UNIVERSIDADE FEDERAL DE ITAJUBÁ

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA E TECNOLOGIA DA COMPUTAÇÃO

DISCRETIZAÇÃO DE ATRIBUTOS CONTÍNUOS EM SISTEMAS DE INFORMAÇÃO UTILIZANDO ALGORITMOS GENÉTICOS PARA A APLICAÇÃO DA TEORIA DOS CONJUNTOS APROXIMADOS

Marcos Alberto de Carvalho

Itajubá, setembro de 2010

UNIVERSIDADE FEDERAL DE ITAJUBÁ

Programa de Pós-Graduação em Ciência e Tecnologia da Computação

Marcos Alberto de Carvalho

DISCRETIZAÇÃO DE ATRIBUTOS CONTÍNUOS EM SISTEMAS DE INFORMAÇÃO UTILIZANDO ALGORITMOS GENÉTICOS PARA A APLICAÇÃO DA TEORIA DOS CONJUNTOS APROXIMADOS

Dissertação submetida ao Programa de Pós-Graduação em Ciência e Tecnologia da Computação como parte dos requisitos para a obtenção do Título de Mestre em Ciências em Ciência e Tecnologia da Computação.

Área de concentração: Matemática da Computação

Orientador: Carlos Henrique Valério de Moraes

Setembro de 2010
Itajubá - MG

AGRADECIMENTOS

A Deus que conduz os nossos passos com sabedoria e amor quando somos capazes de a Ele entregar a nossa vida.

Ao Prof. Carlos Henrique Valério de Moraes pela orientação, dedicação, incentivo e confiança que depositou em mim. O tenho não somente como orientador, mas como amigo.

Ao meu amigo e mestre Prof. Edmilson Marmo Moreira que foi o principal responsável por eu ter a oportunidade de realizar esse mestrado.

Aos professores e colegas que contribuíram, direta ou indiretamente, em vários níveis com esse trabalho.

A professora Jane e minha sobrinha Vanessa pela grande ajuda com a língua portuguesa.

A minha querida esposa Jaqueline e aos meus filhos, João Paulo, Germano e Letícia pelos momentos em que abdicaram de minha presença para que esse trabalho pudesse ser completado. Amo vocês.

SUMÁRIO

LISTA DE TABELAS

LISTA DE FIGURAS

RESUMO

A Teoria dos Conjuntos Aproximados (TCA) trata, basicamente, da extração de conhecimento em bases de dados e torna possível a elaboração de modelos que auxiliem na tomada de decisões. O problema é que essas bases de dados muitas vezes possuem atributos contínuos sendo necessária a discretização desses antes da aplicação das técnicas da TCA. Os métodos clássicos realizam discretizações sem considerar a relação entre os atributos. Isso pode impedir a discretização consistente dificultando a identificação de informações supérfluas e impossibilitando a redução do volume de informações, como também pode inibir a obtenção de regras. Este trabalho apresenta um método de discretização de atributos contínuos utilizando Algoritmos Genéticos (AG). Os AG´s são utilizados para determinar os pontos de corte de cada atributo e obter uma discretização consistente. Nenhum método determinístico é utilizado para gerar pontos de corte para a população inicial, bem como qualquer outro tipo de pré-processamento do Sistema de Informação não discretizado. A população inicial é gerada aleatoriamente e, após gerações, evolui para soluções válidas. Os resultados obtidos mostram a efetividade do método de discretização proposto na aplicação das técnicas da TCA quando comparados com outros métodos de discretização.

ABSTRACT

The Rough Sets Theory broaches basically the knowledge extraction in databases and makes possible the development of models to aid decision making. The problem is that such databases often have continuous attributes is necessary to discretize these before applying the techniques of Rough Sets. The classical methods perform discretization without considering the relation between attributes. This can prevent the discretization consistent hindering the identification of superfluous information and making it impossible to reduce the volume of information, such can also inhibit the acquisition of rules. This work presents a method of discretization of continuous attributes using genetic algorithms (GA). The GA's are used to determine the cut points for each attribute and obtain a consistent discretization. Any deterministic method is used to generate cut points for initial population, as well as any other pre-processing information system not discretized. The initial population is randomly generated and, after generations, evolved into valid solutions. The results show the effectiveness of the discretization method proposed in applying of techniques of Rough Sets when compared to other methods of discretization.

Capítulo 1

INTRODUÇÃO

A tecnologia da informação tornou possível construir grandes bases de dados nas mais diversas áreas do conhecimento. A velocidade de acúmulo de dados é maior que a capacidade humana de processá-los. Conforme o volume de dados aumenta, a proporção dos dados que é analisada e entendida pelas pessoas diminui. Escondido entre todo esse volume de dados está o conhecimento potencialmente útil.

Diversos métodos têm sido propostos para o processamento desses dados com o objetivo de extrair conhecimento da informação contida nessas bases de dados, a fim de elaborar modelos que auxiliem na tomada de decisões gerenciais, operativas ou de diagnósticos.

A área de mineração de dados baseada na Teoria dos Conjuntos Aproximados (TCA) – introduzida por Pawlak (1982) – busca conhecimento a partir dessas bases de dados que, na abordagem da TCA, são representadas como Sistemas de Informações (Pawlak & Skowron, 2007).

Um Sistema de Informação é, nesse trabalho, percebido como uma tabela de dados, colunas, que são chamadas de atributos e linhas, que são chamadas de objetos. Os atributos são os mesmos para cada um dos objetos, mas seus valores podem diferir. Os objetos são classificados considerando um atributo de decisão o qual informa a decisão a ser tomada ou a classe a que pertence o objeto. Essas tabelas de dados podem ser obtidas com resultado de medidas, observações ou representam conhecimento de especialistas ou grupos de especialistas.

Os Sistemas de Informações gerados podem apresentar alguns problemas como:

- Informações supérfluas que não modificam o resultado com relação à classificação dos objetos.

- Falta de informação, gerando problemas durante a tomada de decisão.
- Grande volume de objetos acumulados ao longo do tempo, dificultando a classificação desses objetos para a tomada de decisões.

A TCA é uma ferramenta matemática para tratar incerteza e imprecisão e visa facilitar o entendimento dos Sistemas de Informação da seguinte forma (Carvalho, 2000) (Pawlak & Skowron, 2007):

- Transformando os valores dos atributos contínuos em faixas de valores de acordo com o número de classificações que vierem a ser necessárias.
- Reduzindo a base de conhecimento a um conjunto mínimo de atributos, eliminando a informação supérflua.
- Encontrando padrões escondidos nos dados.
- Transformando um conjunto de exemplos em um conjunto de regras de decisão.

