformação) poderá ser aplicado em todas as posições de uma célula e tem probabilidade de ocorrência $P = 10\%$, calculada para cada posição. Esse operador verifica se o objeto correspondente à posição encontra-se no

grupo correto. Para isso é verificada a similaridade do objeto aos centróides dos grupos formados pela célula, caso ele esteja no grupo correto, ou seja, tenha similaridade maior com o centróide de seu próprio grupo do que com os dos outros seu rótulo é mantido; caso exista um centróide de outro grupo com o qual este objeto tenha maior similaridade do que com o centróide do grupo a que ele está alocado então o rótulo é alterado para o desse centróide com maior similaridade.

Apenas um operador de mutação pode ser aplicado para cada clone. Portanto, para o caso de um clone sofrer mutação, os operadores 1 e 2 tem probabilidade *Pm* = 25% de serem aplicados, enquanto o operador 3 tem probabilidade *Pm* = 50% de ser aplicado.

Para cada clone *i* a probabilidade de aplicação de um operador de mutação segue a Equação 4.1.

$$\mathrm{P}(i) = 1 - f \tag{4.1}$$

na qual *f* é o valor do *fitness* da célula pai e P(*i*) é o valor da probabilidade de ser aplicada a mutação na célula *i*, *i*=1,..,*M*, sendo *M* o tamanho da população na iteração atual. Dessa forma os clones com célula pai que possuem maior *fitness* terão probabilidade menor de sofrerem mutação.

4.7 FUNÇÃO OBJETIVO

Assim como o algoritmo EAC, a função objetivo utilizada para avaliar cada célula da população é baseada no *critério da Silhueta*, considerado uma estratégia robusta para a predição de agrupamentos ótimos de dados (BOLSHAKOVA; AZUAJE, 2002).

Para explicar esse conceito, considere uma base de dados qualquer com *N* objetos. Seja um objeto *i* pertencente a um grupo *A* e a dissimilaridade média entre os outros objetos de *A* em relação a *i* denotada por *a*(*i*). Seja outro grupo *B*, a dissimilaridade do objeto *i* em relação aos objetos de *B* é denotada por *d*(*i*,*B*). Após o cálculo de *d*(*i*,*B*) para todos os grupos $B \neq A$, o de menor valor é escolhido, ou seja, $b(i) = min\ d(i,B)$, $B \neq A$. Este valor representa o valor de dissimilaridade do objeto *i* para o grupo mais próximo. A *Silhueta* é dada pela Equação 4.2:

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}} \tag{4.2}$$

O valor $s(i) \in [-1, +1]$, sendo que quanto maior o valor de $s(i)$ maior é a proximidade do objeto i com um dado grupo. Se o valor de $s(i)$ for zero, então não é possível definir claramente se este objeto i deveria estar no grupo atual ou em outro grupo próximo. Se um grupo A possui apenas um valor então $s(i)$ não é definida e é considerada zero. O *critério da Silhueta* é dado pela média de $s(i)$, $i = 1,..,N$ e a melhor forma de agrupamento é encontrada maximizando esta função (KAUFMAN; ROUSSEEUW, 1990; BOLSHAKOVA; AZUAJE, 2002; HRUSCHKA; DE CASTRO; CAMPELLO, 2006; ARGOUD; GONÇALVES; TIBERTI, 2008).

Para o algoritmo imunológico apresentado nesse trabalho a *Silhueta* foi alterada baseando-se no mesmo critério utilizado pelo algoritmo *EAC* (HRUSCHKA; DE CASTRO; CAMPELLO, 2006). No EAC o *critério da Silhueta* é alterado para que ao invés de ser calculada a dissimilaridade do objeto com todos os outros objetos encontrados na base, a dissimilaridade do objeto tem seu cálculo feito a partir dos centróides de cada grupo encontrado na célula. Sendo assim, o termo $a(i)$ da Equação 4.2 é alterado para ser a dissimilaridade do objeto i ao centróide do seu grupo A. Similarmente, o cálculo do termo $d(i,B)$, que trata da dissimilaridade do objeto i para os objetos do grupo B, $B \neq A$, é alterado para ser a dissimilaridade do objeto i ao centróide do grupo B, $B \neq A$. Esta alteração permite uma redução de tempo computacional e mantém uma semelhança com os operadores de mutação 1 e 2 e também com a busca local (HRUSCHKA; DE CASTRO; CAMPELLO, 2006) apresentados anteriormente.

Para o cálculo da dissimilaridade entre dois objetos **x** e **y** é utilizada a distância Euclidiana:

$$d(\boldsymbol{x}, \boldsymbol{y}) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2} \tag{4.3}$$

na qual x_i representa cada um dos n atributos do objeto **x** e y_i representa cada um dos n atributos do objeto **y**. Por fim, a distância $d(\mathbf{x},\mathbf{y})$ entre os objetos **x** e **y** representa a dissimilaridade entre esses objetos, de forma que quanto maior a distância maior a dissimilaridade, e vice-versa.

4.8 AFINIDADE E SUPRESSÃO

No algoritmo imunológico a afinidade tem como objetivo principal suprimir células que apresentem codificações de agrupamentos semelhantes.

Para a *opt-aiNet* o cálculo da afinidade é feito através da distância Euclidiana entre as células da rede. Para o algoritmo *ocopt-aiNet* a afinidade das células mantidas para a próxima geração é calculada a partir da quantidade de objetos rotulados para um mesmo grupo em duas células de memória, mesmo que rótulos distintos sejam usados para codificar os mesmos grupos. Não se deve utilizar o critério de *fitness* para a afinidade, pois os valores de *fitness* de duas células podem ser iguais ou muito parecidos, mesmo quando as codificações dos agrupamentos gerados pelas células são diferentes.

O cálculo de Afinidade está ilustrado na Equação 4.4.

$$Af(a,b) = \frac{N_i}{N} \tag{4.4}$$

na qual *Af*(*a*,*b*) é o valor da afinidade entre as Células *a* e *b*, N_i representa o total de objetos rotulados em um mesmo grupo nas Células *a* e *b* e *N* representa o número total de objetos de uma célula.

Considere a rede ilustrada na Figura 4.7 que possui quatro células e representam uma base com quinze objetos.

Célula 1 - [112233333334444]
Célula 2 - [221133333334444]
Célula 3 - [121233333334444]
Célula 4 - [121133333334444]

Figura 4.7: Células da rede após a primeira iteração do algoritmo.

Pode-se verificar que as soluções representadas pelas Células 1 e 2, apesar de possuírem rotulações diferentes para alguns grupos, representam um mesmo agrupamento. Então tem-se que *Af*(1,2) = 1, neste caso $N = N_i = 15$;

No caso das soluções representadas pelas células 3 e 4 existe uma alta afinidade, mas que não é máxima. A análise dessas células está ilustrada na Figura 4.8.

Célula 3	- Grupo I	-	Objetos: 1,3
	- Grupo II	-	Objetos: 2,4
	- Grupo III	-	Objetos: 5,..,11
	- Grupo IV	-	Objetos: 12,..,15
Célula 4	- Grupo I	-	Objetos: 1,3,4
	- Grupo II	-	Objetos: 2
	- Grupo III	-	Objetos: 5,..,11
	- Grupo IV	-	Objetos: 12,..,15

Figura 4.8: Grupos gerados pelas células 3 e 4.

Faz-se a comparação entre as células pelos grupos que possuem maior similaridade (maior número de objetos em um mesmo grupo). O Grupo 3.I é comparado ao Grupo 4.I e o Grupo 3.II é comparado com o Grupo 4.II e assim por diante. A diferença entre as células está ilustrada na Figura 4.9.

Grupo 3.I *e Grupo* 4.I	-	1 *objeto diferente*
Grupo 3.II *e Grupo* 4.II	-	1 *objeto diferente*
Grupo 3.III *e Grupo* 4.III	-	0 *objeto diferente*
Grupo 3.IV *e Grupo* 4.IV	-	0 *objeto diferente*

Figura 4.9: Comparação dos grupos da Célula 3 e 4.