Pode ser verificado que a Teoria de Conjuntos Aproximados tem fornecido subsídios nas investigações que procuram o desenvolvimento de sistemas lógicos e métodos dedutivos com a finalidade de melhor representar a informação incompleta (Pawlak, 1982) (Pawlak, 1991) (Pawlak & Skowron, 2007).

1.1 MOTIVAÇÃO

Em um Sistema de Informação, os atributos podem possuir valores contínuos. O problema é que a TCA não é adequada para manipular com esses atributos, pois impedem a extração de padrões de forma eficiente.

A maioria dos métodos de discretização é univariado, ou seja, considera um atributo contínuo por vez, não levando em conta a relação entre eles. Bay (2000) argumenta que o processo de discretização pode evidenciar ou esconder os padrões apresentados pelo Sistema de Informação. Para que a TCA, ou qualquer outro algoritmo de aprendizagem, sejam capazes de descobrir corretamente os relacionamentos existentes entre as variáveis do domínio, uma discretização, que considera seus efeitos sobre todos os atributos do domínio, é fundamental.

A discretização dos atributos contínuos consiste na divisão desses em intervalos, de forma que cada intervalo é mapeado para um valor discreto. De modo geral, deve-se procurar pelo mínimo de intervalos e, ao mesmo tempo, não permitir o enfraquecimento da capacidade de discernibilidade do conjunto de dados. Em outras palavras, a discretização tem que ser mínima e consistente (Dai, 2004).

1.2 OBJETIVOS

O principal objetivo desse trabalho foi a modelagem de um Algoritmo Genético capaz de realizar uma discretização simultânea de todos os atributos contínuos de um Sistema de Informação levando em consideração a relação de dependência entre esses atributos.

Outro objetivo foi a implementação de um programa que utilize o Algoritmo Genético proposto com a capacidade de realizar discretizações de Sistema de Informação com dados que se referem a informações do mundo real, considerando que esses dados já sofreram pré-processamentos como a eliminação de erros, repetições e inconsistências.

Assim, foi realizado um estudo comparativo entre os métodos de *Equal-frequency Binning* e Entropia MDL com a metodologia proposta, possibilitando obter valores relativos entre os resultados alcançados pelos métodos, seus pontos positivos e negativos. Não faz parte do escopo desse trabalho análises minuciosas entre demonstrações formais de efetividade das metodologias existentes com a proposta, ficando como sugestão esse item para trabalhos futuros.

1.3 ORGANIZAÇÃO DO TRABALHO

Esse trabalho está organizado da seguinte maneira:

Capítulo 2 – Teoria dos Conjuntos Aproximados: nesse capítulo será apresentada uma breve introdução sobre a TCA, a definição de indiscernibilidade, a aproximação de conjuntos, qualidade das aproximações e o conceito de redutos.

Capítulo 3 – Algoritmos Genéticos: nesse capítulo serão apresentados os fundamentos teóricos dos Algoritmos Genéticos e suas características gerais.

Capítulo 4 – Discretização de Atributos em Sistemas de Informação: nesse capítulo será apresentada a tarefa de discretização, sua importância e suas dificuldades. Será apresentado também alguns métodos de discretizações existentes.

Capítulo 5 – Discretização de Atributos Contínuos em Sistemas de Informação utilizando Algoritmos Genéticos: nesse capítulo será apresentado o algoritmo genético proposto para a discretização de atributos contínuos em Sistemas de Informação. É apresentado também o programa desenvolvido e suas funcionalidades.

Capítulo 6 – Experimentos: nesse capítulo são apresentados os experimentos realizados com a avaliação das discretizações obtidas.

Capitulo 7 – Conclusões: nesse capítulo são apresentados as conclusões e os possíveis desenvolvimentos futuros sugeridos para continuação ao tema desse trabalho.

Capítulo 2

TEORIA DOS CONJUNTOS APROXIMADOS

2.1 INTRODUÇÃO

Pode-se considerar a Teoria dos Conjuntos Aproximados como uma extensão da Teoria Clássica dos Conjuntos. Foi proposta por Zdzislaw Pawlak (1982) como uma nova ferramenta matemática para o tratamento de incertezas e imprecisões, tendo evoluído a partir de estudos sobre Sistemas de Informação. A Teoria dos Conjuntos Aproximados tem se mostrado útil quando aplicada a problemas do mundo real. Tem sido muito utilizada em Inteligência Artificial, com ênfase nas áreas de diagnóstico, representação de conhecimento incerto, redução do número de atributos do conjunto de treinamento, descoberta de conhecimento em base de dados, controle e planejamento (Pawlak & Skowron, 2007).

Este capítulo apresenta os conceitos básicos da TCA com o intuito de fornecer um embasamento teórico para o trabalho desenvolvido nesta pesquisa. As informações aqui apresentadas encontram-se em (Pawlak, 1982) e (Pawlak & Skowron, 2007).

2.2 SISTEMAS DE INFORMAÇÃO

Um Sistema de Informação (SI) é utilizado para representar o conhecimento. A forma mais comum para a representação dos dados na abordagem da TCA é através de um sistema de informação que contém um conjunto de objetos, sendo que cada objeto tem seus atributos. Esses atributos são os mesmos para cada um dos objetos, mas seus valores podem ser diferentes. Então, um sistema de informação pode ser representado por:

$$S = (U, A) \tag{2.1}$$

sendo U o conjunto finito de objetos, $A = C \cup \{d\}$ onde C é o conjunto composto por atributos de condição e $\{d\}$ o atributo de decisão que informa a decisão a ser tomada. O

atributo de decisão pode ser considerado como a classe ou rótulo de cada objeto.

A Tabela 2.1 apresenta um exemplo de classificação de dados de um sistema de informação, sendo $U=\{x_1,x_2,x_3,x_4,x_5,x_6\}$ e $A=\{a_1,a_2,a_3,d\}$ e sendo $C=\{a_1,a_2,a_3\}$, os atributos de condição e $\{d\}$ o atributo de decisão. Os atributos de condição fornecem o valor do atributo de decisão. Nesse caso, o atributo de decisão pode ser "Normal", "Alerta" ou "Falha".