O objeto que difere as duas comparações é o objeto 4 e deve ser considerado apenas uma vez para comparação. Sendo assim, não se deve replicar o número de objetos rotulados diferentemente; apenas uma diferença entre as duas células deve ser considerada. Para as células 3 e 4 o resultado é $Af(3,4) = 0{,}9334$. Portanto, o algoritmo de cálculo da afinidade entre células considera os agrupamentos formados por cada célula e calcula a similaridade entre eles, independentemente da rotulagem dos dados.

O cálculo da supressão é apresentado pela Equação 4.5.

$$\alpha = 1 - Af \tag{4.5}$$

na qual α é o limiar de supressão, um valor é escolhido a priori para a comparação e Af é o valor da afinidade. Caso esse valor escolhido a priori seja $\alpha = 0$, ou seja, apenas células com $Af = 1$ são suprimidas, para o exemplo apresentado acima apenas a célula 2 seria suprimida, e as outras células seriam mantidas na rede para a próxima iteração do algoritmo.

4.9 PSEUDOCÓDIGO

O algoritmo *ocopt-aiNet* possui os passos resumidos no Algoritmo 4.5. Ela possui dois critérios de parada. O primeiro critério (Passo 2.8), chamado de *critério local* (δ), utiliza a estabilidade do *fitness* das células na população para determinar se a diferença entre as células na iteração atual e as células da iteração anterior é menor que um limiar determinado pelo usuário; o algoritmo passa para o próximo passo caso a diferença seja menor que esse limiar. O segundo critério (Passo 2), chamado *critério global* (Δ), tem por objetivo determinar a estabilidade dos grupos gerados em um determinado número de iterações, chamado de *janela de estabilidade* (*w*). Segundo esse critério, caso a diferença do número médio de grupos não alcance o limiar, o algoritmo finaliza a execução, indicando que o número de grupos das células esteve estável durante a janela de estabilidade.

Algoritmo 4.5: Algoritmo *ocopt-aiNet*.

1. Iniciar a população com *N* células aleatoriamente.
2. **Enquanto** valor médio do número de grupos de *w* tem diferença menor que Δ, **faça**
   - *2.1. *Aplicar refinamento de agrupamentos as células da rede.*
   - 2.2. Avaliar cada célula de acordo com seu *fitness*.
   - 2.3. Aplicar normalização linear no *fitness*.
   - 2.4. Gerar um número *Nc* de clones para cada célula pai.
   - 2.5. Aplicar mutação proporcional o *fitness* da célula pai, mantendo célula pai.
   - 2.6. Determinar o *fitness* de todas as células da população.
   - 2.7. Para cada clone, selecionar o de maior *fitness* e calcular o *fitness* médio da população selecionada.
   - 2.8. Se diferença entre *fitness* médio da população em relação ao *fitness* médio da iteração anterior for menor que δ, continuar. Senão, voltar ao Passo 2.
   - 2.9. Determinar a afinidade de todas as células na população. Suprimir as células com afinidade menor que o limiar de supressão (σ).
   - 2.10. Introduzir um percentual *d%* de células geradas aleatoriamente e retornar ao Passo 2.
3. **Fim Enquanto.**

** A versão do algoritmo apresentada aplica a busca local; em uma segunda versão esse passo é retirado.*

O comportamento do algoritmo pode ser resumido da seguinte forma. Passo 2.1 a 2.7: para cada iteração do algoritmo as células presentes na rede sofrem processos de mutação até que se encontre estabilidade; Passo 2.8 a 2.10: o algoritmo encontrou a estabilidade local para as células presentes nessa iteração baseando-se no *fitness* de suas células e então as células irão interagir entre elas para que ocorra o processo de avaliação da afinidade e finalmente a supressão das células com maior afinidade evitando assim que células redundantes permaneçam na rede. Após a avaliação da afinidade e supressão, novas células geradas aleatoriamente são introduzidas na rede, mantendo assim a característica de uma maior exploração do espaço de busca pelo algoritmo e então o algoritmo reinicia sua execução pelo processo de estabilidade local.

5 AVALIAÇÃO DE DESEMPENHO

Nessa seção são apresentados e discutidos os experimentos realizados com o algoritmo *ocopt-aiNet* em suas duas versões. Para efeitos comparativos foram utilizados dois algoritmos encontrados na literatura e já citados nessa dissertação - *k-médias* e o *EAC* (*Evolutionary Algorithms for Clustering*).

O *EAC* foi desenvolvido com o objetivo de encontrar automaticamente partições ótimas de uma base de dados e o algoritmo *k-médias* é um algoritmo clássico para agrupamento de dados que utiliza o método de geração de *k* partições.

5.1 MATERIAIS E MÉTODOS

Foram utilizadas quatro bases de dados da literatura para os testes comparativos entre os algoritmos.

A primeira base de dados, chamada *Auto MPG*, está disponível no Repositório de Bases de Dados de Aprendizagem de Máquina da Universidade da Califórnia em Irvine (NEWMAN et al., 1998). Esta base possui 396 objetos (cada objeto representa um carro), cada um com 8 atributos, sendo 7 deles numéricos e 1 nominal; nessa base existem objetos com valores ausentes, ou seja, objetos que não possuem um valor definido para algum atributo, e foram retirados da base para os experimentos a serem apresentados. Assim, ao invés de 398 objetos serem utilizados para as execuções dos algoritmos, 392 objetos foram utilizados, pois 6 objetos apresentavam atributos com valores ausentes.

A segunda base, chamada *Animais*, possui 16 objetos com 13 atributos binários cada (HAYKIN, 1999); não há valores ausentes. Esta base está ilustrada na Tabela 2.1 na Seção 2.2.

A terceira base, chamada *Ruspini* (KAUFMAN; ROUSSEEUW, 1990), também ilustrada na Seção 2.2, possui 75 objetos com 2 atributos cada, sendo ambos atributos numéricos e não possui valores ausentes. Esta base possui 4 grupos conhecidos a priori que serão utilizados para validar se os algoritmos podem aproximar ou encontrar um agrupamento ótimo para essa base. Os grupos dessa base possuem 20, 23, 17 e 15 objetos, respectivamente.

A quarta base, chamada *Yeast*, é uma base de dados de bioinformática (YEUNG et al., 2003). Essa base possui 205 objetos e 20 atributos cada, sendo que todos os atributos são

numéricos e não possui valores ausentes. Também possui grupos conhecidos a priori, são 4 grupos dividindo a base em 83, 15, 93 e 14 objetos, respectivamente.

Para cada atributo das bases foi aplicada uma *normalização max-min*:

$$a' = \frac{a - min_a}{max_a - min_a}(novo\_max_a - novo\_min_a) + novo\_min_a \quad (5.1)$$

na qual *a'* representa o valor normalizado, *a* representa o valor que se quer normalizar, max_a e min_a são os valores máximo e mínimo encontrados no atributo a ser normalizado e $novo\_max_a$ e $novo\_min_a$ são os novos valores máximo e mínimo, respectivamente, possíveis para este atributo. Esta normalização realiza uma transformação linear nos dados originais fazendo com que os valores antigos sejam mapeados para novos valores no domínio [$novo\_min_a$, $novo\_max_a$]. Para os testes realizados nessa dissertação os valores $novo\_max_a$ e $novo\_min_a$ foram escolhidos como sendo: $novo\_max_a = 1$ e $novo\_min_a = 0$.

Os algoritmos foram implementados utilizando a linguagem C# e os testes foram realizados em um microcomputador com as configurações a seguir:

- Processador: AMD Turion X2 1.8 Ghz;
- Memória: 2Gb de memória RAM;
- Sistema Operacional: Windows XP Professional x64 Edition SP2.

Para o algoritmo *ocopt-aiNet* os parâmetros de inicialização utilizados, para ambas as versões, foram (sendo *N* o número de células iniciais; *Nc* o número de clones gerados para cada célula presente na rede; *k* o número inicial de grupos a serem gerados aleatoriamente; δ o critério de parada local; Δ o critério de parada global; *w* o número de iterações da janela de estabilidade; σ o limiar de supressão; *d* a porcentagem de novos indivíduos a serem inclusos a cada iteração, baseado no valor inicial de células introduzidas na rede):

- $N = 20$; $Nc = 10$; $k = 10$.
- $\delta = 0.001$; $\Delta = 0.001$; $w = 20$.
- $\sigma = 0.1$; $d = 10\%$.
- k-médias: $max\_it = 5$ (quando utilizado).