Tabela 2.1 – Sistema de Informação

U	a_1	a_2	a_3	d
x_1	B	D	G	Normal
x_2	A	E	I	Alerta
x_3	C	E	I	Falha
x_4	B	F	H	Normal
x_5	A	E	H	Alerta
x_6	B	F	G	Normal

Na Tabela 2.2 estão representados todos os valores que os atributos do Sistema de Informação da Tabela 2.1 podem assumir. Nesse caso, os atributos são discretos possuem um conjunto de valores finitos. Por exemplo, o atributo a_1 pode assumir os valores A,B e C. Em situações reais, os Sistemas de Informações podem possuir atributos contínuos. Nesse caso, a solução é a discretização desses atributos com a divisão em intervalos, de forma que cada intervalo é mapeado para um valor discreto.

Tabela 2.2 – Valores dos atributos

	Atributos	Valores
Atributos Condicionais	a_1	A,B,C
	a_2	D,E,F
	a_3	G,H,I
Atributo de Decisão	d	Normal, Alerta, Falha

2.3 INDISCERNIBILIDADE

A indiscernibilidade existe entre dois objetos quando os valores de seus atributos são idênticos, não podendo ser diferenciados entre si. Considerando o conjunto de atributos condicionais C, para um subconjunto $B \subseteq C$ do sistema de informação S (2.1), uma relação de equivalência $IND_S(B)$ pode ser associada.

Para o Sistema de Informação da Tabela 2.1, os possíveis subconjuntos dos atributos condicionais $B \subseteq C$ são: $\{a_1\}$, $\{a_2\}$, $\{a_3\}$, $\{a_1,a_2\}$, $\{a_1,a_3\}$ e $\{a_2,a_3\}$ e $\{a_1,a_2,a_3\}$. Considerando o subconjunto $B=\{a_1\}$, os objetos $\{x_1,x_4,x_6\}$ são da mesma classe de equivalência e são indiscerníveis entre si. Assim, para alguns dos possíveis subconjuntos $B \subseteq C$, tem-se:

- $B=\{a_1,a_2\}$

  $U/IND_S(B)=\{\{ x_1\},\{ x_2, x_5\},\{ x_3\},\{ x_4, x_6\}\}$

- $B=\{a_3\}$

  $U/IND_S(B)=\{\{ x_1, x_6\},\{ x_2, x_3\},\{ x_4, x_5\}\}$

Para esses agrupamentos não foi considerado o atributo de decisão. Quando esse é considerado, torna-se possível obter a aproximação de conjuntos, bem como calcular a qualidade dessas aproximações e fazer a análise de dependência que é usada para determinar se um grupo de atributos de condições pode caracterizar os valores do atributo de decisão. Dessa forma, atributos irrelevantes podem ser identificados, baseando-se na definição de redutos. Redutos são subconjuntos de atributos capazes de manter as mesmas propriedades de representação de conhecimento quando esta é feita utilizando todos os atributos.

2.4 APROXIMAÇÃO DE CONJUNTOS

A Tabela 2.3 apresenta o agrupamento dos objetos conforme os valores do atributo de decisão. A questão é: Quais são os valores dos atributos condicionais que definem o valor do atributo de decisão $\{d\}$ como sendo $Normal$, $Alerta$ ou $Falha$? Conforme a Tabela 2.3, qualquer objeto com atributos de condição iguais aos dos objetos $\{x_1,x_4,x_6\}$ terão atributo de decisão igual a $Normal$ assim como qualquer objeto com atributos de condição iguais aos dos objetos $\{x_2,x_5\}$ terão atributo de decisão igual a $Alerta$ e qualquer objeto com atributos de condição igual ao do objeto $\{x_3\}$ terá atributo de decisão igual a $Falha$.

Tabela 2.3 – Agrupamento do atributo de decisão

U	a_1	a_2	a_3	d
x_1	B	D	G	$Normal$
x_4	B	F	H	$Normal$
x_6	B	F	G	$Normal$
x_2	A	E	I	$Alerta$
x_5	A	E	H	$Alerta$
x_3	C	E	I	$Falha$

Conforme apresentado na seção anterior, para um subconjunto de atributos $B \subseteq C$ do sistema de informação, temos uma relação de equivalência $IND_S(B)$. Considerando o conjunto X obtido através das informações dos atributos de B juntamente com suas respectivas classes, ou seja, valores definidos nos atributos de decisão, defini-se Aproximação Inferior de X em relação a B, denotado por $\underline{B}(X)$ e Aproximação Superior de X em relação a B denotado por $\overline{B}(X)$, como:

$$\underline{B}(X) = \{x \in U | U/IND_S(B) \subseteq X\} \tag{2.2}$$

e

$$\overline{B}(X) = \{x \in U | U/IND_S(B) \cap X \neq \emptyset\} \tag{2.3}$$

Os objetos da Aproximação Inferior $\underline{B}(X)$ são classificados com certeza como membros de X, utilizando o conjunto de atributos B, enquanto que os objetos da Aproximação Superior $\overline{B}(X)$ podem ser classificados como possíveis membros de X, utilizando o mesmo conjunto X. Obtém-se ainda a Região de Fronteira (ou duvidosa) de X:

$$RF(X) = \overline{B}(X) - \underline{B}(X) \tag{2.4}$$

A Região de Fronteira consiste de objetos impossíveis de serem classificados em X. Ainda é possível definir como Fora de Região de X o conjunto $U - \overline{B}(X)$, ou seja, consiste de objetos que não pertencem a X, considerando o mesmo conjunto B.

De uma maneira informal, como mostra a Figura 2.1, esses conjuntos são

aproximações de um dado conjunto, ou seja, envoltórios superiores e inferiores, gerando uma região em que se encontra o conjunto procurado.

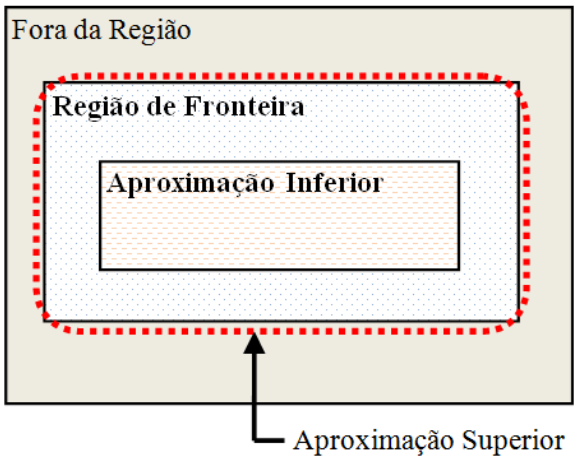


Figura 2.1 - Aproximações

A seguir, serão apresentados alguns exemplos para o subconjunto B de atributos de condição para a Tabela 2.1 e suas aproximações considerando o conjunto X obtido através das informações dos atributos de B e os respectivos valores para o atributo de decisão. Nestes exemplos verificar-se-á se os atributos em questão são capazes de definir a classe correta para todos os objetos do Sistema de Informação.