Com o algoritmo *EAC* foram utilizados os parâmetros iniciais configurados conforme segue (sendo *N* o número de indivíduos da população; *num_it* o número de iterações e *k* o número inicial de grupos a serem gerados aleatoriamente):

- $N = 20$; $num\_it = 5000$; $k = 10$.
- k-médias: $max\_it = 5$.

Como o algoritmo *k-médias* exige que o número *k* de partições seja uma entrada do usuário e este número se mantém como o número de partições a ser gerado no resultado do algoritmo, foi realizado um número maior de testes variando o valor *k*.

Para cada base de dados foram feitas 10 execuções dos algoritmos *ocopt-aiNet* (versão 1 e 2) e *EAC*. Para o algoritmo *k-médias* calculou-se o número de grupos gerados e o desvio padrão das execuções dos algoritmos a serem comparados e utilizou-se esse valor para variar o *k* inicial escolhido a priori. Ou seja, se o algoritmo *EAC* e *ocopt-aiNet* encontrarem uma média de 10 grupos com desvio padrão médio próximo de 2, então tem-se a execução do algoritmo *k-médias* com valores de *k*=8,..,12, sendo que, para cada valor de *k*, são feitas 10 execuções.

Para avaliar as partições obtidas foi utilizado o valor da função objetivo implementada tanto no algoritmo *EAC* quanto na *ocopt-aiNet*, a *Silhueta*. Essa função apresenta a qualidade das partições obtidas pelas simulações dos algoritmos. A função objetivo normalizada varia de [0,1], sendo que valores maiores representam melhores soluções para o agrupamento.

Um dos objetivos da *ocopt-aiNet* é obter múltiplas partições de boa qualidade em uma mesma simulação. Por isso, foi utilizado um índice para a avaliação da diversidade das partições obtidas, o *Adjusted Rand Index* (YEUNG; RUZZO, 2001; KUNCHEVA; HADJITODOROV, 2004).

Para explicar o Adjusted Rand Index considere a matriz de confusão de duas possíveis partições A e B, como ilustrado na Tabela 5.1. As linhas correspondem aos grupos de *A* e as colunas correspondem aos grupos de *B*. *Nij* é o valor na posição *i, j* da matriz de confusão, correspondente ao número de objetos no grupo *i* da partição *A* e grupo *j* na partição *B*. *Ni* é o valor da soma de todas as colunas da linha *i*; dessa forma, tem-se *Ni* correspondendo ao número de objetos no grupo *i* da partição *A*. *Nj* é o valor da soma de todas as linhas da coluna *j*; dessa forma, tem-se *Nj* correspondendo ao número de objetos no grupo *j* da partição *B*. Tendo duas partições *A* e *B* geradas pelos algoritmos não existe a necessidade de que ambas as partições tenham mesmo número de grupos. Tendo *Ga* como o número de grupos de *A* e *Gb* o número de grupos de *B*, o valor do *Adjusted Rand Index*, *ARI*, é calculado pelos valores *Nij* da matriz de confusão das duas partições (YEUNG; RUZZO, 2001; KUNCHEVA; HADJITODOROV, 2004), como apresentado nas Equações 5.2 e 5.3.

A/B	G_{b1}	...	G_{bN}	*Soma*
G_{a1}	$N_{1,1}$	...	$N_{1,j}$	N_{a1}
:	:		:	:
G_{aN}	$N_{i,1}$	...	$N_{i,j}$	N_{aj}
Soma	N_{b1}	...	N_{bi}	N

Tabela 5.1: Matriz de confusão das partições A e B.

A medida ARI tem variação no intervalo [–1,1], na qual um valor igual a 1 indica uma concordância perfeita entre partições, e valores próximos a 0 ou negativos correspondem a concordâncias encontradas ao acaso. Sendo assim, um valor baixo, próximo de 0, para o *Adjusted Rand Index* em partições de uma mesma simulação implica que o algoritmo avaliado gera partições com alta diversidade.

$$t1 = \sum_{i=1}^{Ga} \binom{Nbi}{2};\ t2 = \sum_{j=1}^{Gb} \binom{Naj}{2} \tag{5.2}$$

$$t3 = \frac{2t1t2}{N(N-1)}$$

$$ARI(A,B) = \frac{\sum_{i=1}^{Ga}\sum_{j=1}^{Gb}\binom{Nij}{2} - t3}{\frac{1}{2}(t1+t2) - t3} \tag{5.3}$$

Para apresentar o índice de diversidade será utilizado o resultado da Equação 5.4:

$$Di(\mathrm{A},\mathrm{B}) = 1 - \mathrm{ARI}(\mathrm{A},\mathrm{B}) \tag{5.4}$$

onde *Di* é o valor da diversidade encontrado para duas partições quaisquer A e B, A ≠ B e *ARI* é o valor encontrado no cálculo do *Adjusted Rand Index* para essas duas partições.

Além dessas métricas são apresentados os valores máximo, médio, mínimo e desvio padrão para o número de soluções e grupos nas soluções encontradas pelos algoritmos em todas as bases. Para todas as simulações o tempo foi armazenado e é apresentado na comparação entre os algoritmos.

As comparações são feitas pelos valores médios obtidos dos índices utilizados. Como o algoritmo *EAC* sempre apresenta apenas uma solução ao final de uma simulação então para a comparação é utilizada a média de todas as simulações. Para o *k-médias*, após todas as simulações para cada *k* inicial, é feita a média de todas as soluções obtidas. E da mesma forma

é feita para os algoritmos *ocopt-aiNet* versão 1 (*v1*) e versão 2 (*v2*), após todas as simulações são computados os valores obtidos por todas as soluções encontradas, como esses últimos apresentam múltiplas soluções em uma mesma simulação é feita a média de todas as soluções encontradas independentemente de simulações.

5.2 RESULTADOS E DISCUSSÕES

Nessa seção são apresentados os resultados e discussões sobre os testes que foram feitos utilizando os algoritmos citados. Para cada base de dados utilizada serão apresentados os resultados separadamente. Para as bases *Animais* e *Ruspini* são apresentados alguns agrupamentos resultantes dos testes com os algoritmos a fim de ilustrar as soluções encontradas por cada algoritmo.

O algoritmo *ocopt-aiNet* em suas duas versões está representado por *v1* (com *k-médias*) e *v2* (sem *k-médias*).

5.2.1 Auto MPG

Para essa base o algoritmo *k-médias* tem como parâmetro inicial *k* o intervalo fixado com os valores [3,...,7].

Na Tabela 5.2 são apresentados os valores da função objetivo para as simulações utilizando a base de *Auto MPG*. Por essa tabela pode-se verificar que o algoritmo *EAC* encontra uma mesma solução em todas as simulações, mantendo assim o valor da função objetivo constante. Já o *k-médias* tem em seu valor máximo a mesma solução encontrada pelo *EAC*, mas não mantém a qualidade para todas as simulações gerando soluções com baixa qualidade em alguns casos. Já os algoritmos *ocopt-aiNet v1* e *v2* mantêm em suas simulações um valor alto para a função objetivo, mostrando assim que as soluções encontradas são de boa qualidade. Devido à presença do algoritmo *k-médias* o algoritmo *ocopt-aiNet v1* obteve algumas soluções com baixo valor da função objetivo, diferente do algoritmo *ocopt-aiNet v2* que teve um desvio menor para suas soluções em todas as simulações.

	FUNÇÃO OBJETIVO			
	Max	*Méd*	*Min*	*Desvio*
ocopt-aiNet v1	0,743	0,730	0,664	0,023
ocopt-aiNet v2	0,737	0,725	0,713	0,007
EAC	0,774	0,774	0,774	---
k-médias	0,774	0,499	0,425	0,135

Tabela 5.2: Função objetivo para a base Auto MPG.