Exemplo 2.1: Considerando $B = \{a_1, a_2\}$.

A Tabela 2.4 mostra todos os objetos e os valores de seus atributos para o subconjunto de atributos $\{a_1, a_2\}$ e o respectivo valor do atributo de decisão $\{d\}$. Nessa tabela, os objetos que são indiscerníveis são agrupados na mesma linha.

Tabela 2.4 – Atributos de condição $B = \{a_1, a_2\}$ e decisão

U	a_1	a_2	d
x_1	B	D	$Normal$
x_2, x_5	A	E	$Alerta$
x_3	C	E	$Falha$
x_4, x_6	B	F	$Normal$

A partir da Tabela 2.4 o conjunto X é definido:

$$X = \{\{x_1\}, \{x_2, x_5\}, \{x_3\}, \{x_4, x_6\}\}$$

A Tabela 2.5 mostra a relação de equivalência $IND_S(B)$. Nesse caso, apenas os atributos de condição são considerados. Nessa tabela, os objetos indiscerníveis também são agrupados na mesma linha.

Tabela 2.5 – $IND_S(B)$ para $B = \{a_1, a_2\}$

U	a_1	a_2
x_1	B	D
x_2,x_5	A	E
x_3	C	E
x_4, x_6	B	F

A partir da Tabela 2.5 a relação de equivalência é definida:

$$U/IND_S(B) = \{\{x_1\}, \{x_2, x_5\}, \{x_3\}, \{x_4, x_6\}\}$$

As aproximações Inferior e Superior são respectivamente:

$$\underline{B}(X) = \{x_1, x_2, x_5, x_3, x_4, x_6\}$$

$$\overline{B}(X) = \{x_1, x_2, x_5, x_3, x_4, x_6\}$$

Todos os objetos de U fazem parte da aproximação inferior e são classificados com certeza como membros de X. Então, é possível concluir que os atributos a_1 e a_2 classificam todos os objetos do SI sem a necessidade do atributo a_3. Observa-se, nesse caso, que a Região de Fronteira (Equação 2.4) é um conjunto vazio. A Figura 2.2 mostra as aproximações.

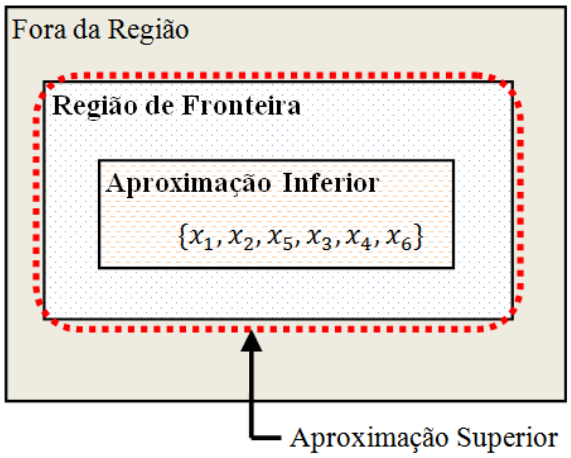


Figura 2.2 – Aproximações para $B = \{a_1, a_2\}$.

Exemplo 2.2 - Considerando $B = \{a_2, a_3\}$.

No exemplo anterior, foram obtidas as aproximações para os atributos $\{a_1, a_2\}$. Neste exemplo serão considerados os atributos $\{a_2, a_3\}$. A Tabela 2.6 mostra todos os objetos e os valores dos atributos $\{a_2, a_3\}$ e decisão $\{d\}$. Nesse caso, todos os objetos são discerníveis entre si, ou seja, são diferentes.

Tabela 2.6 – Atributos de condição $B = \{a_2, a_3\}$ e decisão

U	a_2	a_3	d
x_1	D	G	$Normal$
x_2	E	I	$Alerta$
x_3	E	I	$Falha$
x_4	F	H	$Normal$
x_5	E	H	$Alerta$
x_6	F	G	$Normal$

A partir da Tabela 2.6 o conjunto X é definido:

$$X = \{\{x_1\}, \{x_2\}, \{x_3\}, \{x_4\}, \{x_5\}, \{x_6\}\}$$

A Tabela 2.7 mostra a relação de equivalência $IND_S(B)$, onde apenas os atributos de condição são considerados. Nesse caso, ocorre o agrupamento entre os objetos $\{x_2, x_3\}$.

Tabela 2.7 – $IND_S(B)$ para $B = \{a_2, a_3\}$

U	a_2	a_3
x_1	D	G
x_2,x_3	E	I
x_4	F	H
x_5	E	H
x_6	F	G

A partir da Tabela 2.7 a relação de equivalência é definida:

$$U/IND_S(B) = \{\{x_1\}, \{x_2, x_3\}, \{x_4\}, \{x_5\}, \{x_6\}\}$$

Nesse caso, as aproximações Inferior e Superior são respectivamente:

$$\underline{B}(X) = \{x_1, x_4, x_5, x_6\}$$

$$\overline{B}(X) = \{x_1, x_2, x_3, x_4, x_5, x_6\}$$

Nem todos os objetos de U fazem parte da Aproximação Inferior. Nesse caso, o conjunto os elementos que compõem a Região de Fronteira (2.4) são:

$$RF(X) = \{x_2, x_3\}$$

Então, concluiu-se que, utilizando apenas os atributos a_2 e a_3, não é possível classificar todos os objetos do Sistema de Informação. Os objetos x_2e x_3 possuem nesse caso os mesmos valores para os atributos de condição, porém estão rotulados com classes diferentes. A Figura 2.2 mostra as aproximações.

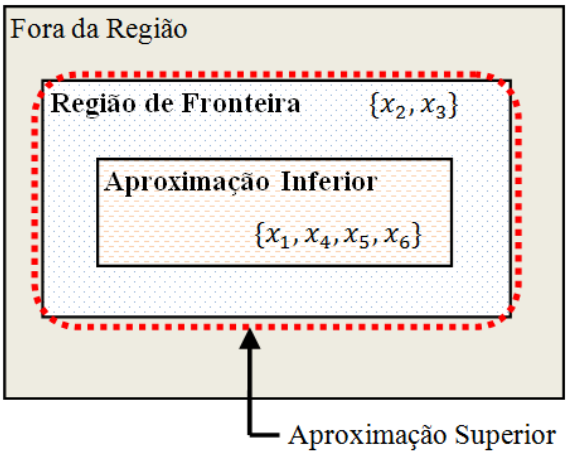


Figura 2.3 – Aproximações para $B = \{a_2, a_3\}$.