Analisando a diversidade e número de soluções obtidas, pela Tabela 5.3, pode-se confirmar que o número de soluções obtidas pelo *EAC* foi sempre o mesmo para todas suas simulações. O algoritmo *k-médias* obteve em uma de suas simulações sempre o mesmo valor, que pela função objetivo verificada anteriormente foi o mesmo encontrado pelo *EAC*. Em relação à diversidade entre todas as simulações do *k-médias* foram encontrados resultados contendo agrupamentos próximos entre si, diferentemente dos algoritmos *ocopt-aiNet v1* e *v2*. Esses últimos encontraram uma alta diversidade pela geração de múltiplas soluções. O algoritmo *ocopt-aiNet v1* obteve valores de diversidade melhores, indicando que seus indivíduos possuem diferenças significativas entre si, a versão *v2* apresentou um valor de diversidade menor, mas isso devido à presença de um número de indivíduos maior do que a versão *v1*, mostrando que a variedade de indivíduos encontrada foi maior.

	DIVERSIDADE				NRO. DE INDIVÍDUOS			
	Max	*Méd*	*Min*	*Desvio*	*Max*	*Méd*	*Min*	*Desvio*
ocopt-aiNet v1	0,511	0,489	0,466	0,017	13,00	11,40	9,00	1,173
ocopt-aiNet v2	0,518	0,436	0,364	0,039	41,00	30,50	26,00	4,089
EAC	0,000	0,000	0,000	---	1,00	1,00	1,00	---
k-médias	0,370	0,254	0,000	0,129	---	---	---	---

Tabela 5.3: Diversidade e número de soluções para a base Auto MPG.

Na Tabela 5.4 é apresentada a quantidade média de grupos obtida em todas as simulações dos algoritmos. O algoritmo *EAC*, como obteve apenas uma solução em todas as simulações, teve o número de grupos para a solução constante. O algoritmo *k-médias* obteve diferentes números de grupos em suas soluções dependendo de seu *k* inicial. O algoritmo *ocopt-aiNet v1* apresentou um número de grupos médio menor que sua versão *v2*, devido ao fato de ter presente o algoritmo *k-médias* que aproximou as soluções.

NÚMERO DE GRUPOS				
	Max	*Méd*	*Min*	*Desvio*
ocopt-aiNet v1	5,333	5,229	5,000	0,092
ocopt-aiNet v2	7,875	7,401	6,827	0,283
EAC	2,000	2,000	2,000	---
k-médias	6,600	4,300	2,000	1,748

Tabela 5.4: Número de grupos nas soluções para a base Auto MPG.

Pela comparação do tempo utilizado para cada simulação, Tabela 5.5, pode ser verificado que o algoritmo *ocopt-aiNet v2* exige tempo maior que os outros para finalizar e que tem uma alta variação entre suas simulações. O algoritmo *EAC* possui tempos semelhantes em suas simulações devido a finalizar sempre com o mesmo número de iterações. Em média o algoritmo *ocopt-aiNet v1* finaliza sua execução com menor tempo que o *EAC* mostrando que encontra a estabilidade de maneira rápida. O algoritmo *k-médias*, devido a sua dinâmica simples e eficiente, finaliza rapidamente, não sendo possível apresentar seus valores em segundos conforme apresentado por essa comparação.

TEMPO COMPUTACIONAL (hh:mm:ss)				
	Max	*Méd*	*Min*	*Desvio*
ocopt-aiNet v1	00:46:58	00:17:49	00:03:21	00:14:11
ocopt-aiNet v2	02:12:36	00:55:04	00:11:06	00:41:13
EAC	00:35:39	00:31:50	00:30:19	00:01:32
k-médias	00:00:00	00:00:00	00:00:00	---

Tabela 5.5: Tempo computacional para a base Auto MPG.

O algoritmo EAC apresentou o mesmo indivíduo ao final de todas as simulações para a base Auto MPG. Na Tabela 5.6 é apresentada uma parte do indivíduo resultante das simulações executadas. Pode-se verificar que os atributos Peso, Deslocamento (Desl) e Cilindros (Cil.) tiveram maior influência para a forma de agrupamento gerada pelo algoritmo.

Grupo	Carro	MPG	Cil.	Desl.	Pot.	Peso	Aceleração	Ano	Origem
1	Ford granada gl	20,2	6	200	88	3060	17,1	81	1
	Chrysler lebaron salo	17,6	6	225	85	3465	16,6	81	1
	Chevrolet monte carl	15	8	400	150	3761	9,5	70	1
	Buick estate wagon	14	8	455	225	3086	10	70	1
2	Toyota corona mark	24	4	113	95	2372	15	70	3
	Datsun pl510	27	4	97	88	2130	14,5	70	3
	Volkswagen 1131	26	4	97	46	1835	20,5	70	2
	Peugeot 504	25	4	110	87	2672	17,5	70	2

Tabela 5.6: Agrupamento da base Auto MPG pelo EAC.

Um indivíduo gerado pelo algoritmo *ocopt-aiNet v2* está representado em parte na Tabela 5.7. Ele teve um número maior de grupos que o *EAC* e assim apresentou um refinamento maior para esses grupos. Para este indivíduo verifica-se uma influência de vários atributos o que difere do agrupamento apresentado anteriormente. Para essa base existem relações entre os diversos atributos que a compõem. O atributo MPG (*Miles Per Gallon*, ou Milhas por Galão) indica o quanto um veículo consome de combustível. Pelos dados apresentados este atributo está relacionado ao deslocamento e a potência (Pot.) apresentados pelo carro e também a quantidade de cilindros (Cil.) influi na potência do carro. Por fim, o atributo Aceleração está bastante relacionado com o peso que o carro apresenta, carros mais pesados tendem a ter uma aceleração menor. Analisando os grupos formados verifica-se que os carros com maior quantidade de cilindros e, assim, maiores potências formaram grupos separados dos carros de menor quantidade de cilindros, e houve um refinamento nos carros com menor quantidade de cilindros formando mais de um grupo, nos quais existe a influência da potência juntamente com o peso e aceleração para esse refinamento.

Grupo	Carro	MPG	Cil.	Desl.	Pot.	Peso	Aceleração	Ano	Origem
1	Datsun b-210	32	4	85	70	1990	17	76	3
	Toyota corolla	28	4	97	75	2155	16,4	76	3
	Honda accord cvcc	31,5	4	98	68	2045	18,5	77	3
	Renault 5 gtl	36	4	79	58	1825	18,6	77	2
2	Dodge colt (sw)	28	4	98	80	2164	15	72	1
	Ford punto	19	4	122	85	2310	18,5	73	1
	Chevrolet vega	25	4	140	75	2542	17	74	1
	Dodge colt	28	4	90	75	2125	14,5	74	1
3	Ford maverick	21	6	200	85	2587	16	70	1
	Amc gremlin	21	6	199	90	2648	15	70	1
	Plymouth duster	22	6	198	95	2833	15,5	70	1
	Amc hornet	18	6	199	97	2774	15,5	70	1
4	Datsun pl510	27	4	97	88	2130	14,5	70	3
	Peugeot 504	25	4	110	87	2672	17,5	70	2
	Volkswagen 1131	26	4	97	46	1835	20,5	70	2
	Toyota corona mar	24	4	113	95	2372	15	70	3
5	Chevrolet chevelle	18	8	307	130	3504	12	70	1
	Buick skylark 320	15	8	350	165	3693	11,5	70	1
	Blymouth satellite	18	8	318	150	3436	11	70	1
	Amc rebel sst	16	8	304	150	3433	12	70	1

Tabela 5.7: Agrupamento da base Auto MPG pelo algoritmo ocopt-aiNet v2.

5.2.2 Animais

Para o algoritmo *k-médias* o parâmetro inicial *k* teve seu intervalo fixado com os valores [3,...,7] para essa base.

Pela Tabela 5.8 tem-se que o algoritmo *EAC* tem uma variação baixa da função objetivo tentando sempre se aproximar de uma mesma solução. Para os algoritmos *ocopt-aiNet v1* e *v2* verifica-se que possuem variações do valor da função objetivo maior que o *EAC*, mas a qualidade das soluções encontradas ainda fica acima das encontradas pelo algoritmo *k-médias*. Esta variação, que mantém ainda uma boa qualidade das soluções encontradas, ocorre devido à manutenção de várias soluções por estes algoritmos.

	FUNÇÃO OBJETIVO			
	Max	*Méd*	*Min*	*Desvio*
ocopt-aiNet v1	0,775	0,757	0,740	0,012
ocopt-aiNet v2	0,768	0,758	0,749	0,004
EAC	0,807	0,804	0,799	0,003
k-médias	0,759	0,742	0,730	0,012

Tabela 5.8: Função objetivo para a base Animais.

Pela análise da Tabela 5.9 tem-se que a diversidade encontrada pelos algoritmos *ocopt-aiNet v1 e v2* são maiores do que as produzidas pelos outros algoritmos. Em especial, o algoritmo *ocopt-aiNet v2* encontrou maior quantidade de indivíduos do que o *ocopt-aiNet v1*, além de ter produzido diversidade muito ampla. Observe-se ainda que, em conjunto, as Tabela 5.8 e 5.9 indicam que os indivíduos encontrados, além de diversificados, são de boa qualidade. O algoritmo *EAC*, mesmo avaliando essa base que não possui grupos definidos a priori, apresentou um número menor de indivíduos resultantes, gerando apenas 2 indivíduos em todas as simulações executadas. O *k-médias* apresentou diversidade menor que os algoritmos *ocopt-aiNet* mas pode-se considerar que existiu um fator limitante para esse algoritmo que foi a quantidade de simulações para cada valor inicial do parâmetro *k*.

	DIVERSIDADE				NRO. DE INDIVÍDUOS			
	Max	*Méd*	*Min*	*Desvio*	*Max*	*Méd*	*Min*	*Desvio*
ocopt-aiNet v1	0,510	0,441	0,356	0,061	6,00	4,60	3,00	0,966
ocopt-aiNet v2	0,646	0,622	0,603	0,011	51,00	44,40	34,00	5,274
EAC	0,549	0,549	0,549	---	2,00	2,00	2,00	---
k-médias	0,593	0,480	0,296	0,114	---	---	---	---

Tabela 5.9: Diversidade e número de soluções para a base Animais.

Pela análise do número de grupos gerados pelos indivíduos, apresentado na Tabela 5.10, pode-se verificar que o número de grupos gerados pelo algoritmo *ocopt-aiNet v2* é maior que o de todos os outros algoritmos. Como esse algoritmo tende a manter diversos indivíduos e não possui método interno de aproximação entre eles (busca local como a *v1*) o número de indivíduos é maior, mas pelo número de grupos quase sempre constante verifica-se que este número é aproximadamente aquele onde se encontram as melhores formas de agrupamento. O algoritmo *EAC* aproxima seu melhor resultado com o mesmo número médio de grupos da *ocopt-aiNet v2*, mostrando que as melhores soluções possíveis são encontradas com este número de grupos.

NÚMERO DE GRUPOS				
	Max	*Méd*	*Min*	*Desvio*
ocopt-aiNet v1	4,000	3,848	3,600	0,143
ocopt-aiNet v2	6,591	6,490	6,291	0,117
EAC	6,000	4,800	2,000	1,932
k-médias	4,500	3,940	3,000	0,680

Tabela 5.10: Número de grupos das soluções para a base Animais.

Verificando o tempo computacional necessário para as simulações, Tabela 5.11, o algoritmo *k-médias* sempre é mais rápido que os outros algoritmos devido a seu algoritmo simples e de fácil execução. Para o *EAC* e *ocopt-aiNet v2* existe semelhança nos tempos dos algoritmos, ambos obtiveram médias semelhantes, mostrando que para essa base o algoritmo encontra sua estabilidade global rapidamente. No algoritmo *ocopt-aiNet v1* a estabilidade foi encontrada de forma mais rápida que para os dois últimos algoritmos citados, mas isso devido à formação de um menor número de indivíduos que a versão *v2*.

TEMPO COMPUTACIONAL (hh:mm:ss)				
	Max	*Méd*	*Min*	*Desvio*
ocopt-aiNet v1	00:00:54	00:00:24	00:00:01	00:00:19
ocopt-aiNet v2	00:01:07	00:00:39	00:00:19	00:00:17
EAC	00:01:13	00:00:42	00:00:34	00:00:13
k-médias	00:00:00	00:00:00	00:00:00	---

Tabela 5.11: Tempo computacional para a base Animais.

Para ilustrar os grupos formados pelos indivíduos dos algoritmos, na Tabela 5.12 são apresentados agrupamentos encontrados em uma mesma simulação do algortimo *ocopt-aiNet*

v2 que apresentou melhores resultados para a base utilizada nos testes descritos nessa seção. Essa tabela possui cinco soluções encontradas pelo algoritmo, a tabela está dividida pelos grupos gerados pelo algoritmo (G1,..,G*n*, sendo *n* o número de grupos formados na solução) e a última linha da tabela apresenta o valor da função objetivo de cada solução apresentada. Pode-se verificar que as soluções 4 e 5 possuem o mesmo valor para a função objetivo, mas os grupos formados são diferentes. Ao analisar os dados presentes na Tabela 2.1, nota-se que as características dos animais agrupados são semelhantes. Dessa forma pode-se afirmar que o algoritmo foi capaz de encontrar mais de uma maneira de agrupamento para a base apresentada.

1	2	3	4	5
G1	G1	G1	G1	G1
Pombo	Pombo	Pombo	Pombo	Pombo
Galinha	Galinha	Galinha	Ganso	Galinha
Pato	Pato	Pato	G2	G2
Ganso	Ganso	Ganso	Galinha	Pato
Coruja	Coruja	Coruja	Pato	Ganso
Gavião	Gavião	Gavião	G3	G3
Águia	Águia	Águia	Coruja	Coruja
G2	G2	G2	Gavião	Gavião
Raposa	Raposa	Raposa	G4	G4
Cão	Cão	Gato	Águia	Águia
Lobo	Lobo	G3	G5	G5
Gato	Gato	Cão	Raposa	Raposa
Tigre	G3	Lobo	Gato	Gato
Leão	Tigre	G4	G6	G6
Cavalo	Leão	Tigre	Cão	Cão
Zebra	G4	Leão	Lobo	Lobo
Vaca	Cavalo	G5	G7	G7
	Zebra	Cavalo	Tigre	Tigre
	G5	Zebra	Leão	Leão
	Vaca	G6	G8	G8
		Vaca	Cavalo	Cavalo
			Zebra	Zebra
			G9	G9
			Vaca	Vaca
0,79921	0,79645	0,80709	0,80001	0,80001

Tabela 5.12: Agrupamentos da base Animais pela ocopt-aiNet v2.

5.2.3 Ruspini

Para o algoritmo *k-médias* o parâmetro inicial *k* teve seu intervalo fixado com os valores [4,...,6] para essa base.

Na Tabela 5.13 comparam-se os valores de função objetivo derivados de cada algoritmo, em todas as simulações. Verifica-se que o algoritmo *EAC*, por desempenhar uma busca pelo melhor agrupamento, tenderá a encontrar, ou se aproximar, de uma mesma solução. Para essa base, em todas as simulações a solução final sempre foi a mesma, correspondente ao agrupamento correto. Todos os outros algoritmos encontraram a solução ideal para essa base. Os algoritmos *ocopt-aiNet v1* e *v2* encontraram o agrupamento correto para essa base em todas as simulações. Já o algoritmo *k-médias* tem a dependência de seus centróides iniciais, muitas vezes esses centróides podem acabar apresentando soluções que não sejam as melhores para o agrupamento. Pode-se verificar que devido à ausência do algoritmo *k-médias* no algoritmo *ocopt-aiNet v2*, esta versão apresentou uma exploração maior e algumas soluções de qualidade menor, mas mesmo assim manteve uma média próxima dos outros algoritmos, mostrando que gerou também soluções de boa qualidade.

	FUNÇÃO OBJETIVO			
	Max	*Méd*	*Min*	*Desvio*
ocopt-aiNet v1	0,859	0,849	0,832	0,008
ocopt-aiNet v2	0,841	0,821	0,778	0,020
EAC	0,908	0,908	0,908	---
k-médias	0,872	0,865	0,854	0,009

Tabela 5.13: Função objetivo para a base Ruspini.