Exemplo 2.3 - Considerando $B = \{a_2, a_3\}$ e atributo de decisão $d = Normal$.

Neste exemplo serão mostradas as aproximações para uma determinada classe, no caso, a que classifica os objetos com $d = Normal$. Assim o conjunto X possui apenas os objetos dessa classe.

$$X = \{\{x_1\}, \{x_4\}, \{x_6\}\}$$

A relação de equivalência $IND_S(B)$ para $B = \{a_2, a_3\}$, já foi obtida no Exemplo 2.2, é:

$$U/IND_S(B) = \{\{x_1\}, \{x_2, x_3\}, \{x_4\}, \{x_5\}, \{x_6\}\}$$

As aproximações Inferior e Superior são respectivamente:

$$\underline{B}(X) = \{x_1, x_4, x_6\}$$

$$\overline{B}(X) = \{x_1, x_4, x_6\}$$

Nesse caso, o conjunto dos elementos que compõem a Região de Fronteira é vazio, enquanto tem-se elementos Fora de Região:

$$U - \overline{B}(X) = \{ x_2, x_3, x_5\}$$

A Figura 2.2 mostra as aproximações.

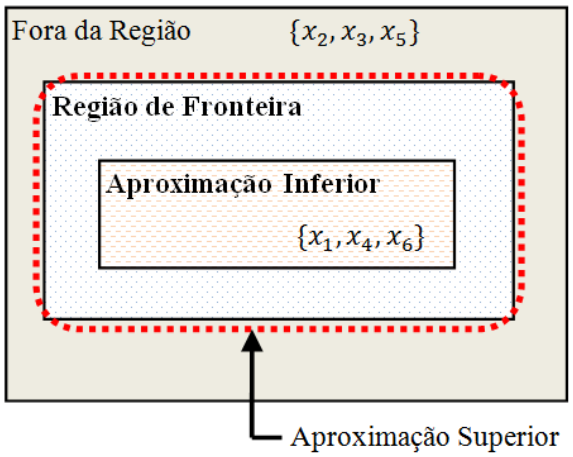


Figura 2.4 – Aproximações para $B = \{a_2, a_3\}$ e d=Normal

Observa-se, nesse caso, que na região de fronteira não existem objetos. Essa aproximação permitiu mostrar a não existência de inconsistência para a classe $d = Normal$ para os atributos a_2 e a_3. Os demais objetos que não pertencem a esta classe estão no conjunto Fora de Região.

2.5 QUALIDADE DAS APROXIMAÇÕES

A qualidade das aproximações obtidas pelas definições dadas previamente pode ser caracterizada numericamente a partir dos próprios elementos que a definem com o cálculo do **coeficiente de qualidade** das aproximações.

O coeficiente de qualidade de um conjunto X para o conjunto de atributos $B \subseteq A$, expressa por um valor situado entre 0 e 1, é definido como:

$$\gamma_B(X) = \left|\underline{B}(X)\right|/|U| \qquad (2.5)$$

onde $\left|\underline{B}(X)\right|$ denota a cardinalidade da Aproximação Inferior e $|U|$ denota a cardinalidade do conjunto de objetos do Sistema de Informação. Se $\gamma_B(X) = 1$, X é preciso em relação ao conjunto de atributos B. Caso contrário, X é não preciso em relação ao conjunto de atributos B, o que indica a existência de inconsistências na aproximação.

Considerando a aproximação obtida no Exemplo 2.1, tem-se o seguinte valor para o coeficiente de qualidade:

$$\gamma_B(X) = \frac{|\{x_1, x_2, x_5, x_3, x_4, x_6\}|}{|\{x_1, x_2, x_3, x_4, x_5, x_6\}|} = \frac{6}{6} = 1$$

O valor 1 significa que a dependência é completa. Isso mostra que a_1 e a_2 são suficientes para classificar todos os objetos que pertencem a X, preservando a relação de indiscernibilidade. O atributo a_3, nesse caso, é considerado irrelevante ou supérfluo.

Para a aproximação obtida no Exemplo 2.2, tem-se o seguinte valor para o coeficiente de qualidade:

$$\gamma_B(X) = \frac{|\{x_1, x_4, x_5, x_6\}|}{|\{x_1, x_2, x_3, x_4, x_5, x_6\}|} = \frac{4}{6} = 0{,}67$$

O coeficiente de qualidade mostrou que 0,67 (67%) de X é preciso, com respeito ao conjunto de atributos B. Isso mostra que os atributos a_2 e a_3 não são suficientes para classificar todos os objetos corretamente.

2.6 REDUTOS

Um reduto é um subconjunto de atributos de condição do SI que é capaz de classificar todos os objetos em suas respectivas classes, possuindo o mesmo poder de representação que o conjunto original de atributos.

Um reduto de C sobre um sistema de informação S é um conjunto de atributos C^*, em que $C^* \subseteq C$. Os atributos que não pertencem a C^* são então dispensáveis desde que $U/IND_S(C) = U/IND_S(C^*)$.

Os Exemplos 2.1 e 2.2 realizam aproximações onde seus coeficientes de qualidade foram obtidos na Seção 2.5. No Exemplo 2.1, os atributos a_1 e a_2 podem ser considerados um reduto do SI.

2.7 CONSIDERAÇÕES FINAIS

Os conceitos básicos da Teoria dos Conjuntos Aproximados foram apresentados com o intuito de fornecer um embasamento teórico para parte do trabalho desenvolvido nesta pesquisa.

O coeficiente de qualidade permite verificar a qualidade de uma aproximação através da relação de indiscernibilidade de um conjunto de atributos. O processo de discretização de atributos contínuos de um sistema de informação pode resultar no surgimento de inconsistências que devem ser evitadas pelo algoritmo de discretização, considerando que o Sistema de Informação original seja consistente. A TCA nesse caso pode ser utilizada para determinar se um conjunto de atributos, após serem discretizados, preserva a relação de indiscernibilidade de forma que todos os objetos do Sistema de Informação continuem sendo classificados corretamente.

Capítulo 3

ALGORITMOS GENÉTICOS

3.1 INTRODUÇÃO

Os Algoritmos Genéticos (AGs) são métodos de busca e otimização global, baseados nos mecanismos da genética e no processo evolutivo dos seres vivos. Eles empregam uma estratégia de busca paralela e estruturada, mas aleatória, que é voltada em direção ao reforço da busca de pontos de "alta aptidão", ou seja, pontos nos quais a função a ser minimizada (ou maximizada) tem valores relativamente baixos (ou altos) (Rezende, 2005).

Os princípios básicos dos AGs que serão apresentados neste capítulo foram estabelecidos por Holland (1992), podendo ser encontrados em GoldBerg (1989) e Rezende (2005). O objetivo aqui é apresentar os princípios de funcionamento dos AGs para compreensão do método de discretização proposto.

3.2 CARACTERÍSTICAS GERAIS

As técnicas de busca e otimização tradicionais iniciam-se com um único candidato que, iterativamente, é manipulado utilizando algumas heurísticas (estáticas) diretamente associadas ao problema a ser solucionado. Geralmente, estes processos heurísticos não são algorítmicos e sua simulação em computadores pode ser muito complexa. Apesar desses métodos não serem suficientemente robustos, isto não implica que eles sejam inúteis. Na prática, eles são amplamente utilizados, com sucesso, em inúmeras aplicações (GoldBerg, 1989) (Rezende, 2005).

Por outro lado, as técnicas de computação evolucionária operam sobre uma população de candidatos em paralelo. Assim, elas podem fazer a busca em diferentes áreas do espaço de solução, alocando um número de membros apropriado para a busca em várias regiões.

Os Algoritmos Genéticos (AGs) diferem dos métodos tradicionais de busca e otimização, principalmente em quatro aspectos (GoldBerg, 1989):

1. AGs trabalham com uma codificação do conjunto de parâmetros e não com os próprios parâmetros.

2. AGs trabalham com uma população e não com um único ponto.

3. AGs utilizam informações de custo ou recompensa e não derivadas ou outro conhecimento auxiliar.

4. AGs utilizam regras de transição probabilísticas e não determinísticas.

Os AGs usam uma analogia direta com o comportamento natural. Trabalham com uma população de indivíduos, cada qual representando uma solução do problema dado. A cada indivíduo associa-se um grau de aptidão, o que determina a sua capacidade de competir com os demais membros da população. Quanto maior for sua aptidão, maior a probabilidade do mesmo ser selecionado para se reproduzir, cruzando seu material genético (*crossover*) com o de outro indivíduo selecionado da mesma forma. Esse cruzamento produzirá novos indivíduos com características de seus pais. Este processo, chamado de reprodução, é repetido até que uma solução satisfatória seja encontrada. Sobre esses novos indivíduos deverá atuar ainda o operador de mutação que é necessário para a introdução e manutenção da diversidade genética da população.

A seguir é apresentado o diagrama geral do ciclo de vida de um AG:

t = 0;
Gerar População Inicial P(0);
Avaliar P(0);
enquanto** Critério de parada não for satisfeito **faça
    t = t + 1;
    Selecionar população P(t) a partir de P(t-1);
    Aplicar operadores de cruzamento e mutação sobre P(t);
    Avaliar P(t);
fim enquanto.

Uma questão fundamental para a utilização dos AGs é a forma de representação ou codificação que seja adequada ao problema. Outro ponto que é determinante na convergência do AG para uma solução do problema é a função de avaliação ou adaptação, à qual será associado um número real a cada possível solução codificada.

3.3 REPRESENTAÇÃO

Tradicionalmente, os indivíduos são representados genotipicamente por vetores binários, onde cada elemento de um vetor denota a presença (1) ou ausência (0) de uma determinada característica (Rezende, 2005). Entretanto, existem aplicações onde é mais conveniente o uso de representações por inteiros como a que será apresentada nesse trabalho.

Como exemplo será considerado um problema inspirado em GoldBerg (1989), de se encontrar o máximo da função: $f(x) = x^2$ sobre os inteiros do conjunto [0,1,..,31]. Logicamente que para se chegar ao valor ótimo, basta promover uma busca exaustiva, uma vez que o espaço de busca possui uma baixa cardinalidade. Portanto, trata-se apenas de um exemplo que permite demonstrar o algoritmo descrito anteriormente.

A Tabela 3.1 mostra uma população de quatro indivíduos que constituem a população inicial juntamente com sua respectiva aptidão ($f_i(x)$). A aptidão relativa determina a probabilidade de cada indivíduo ser selecionado para próxima geração.

Tabela 3.1 - População inicial, aptidão e aptidão relativa

I	Indivíduo x_i	$f_i(x)$ Aptidão (x^2)	Aptidão Relativa
x_0	01010	100	0,12
x_1	01110	196	0,23
x_2	10110	484	0,56
x_3	01001	81	0,09
Total		861	1,00

3.4 SELEÇÃO

O critério de seleção vai fazer com que, depois de muitas gerações, o conjunto inicial de indivíduos gere indivíduos mais aptos (Rezende, 2005). Associada uma nota ou aptidão a cada indivíduo da população, o processo de seleção escolhe então um subconjunto de indivíduos da população atual, gerando uma população intermediária. O método de seleção mais utilizado é o da Roleta, que utiliza uma roleta semelhante à roleta utilizada em jogos de azar.

A Figura 3.1 ilustra a roleta criada a partir dos valores dos indivíduos da população da Tabela 3.1.

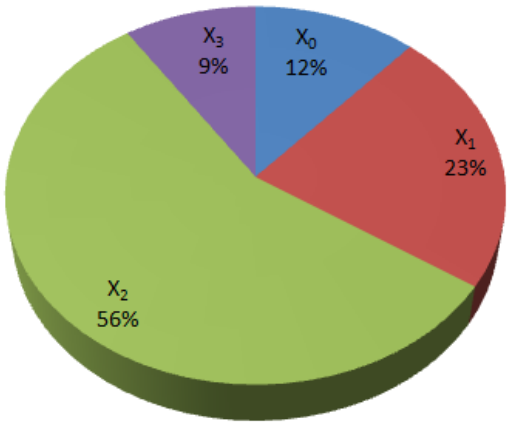


Figura 3.1 - Roleta

Nesse método, cada indivíduo da população é representado na roleta proporcionalmente ao seu índice de aptidão. Assim, aos indivíduos com alta aptidão é dada uma porção maior da roleta, enquanto aos de aptidão mais baixa é dada uma porção relativamente menor da roleta. Finalmente, a roleta é girada um determinado número de vezes, dependendo do tamanho da população, e são escolhidos, como indivíduos que participarão da próxima geração, aqueles sorteados na roleta.