Na Tabela 5.14 são apresentados os valores de diversidade e indivíduos encontrados pelos algoritmos. Os algoritmos *ocopt-aiNet v1* e *v2* têm uma variação maior de diversidade dos indivíduos encontrados devido à manutenção de múltiplas soluções de boa qualidade. Verifica-se ainda que a versão *v2*, por não possuir o algoritmo *k-médias*, encontrou um número de soluções maior que a versão *v1* e então se pode afirmar que houve uma exploração mais ampla do espaço de busca por este algoritmo também para essa base. Para o algoritmo *k-médias* existe a presença de diversidade entre as soluções encontradas, mas esta diversidade é menor quando comparada com os algoritmos *ocopt-aiNet v1* e *v2*, mostrando que o algoritmo *k-médias* tem a tendência de aproximar as soluções, como destacado anteriormente.

	DIVERSIDADE				NRO. DE INDIVÍDUOS			
	Max	*Méd*	*Min*	*Desvio*	*Max*	*Méd*	*Min*	*Desvio*
ocopt-aiNet v1	0,414	0,311	0,352	0,077	5,00	3,80	3,00	0,632
ocopt-aiNet v2	0,497	0,365	0,288	0,063	18,00	12,20	7,00	3,359
EAC	0,000	0,000	0,000	---	1,00	1,00	1,00	---
k-médias	0,267	0,215	0,188	0,044	---	---	---	---

Tabela 5.14: Diversidade das soluções para a base Ruspini.

Na Tabela 5.15 são apresentados os números médios de grupos formados por cada algoritmo. Pode-se ilustrar por essa tabela que os algoritmos *ocopt-aiNet* executaram uma busca mais exploratória do que os outros algoritmos avaliados. Para o *k-médias* tem-se uma pequena variação no número de grupos devido ao intervalo escolhido de seu parâmetro *k* inicial. Apesar de seu desvio padrão ser maior que o dos outros algoritmos, pela análise anteriormente apresentada, o algoritmo tende a encontrar soluções próximas.

	NÚMERO DE GRUPOS			
	Max	*Méd*	*Min*	*Desvio*
ocopt-aiNet v1	6,333	5,548	4,750	0,509
ocopt-aiNet v2	7,071	6,678	6,000	0,307
EAC	4,000	4,000	4,000	---
k-médias	5,600	4,700	3,600	1,014

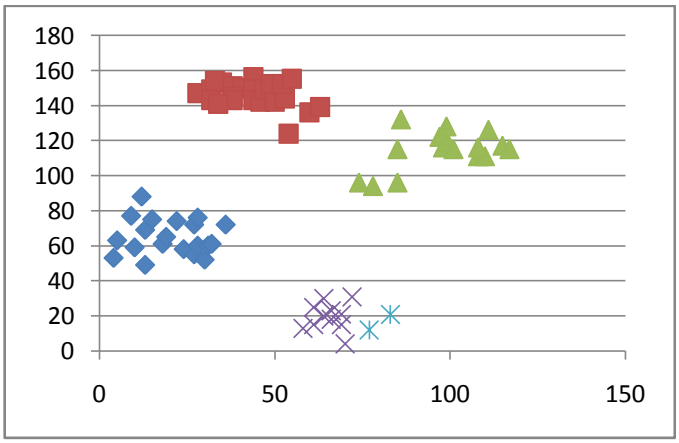
Tabela 5.15: Número de grupos das soluções para a base Ruspini.

Para a base *Ruspini* o tempo computacional, apresentado na Tabela 5.16, para os algoritmos *ocopt-aiNet v1* e *v2* teve alta variação, mesmo apresentando em alguns casos valores próximos. Isso ocorre devido à estabilidade do algoritmo *ocopt-aiNet v1* ser encontrada de maneira mais rápida e a presença do algoritmo *k-médias* favorecer para que essa versão estabilize de maneira mais rápida que a versão *v2*. O algoritmo *k-médias* obteve tempos menores que um segundo, mostrando que para qualquer base utilizada sua velocidade de execução é mantida. Comparando com o *EAC* os algoritmos *ocopt-aiNet v1* e *v2* obtiveram melhores médias de tempo para estabilidade global, mostrando que os algoritmos tendem a encontrar de maneira rápida soluções de boa qualidade.

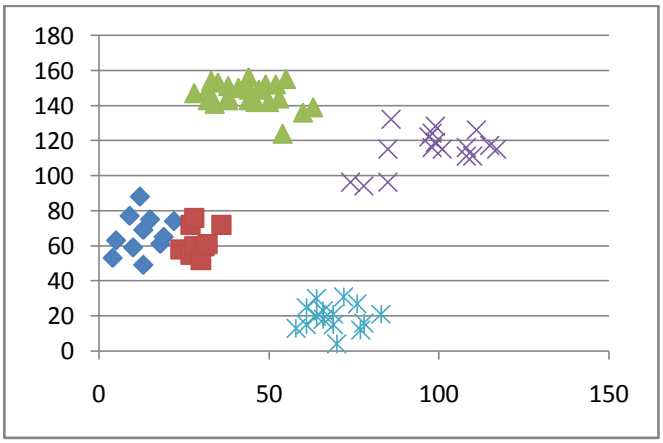
TEMPO COMPUTACIONAL (hh:mm:ss)				
	Max	*Méd*	*Min*	*Desvio*
ocopt-aiNet v1	00:01:16	00:00:26	00:00:05	00:00:22
ocopt-aiNet v2	00:05:34	00:01:57	00:00:08	00:02:00
EAC	00:02:31	00:02:09	00:01:41	00:00:19
k-médias	00:00:00	00:00:00	00:00:00	---

Tabela 5.16: Tempo computacional para a base Ruspini.

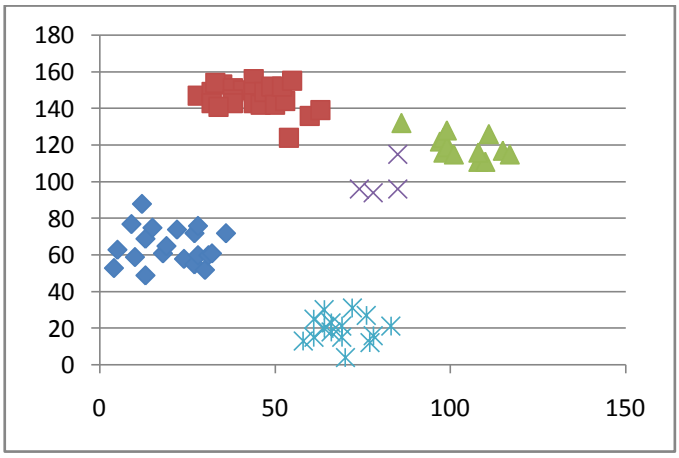
Na Figura 5.1 estão ilustrados os três agrupamentos (objetos de diferentes grupos representados por diferentes formas geométricas e níveis de cinza) com maior valor da Função Objetivo que foram apresentados na saída dos algoritmos *ocopt-aiNet*, já excluindo o agrupamento correto que foi encontrado em todas as simulações. Pela ilustração pode-se notar que os algoritmos apresentados podem encontrar grupos com um maior refinamento, ou seja, separa objetos mais próximos que formavam apenas um grupo para então formar novos grupos. Este fato pode ser relevante para algumas bases, como visto para o caso da base de dados dos carros.



a) Fitness: 0,89381



b) Fitness: 0,87405



c) Fitness: 0,87988

Figura 5.1: Agrupamentos para a base Ruspini.

5.2.4 Yeast

Para a base *Yeast* o algoritmo *k-médias* foi capaz de manter soluções de boa qualidade com os parâmetros iniciais utilizados para sua simulação. Essas soluções foram, em média, melhores do que as encontradas pelos algoritmos *ocopt-aiNet v1* e *v2* (Tabela 5.17). Para esses últimos algoritmos os valores da função objetivo mantiveram-se próximos em todas as simulações, não obtendo uma variação grande como pode ser analisado pelo desvio padrão encontrado.

	FUNÇÃO OBJETIVO			
	Max	*Méd*	*Min*	*Desvio*
ocopt-aiNet v1	0,782	0,762	0,742	0,011
ocopt-aiNet v2	0,759	0,737	0,704	0,017
EAC	0,841	0,841	0,841	---
k-médias	0,839	0,805	0,771	0,026

Tabela 5.17: Função objetivo para a base Yeast.