3.5 OPERADORES GENÉTICOS

O princípio básico dos operadores genéticos é transformar a população através de

sucessivas gerações, estendendo a busca até chegar a um resultado satisfatório. Os operadores genéticos são necessários para que a população se diversifique e mantenha características de adaptação adquiridas pelas gerações anteriores. Estes operadores são cruzamento (*crossover*) e mutação.

O cruzamento é o operador responsável pela recombinação de características dos pais durante a reprodução, permitindo que as próximas gerações herdem essas características. Ele é considerado o operador genético predominante, por isso é aplicado com probabilidade dada pela taxa de *crossover Pc*, que deve ser maior que a taxa de mutação.

Esse operador pode, ainda, ser utilizado de várias maneiras. As mais utilizadas são (Rezende, 2005):

Um-ponto: Um ponto de cruzamento é escolhido e a partir desse ponto as informações genéticas dos pais serão trocadas. As informações anteriores a esse ponto em um dos pais são ligadas às informações posteriores a este ponto no outro pai, como é mostrado no exemplo da Figura 3.2.

Multi-pontos: É uma generalização desta idéia de troca de material genético através de pontos, onde muitos pontos de cruzamento podem ser utilizados.

Uniforme: Não utiliza pontos de cruzamento, mas determina, através de um parâmetro global, qual a probabilidade de cada variável ser trocada entre os pais.

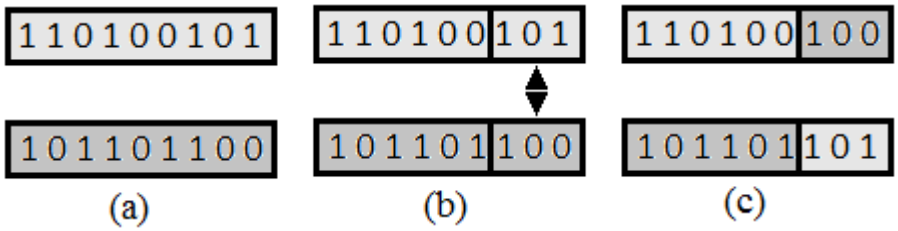


Figura 3.2 - Um exemplo de *crossover* de um ponto.
(a) dois indivíduos são escolhidos.
(b) um ponto de *crossover* é escolhido.
(c) são recombinadas as características, gerando dois novos indivíduos.

O operador de mutação é necessário para a introdução e manutenção da diversidade genética da população, alterando arbitrariamente um ou mais componentes de uma estrutura

escolhida, como é ilustrado na Figura 3.3, fornecendo assim, meios para introdução de novos elementos na população. Desta forma, a mutação assegura que a probabilidade de se chegar a qualquer ponto do espaço de busca nunca será zero, além de contornar o problema de mínimos locais, pois com este mecanismo, altera-se levemente a direção da busca. O operador de mutação é aplicado aos indivíduos com uma probabilidade dada pela taxa de mutação *Pm*; geralmente se utiliza uma taxa de mutação pequena, pois é um operador genético secundário.

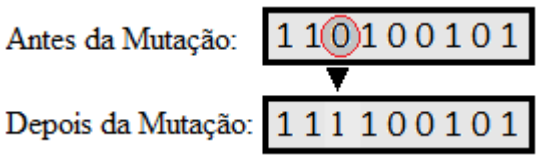


Figura 3.3 – Exemplo de mutação.

Como foi apresentado, o método da seleção pela roleta e os operadores genéticos transformam a população a cada geração. Como efeito disso, pode ocorrer que bons indivíduos desapareçam da população pela manipulação desses operadores. Isso pode ser evitado estabelecendo a reprodução elitista em que esses indivíduos são automaticamente colocados na próxima geração.

3.6 PARÂMETROS GENÉTICOS

É importante também analisar de que maneira alguns parâmetros influem no comportamento dos Algoritmos Genéticos, para que se possa estabelecê-los conforme as necessidades do problema e dos recursos disponíveis. A seguir, são discutidos alguns critérios para a escolha dos principais parâmetros:

Tamanho da População: O tamanho da população afeta o desempenho global e a eficiência dos AGs. Com uma população pequena, o desempenho pode cair, pois deste modo a população fornece uma pequena cobertura do espaço de busca do problema. Uma grande população geralmente fornece uma cobertura representativa do domínio do problema, além de prevenir convergências prematuras para soluções locais ao invés de globais. No entanto, para se trabalhar com grandes populações, são necessários maiores recursos computacionais, ou que o algoritmo trabalhe por um período de tempo muito maior.

Taxa de Cruzamento: Quanto maior for esta taxa, mais rapidamente novas estruturas serão introduzidas na população. Mas se esta for muito alta, a maior parte da população será substituída, podendo ocorrer perda de estruturas de alta aptidão. Com um valor baixo, o algoritmo pode tornar-se muito lento.

Taxa de Mutação: Uma baixa taxa de mutação evita que uma dada posição fique estagnada em um valor, além de possibilitar que se chegue em qualquer ponto do espaço de busca. Com uma taxa muito alta a busca se torna essencialmente aleatória.

Critério de Parada: Diferentes critérios podem ser utilizados para terminar a execução de um Algoritmo Genético, por exemplo, após um dado número de gerações, quando a aptidão média ou do melhor indivíduo não melhorar mais ou quando as aptidões dos indivíduos de uma população tornarem-se muito parecidas. Ao conhecer a resposta máxima da função de aptidão é possível utilizar esse valor como critério de parada.

3.7 CONSIDERAÇÕES FINAIS

Nesse capítulo, foram apresentados os princípios básicos dos AGs que são utilizados nesse trabalho para realizar uma busca cujo objetivo é encontrar pontos de corte para os atributos contínuos de um SI, fazer a discretização desses atributos e gerar um número mínimo, preferencialmente nulo, de inconsistências. Os AGs são então implementados aqui como sendo o algoritmo de discretização.

O próximo capítulo apresenta alguns métodos tradicionais de discretização. Alguns desses métodos serão utilizados para avaliação de resultados do método proposto a partir de um conjunto de características dos dados discretizados por estes métodos.