Para a diversidade os algoritmos *ocopt-aiNet v1* e *v2* apresentaram valores melhores do que os encontrados pelos algoritmos *k-médias* e o *EAC*. O algoritmo *EAC* em todas as simulações apresentou apenas um indivíduo para a base, não existindo a possibilidade de se analisar a diversidade dos indivíduos. O algoritmo *ocopt-aiNet v2* encontrou um número maior de indivíduos e com diversidade maior que os outros algoritmos avaliados. Sendo assim, apresenta uma qualidade menor de indivíduos. Pela Tabela 5.18 verifica-se a diversidade e indivíduos encontrados pelos algoritmos.

	DIVERSIDADE				NRO. DE INDIVÍDUOS			
	Max	*Méd*	*Min*	*Desvio*	*Max*	*Méd*	*Min*	*Desvio*
ocopt-aiNet v1	0,559	0,380	0,255	0,104	8,00	6,10	4,00	1,197
ocopt-aiNet v2	0,519	0,397	0,269	0,078	23,00	17,20	14,00	4,442
EAC	0,000	0,000	0,000	---	1,00	1,00	1,00	---
k-médias	0,246	0,146	0,056	0,077	---	---	---	---

Tabela 5.18: Diversidade e número de grupos para a base Yeast.

Pela análise do número de grupos encontrados, Tabela 5.19, o algoritmo *ocopt-aiNet v2* executou uma busca mais ampla do que os outros algoritmos. Os algoritmos *ocopt-aiNet v1* e *k-médias* apresentaram número próximo de grupos, o que mostra que o melhor agrupamento encontrado pelo *k-médias* fez com que o algoritmo *ocopt-aiNet v1* mantivesse

indivíduos próximos a este valor de número de grupos. Para o *EAC*, como apresentou apenas um indivíduo, então apenas um número de grupos é apresentado, sem variação.

Como a base *Yeast* possui classes conhecidas a priori, tem-se que destacar a ausência do agrupamento correto nas soluções obtidas por todos os algoritmos.

NÚMERO DE GRUPOS				
	Max	*Méd*	*Min*	*Desvio*
ocopt-aiNet v1	5,50	5,20	4,80	0,224
ocopt-aiNet v2	7,95	7,48	7,11	0,306
EAC	3,00	3,00	3,00	---
k-médias	5,40	4,22	2,80	0,998

Tabela 5.19: Número de grupos das soluções para a base Yeast.

Comparando os tempos computacionais dos algoritmos avaliados (Tabela 5.20), a *ocopt-aiNet v2* apresentou tempos elevados em relação aos outros algoritmos, obtendo uma variação grande entre as diversas simulações. A *ocopt-aiNet v1* apresentou um tempo mais próximo do algoritmo *EAC*, lembrando que esse algoritmo possui o número de iterações como critério de parada, dessa forma a variação entre as simulações é pequena. O algoritmo *k-médias*, como nas outras análises de tempo computacional, se apresenta como sendo um algoritmo simples, e assim, apresenta estabilidade de forma rápida.

TEMPO COMPUTACIONAL				
	Max	*Méd*	*Min*	*Desvio*
ocopt-aiNet v1	00:46:07	00:18:58	00:04:12	00:14:28
ocopt-aiNet v2	03:29:25	01:46:09	00:09:47	01:05:01
EAC	00:30:10	00:29:35	00:29:07	00:00:21
k-médias	00:00:00	00:00:00	00:00:00	---

Tabela 5.20: Tempo computacional para a base Yeast.

5.2.5 Discussões Gerais

Encontrar diversidade em bases de dados é uma capacidade importante para auxiliar na tomada de decisão por parte do usuário (HAN; KAMBER, 2000), permitindo uma análise mais ampla das relações existentes entre os objetos pertencentes à base. Dessa forma é importante destacar que, após a apresentação dos resultados comparativos das simulações

executadas para as quatro bases, o algoritmo *ocopt-aiNet v2* mostrou-se eficaz na geração e manutenção da diversidade para as bases apresentadas. Além disso, os indivíduos encontrados por esse algoritmo, mesmo que com alguma perda, são de boa qualidade, mostrando uma dinâmica eficaz para manutenção dos melhores indivíduos. Mesmo em casos em que o algoritmo *k-médias* apresentou uma qualidade média melhor, como para a base *Yeast*, a diversidade encontrada por esse algoritmo foi menor do que a encontrada pela *ocopt-aiNet v2*.

Para bases em que não existiam grupos bem definidos, como as bases Animais e Auto MPG, o algoritmo *ocopt-aiNet v2* apresentou um grande número de indivíduos como soluções candidatas para o agrupamento, diferentemente dos outros algoritmos que tiveram números menores mais frequentes.

Em contrapartida à diversidade, o algoritmo *ocopt-aiNet* apresenta o inconveniente de um esforço computacional mais elevado em relação aos outros algoritmos avaliados. Entretanto, é importante salientar que este é o preço pago tanto pela busca automática do número de grupos, quanto pela manutenção da diversidade na população. Fica evidente a existência de um equilíbrio entre multimodalidade, busca dinâmica da parametrização de grupos e eficiência computacional.

6 CONCLUSÃO E TRABALHOS FUTUROS

Nesse trabalho foi proposto um novo algoritmo imunológico para agrupamento de dados, a *ocopt-aiNet*, em duas versões. Ambas as versões possuem as características de determinação automática do número de grupos, a geração e manutenção de múltiplas soluções de boa qualidade em uma mesma execução do algoritmo e maior exploração do espaço de busca. A diferença entre as duas versões é a presença de busca local em uma delas, o que fez com que o algoritmo encontrasse um menor número de soluções nos experimentos, mas que também beneficiou essa versão no que tange ao encontro da estabilidade global do algoritmo.

Pelos experimentos pode-se destacar que a qualidade das soluções obtidas foi sempre mantida para todas as simulações realizadas com o algoritmo.

Para bases que possuem elevado número de objetos o tempo gasto pelo algoritmo proposto, em sua versão que não possui a busca local (*v2*), se mostrou elevado. Esse fato ocorre devido à dificuldade encontrada pelo algoritmo para estabilizar suas soluções, mas ao mesmo tempo faz com que o algoritmo encontre um maior número de soluções e, em alguns experimentos, de melhor qualidade e diversidade que os outros algoritmos.

Para bases que possuem alta diversidade, como as bases Auto MPG e Animais, os experimentos mostraram que o algoritmo proposto, principalmente em sua versão sem a busca local (*v2*), consegue encontrar alto número de soluções beneficiando assim uma possível análise sobre agrupamentos que podem ser formados para essas bases. Nesse caso, para bases em que não se conhecem os grupos a priori e que permitem a geração de grande número de soluções, o algoritmo proposto se mostra eficaz. Por fim, para casos em que existe a necessidade de se encontrar múltiplas soluções para uma mesma base o algoritmo *ocopt-aiNet* mostra que possui resultados relevantes. Em casos que a necessidade é encontrar um único agrupamento ótimo esse algoritmo não é tão eficiente devido ao inconveniente de que o custo computacional para sua execução se mostrou maior do que os outros algoritmos avaliados.

Como perspectivas de trabalhos futuros, destacam-se:

- Realização de novos experimentos variando os parâmetros da *ocopt-aiNet* com o objetivo de analisar a sensibilidade paramétrica da proposta e tentar melhorar os tempos encontrados para bases que contenham grande número de objetos.
- Testar o algoritmo em outras bases de dados de forma a investigar casos nos quais a multimodalidade de agrupamentos é relevante na prática.

- Comparar com outros algoritmos de agrupamento como os algoritmos que utilizam conceitos Fuzzy.
- Adaptar o algoritmo para aplicação em bases de dados textuais.
- Propor e avaliar diferentes operadores de mutação.
- Alterar a medida de dissimilaridade entre os objetos.

REFERÊNCIAS BIBLIOGRÁFICAS

ARGOUD, A. R. T. T.; GONÇALVES, E. V. F.; TIBERTI, A. J. Algoritmo genético de agrupamento para formação de módulos de arranjo físico. *Gestão de Produção*, v. 15, n. 2, São Carlos, 2008.

BOLSHAKOVA, N.; AZUAJE, F. Cluster validation techniques for genome expression data. Dublin, 2002. Disponível em: <https://www.cs.tcd.ie/publications/tech-reports/reports.02/TCD-CS-2002-33.pdf>. Acesso em 19 de out. de 2008.

BURNET, F. M. *The Clonal Selection Theory of Acquired Immunity*. Cambridge University Press. London, 1959.

CHEN, M.; HAN, J.; YU, P. S. Data Mining: An Overview from a Database Perspective. *IEEE Transactions on Knowledge and Data Engineering*, vol. 8, n. 6, pp. 866-883, 1996.

DASGUPTA, D.; JI, Z.; GONZALEZ, F. Artificial Immune System (AIS) Research in the Last Five Years. In: CEC'03. *Proceedings of IEEE Congress on Evolutionary Computation*.

DE CASTRO, L. N. *Fundamentals of Natural Computing*: Basic Concepts, Algorithms, and Applications. 1a Edição. Boca Raton: Chapman & Hall/CRC, 2006.

DE CASTRO, L. N. Fundamentals of natural computing: an overview. *Physics of Life Reviews*, v. 4, p. 1-36, 2007.

DE CASTRO, L. N.; TIMMIS, J. An Artificial Immune Network for Multimodal Function Optimization. In: CEC'02. *Proceedings of IEEE Congress on Evolutionary Computation*, Hawaii, 2002, v. 1, pp. 699-674.

DE CASTRO, L. N.; TIMMIS J. *Artificial Immune Systems*: A New Computational Intelligence Approach. 1a Edição. London: Springer, 2002.

DE CASTRO, L. N.; TIMMIS, J. I. Artificial immune systems as a novel soft computing paradigm. *Soft Computing* 7, pp. 526–544, Springer-Verlag, 2003.

DE CASTRO, L. N.; VON ZUBEN, F. J. The Clonal Selection Algorithm with Engineering Applications. In: GECCO'00. *Proceedings of GECCO'00*, Workshop on Artificial Immune Systems and Their Applications, pp. 36-37, 2000.

DE CASTRO, L. N.; VON ZUBEN, F. J. aiNet: An Artificial Immune Network for Data Analysis. In: ABBASS, Hussein A.; SARKER, Ruhul A.; NEWTON, Charles S. (Eds.). *Data Mining*: A Heuristic Approach. Idea Group Publishing, USA, 2001, Chapter XII, pp. 231-259.

DE CASTRO, L. N.; VON ZUBEN, F. J. Learning and Optimization Using the Clonal Selection Principle. *IEEE Transactions on Evolutionary Computation*, Special Issue on Artificial Immune Systems, v. 6, n. 3, pp. 239-251, 2002.

ESTER, M. et al. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In: KDD-96. *Proceedings of KDD-96*, 1996, pp. 226-231.

EVERITT, B. S. *Cluster Analysis*. Edward Arnold, Ltd., London, UK, 1993.

FAYYAD, U. M.; SHAPIRO, G. P.; SMYTH, P. *From Data Mining to Knowledge Discovery*: An Overview. Editors, MIT Press, p. 1-37, 1996.

FOGEL, D. B. Evolutionary Computing. *IEEE Sprectrum*, 2000, pp. 26-32.

FORSDYKE, D. R. The origins of the clonal selection theory of immunity as a case study for evaluation in science, *The FASEB journal*. Department of Biochemistiy, Queen's University, Kingston, v. 9, 1995.

FREITAS, A. A. *Data Mining and Knowledge Discovery with Evolutionary Algorithms*. Berlin: Springer-Verlag, 2002. 264 p.

HAN, J.; KAMBER, M. *Data Mining*: Concepts and Techniques, Morgan Kaufman, 2000.

HAND, D.; MANNILA, H.; SMYTH, P. *Principles of Data Mining*. The MIT Press, London: 2001. 546 p.

HAYKIN, S. *Neural Networks*: A Comprehensive Foundation, Prentice Hall, 2a Edição, 1999.

HOFMEYR, S. A.; FORREST, S. Architecture for an Artificial Immune System. Massachusetts Institute of Technology, *Evolutionary Computation* 7, v. 1, pp. 45-68, 1999.

HRUSCHKA, E. R.; CAMPELLO, R. J. G. B.; DE CASTRO, L. N. Improving the Efficiency of a Clustering Genetic Algorithm. In: C. Lemaître, C.A. Reyes, and J.A. Gonzalez (Eds.): *IBERAMIA* 2004, LNAI 3315, Springer-Verlag Berlin Heidelberg 2004, pp. 861–870, 2004.

HRUSCHKA, E. R.; CAMPELLO, R. J. G. B.; DE CASTRO, L. N. Evolving clusters in gene-expression data. *Information Sciences 176*, pp. 1898–1927, 2006.

HRUSCHKA, E. R. et al. A Survey of Evolutionary Algorithms for Clustering. *IEEE Transactions on Systems, Man, and Cybernetics* - Part C: Applications and Reviews, 2009.

HUNT, J. E.; COOKE, D. E. Learning using an artificial immune system. *Journal of Network and Computer Applications 19*, pp. 189–212, 1996.

JAIN, A. K.; MURTY, M. N.; FLYNN, P. J. Data Clustering: A Review. *ACM Computing Surveys*, v. 31, n. 3, 1999.

JERNE, N. K. Towards a network theory of the immune system. *Ann Immunol* (Inst. Pasteur) 125C, pp 373–389, 1974.

KANUNGO, T. et al. An efficient k-means clustering algorithm: analysis and implementation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 24, n. 7, Pp. 881 – 892, 2002.

KAUFMAN, L.; ROUSSEEUW, P. J. *Finding groups in data*: an introduction to cluster analysis. New York: John Wiley & Sons, 1990.

KUNCHEVA, L. I. Combining Pattern Classifiers. *Combing Clustering Results*, Sec. 8.3, Wiley, pp. 251-264, 2004.

KUNCHEVA, L.; HADJITODOROV, S. Using Diversity in Cluster Ensembles. *IEEE International Conference on Systems, Man and Cybernetics*, 2004.

MALOOF, M. A. *Machine Learning and Data Mining for Computer Security*: Methods and Applications. Springer-Verlag. London: 2006.

MILLIGAN, G. W.; COOPER, M. C. A study of the comparability of external criteria for hierarchical cluster analysis. *Multivariate Behavioral Research*, 21, pp. 441–458, 1986.

NALDI, M. C. *Investigação de técnicas de computação evolutiva em problemas de agrupamento de dados*. 2008. Qualificação (Doutorado em Ciências da Computação e Matemática Computacional), Universidade de São Paulo, São Carlos, 2008.

NEWMAN, D. J. et al. Uci machine learning repository. 1998 Disponível em: <http://archive.ics.uci.edu/ml/>. Acesso em 18 out. de 2009.

NG, T. R.; HAN, J. Efficient and Effective Clustering Methods for Spatial Data Mining. In Conference VLDB. *Proceedings of 20th VLDB Conference*, Santigo, Chile, 1994.

TIMMIS, J.; EDMONDS, C. A comment on opt-aiNet: an immune network algorithm for optimisation. *Genetic and Evolutionary Computation*, Springer, pp. 308-317, 2004.

VARELA, F. J.; COUTINHO, A. Second Generation Immune Networks, *Immunology Today*, 12(5), pp. 159-166, 1991.

VIZINE, A. L. et al. Towards Improving Clustering Ants: An Adaptive Ant Clustering Algorithm. *Informatica* 29, pp. 143–154, 2005.

WITTEN, I. H.; FRANK, E. *Data Mining*: Practical Machine Learning Tools and Techniques. 2a Edição. Elsevier Inc., 2005.

YEUNG, K. Y.; MEDVEDOVIC, M.; BUMGARNER, R.E. Clustering gene-expression data with repeated measurements, *Genome Biology*, v.4, n. 5, article R34, 2003.

YEUNG, K. Y.; RUZZO, W. Details of the adjusted rand index and clustering algorithms. Supplement to the paper "an experimental study on principal component analysis for clustering gene expression data". *Bioinformatics* (17), pp. 763–774, 2001.

ZHAO Y.; KARYPIS, G. Hierarchical Clustering Algorithms for Document Datasets. *Data Mining and Knowledge Discovery*, Springer Science + Business Media, Inc., v. 10, pp. 141–168, 2005.