Capítulo 4

DISCRETIZAÇÃO DE ATRIBUTOS EM SISTEMAS DE INFORMAÇÃO

4.1 INTRODUÇÃO

No Capítulo 2, foram apresentados os aspectos principais da TCA e a definição de Sistema de Informação com atributos discretos e como a TCA trata esses atributos que possuem um conjunto de valores finitos. Se os atributos de entrada possuírem valores contínuos como, por exemplo, altura, peso ou idade, o conjunto de valores possíveis para esses atributos geralmente serão infinitos. A discretização é então essencial no pré-processamento de dados para a aplicação da TCA que não é adequada para manipular com atributos contínuos.

A discretização de atributos contínuos consiste na tarefa de transformar esses valores em um número reduzido de intervalos. Para isso, deve ser decidido quais serão os *pontos de corte*, que são os limiares entre um intervalo e outro. Seja o exemplo de uma temperatura que pode assumir qualquer valor entre 10 e 50 ºC. Essa temperatura pode ser discretizada em frio, normal ou quente. Para isso, deve ser decidido quais serão os pontos de corte. Nesse exemplo, o ponto de corte entre os intervalos frio e normal poderia ser igual a 20 ºC. Essa maneira manual de discretização, embora muito utilizada, torna-se praticamente inviável quando o número de atributos a serem discretizados é elevado ou não se tem conhecimento do domínio dos dados. É importante ressaltar que qualquer algoritmo de extração de padrões, que trabalha com dados numéricos, pode se beneficiar quando os atributos numéricos são discretizados previamente (Bay, 2000) (Voltolini, 2006).

Russell & Norvig (2004) consideram que encontrar bons pontos de corte é, de longe, a parte mais dispendiosa das aplicações onde a discretização é necessária. Dai (2004) caracteriza a discretização mínima e consistente como um problema NP-completo.

Nesse capítulo, será apresentada a classificação e alguns métodos tradicionais de discretização. Alguns deles serão utilizados para avaliação de resultados do método proposto

nesse trabalho a partir de um conjunto de características dos dados discretizados por esses métodos. As informações aqui apresentadas encontram-se em Kerber (1992), Dougherty (1995), Liu et al. (2002) e Voltoni (2006).

4.2 CLASSIFICAÇÃO DOS MÉTODOS DE DISCRETIZAÇÃO

Os métodos de discretização podem ser subdivididos em:

Estático ou dinâmico: Esta classificação da discretização refere-se ao momento em que a discretização é realizada. A discretização estática realiza a discretização antes da extração de padrões do Sistema de Informações. Na discretização dinâmica, a discretização ocorre ao mesmo tempo em que os padrões são obtidos.

***Top-down* ou *bottom-up*:** Os métodos *top-down* iniciam a discretização com uma lista vazia de pontos de corte e, durante a discretização, novos pontos são inseridos dividindo os valores em intervalos menores. Já os métodos *bottom-up*, no início da discretização, cada valor do atributo contínuo pertence a um intervalo distinto e esses intervalos são agrupados sucessivamente de acordo com algum critério.

Direto ou incremental: Os métodos de discretização diretos dividem os valores do atributo em um número de intervalos previamente definido pelo usuário, enquanto que, em métodos incrementais, o número de intervalos é definido por um processo de otimização, o qual necessita de um critério de parada previamente determinado.

Univariado ou multivariado: Métodos univariados consideram um atributo contínuo por vez, não levando em conta a relação entre os atributos. Já os métodos multivariados consideram, simultaneamente, múltiplos atributos para serem discretizados e levam em consideração a relação de dependência entre eles.

Supervisionado ou não-supervisionado: A discretização é chamada supervisionada quando leva em consideração o atributo classe, podendo ser tipo univariado ou multivariado. Na discretização não-supervisionada, o atributo contínuo é discretizado desprezando a relação com o atributo classe ou com qualquer outro atributo do SI.

4.3 REVISÃO DE ALGUNS MÉTODOS DE DISCRETIZAÇÃO

Ao longo dos anos, muitos algoritmos de discretização foram propostos e testados com o intuito de mostrar as vantagens da discretização em diversas áreas. Estes são alguns dos métodos apresentados pela comunidade científica:

***Equal-with Binning*:** Este método é não-supervisionado e o mais simples de discretização. Apenas divide o espaço dos valores observados em intervalos de igual tamanho.

***Equal-frequency Binning*:** Este método pode produzir intervalos de tamanhos diferentes, porém cada intervalo deve conter, aproximadamente, o mesmo número de exemplos. Nesse método, o número de intervalos a serem criados deve ser previamente definido. Esse é também um método não-supervisionado e univariado.

Entropia MDL: Este método é supervisionado e univariado apresentado por Fayyad e Irani (1993). O método de discretização Entropia MDL é baseado numa heurística de entropia mínima. Este não requer nenhum parâmetro, a discretização ocorre automaticamente sem intervenção humana. A entropia é medida em bits e pode ser definida como (Fayyad e Irani, 1993) (Voltoline, 2006): Dado um conjunto de exemplos S, contendo exemplos rotulados com as classes positiva e negativa, a entropia de S relativa a essa classificação booleana é:

$$Entropia(S) = -p_{\oplus} log_2 p_{\oplus} - p_{\ominus} log_2 p_{\ominus} \quad (4.1)$$

onde $p_{\oplus}$ é a proporção de exemplos positivos em S e $p_{\ominus}$ é a proporção de exemplos negativos em S.

A proporção de exemplos p_{C_y}é dada pela razão entre o número de exemplos da classe C_y e o número total de exemplos. Caso o número de classes seja maior que dois, a Equação 4.1 pode ser generalizada, resultando na Equação 4.2.

$$Entropia(S) = \sum_{i=1}^{N_{Cl}} -p_i log_2 p_i \quad (4.2)$$

onde N_{Cl} é o número de classes distintas do conjunto S.

É possível observar que o valor de entropia tem valor mínimo de 0 bits quando a proporção de uma das classes for 1. Nesse caso, todos os exemplos pertencem a mesma classe.

Na tarefa de discretização, inicialmente todos os valores do atributo X_i a ser discretizado pertencem a um mesmo intervalo. Todos os possíveis pontos de corte, dados pelo valor médio entre cada par de valores adjacentes do atributo, são testados. Esse teste calcula, para cada ponto de corte pc_j, a quantidade de informação necessária esperada para a classificação dos exemplos, considerando somente o atributo X, discretizado pelo ponto de corte pc_j. Essa quantidade de informação é calculada por meio da fórmula a seguir.

$$Info_i(pc_j) = \frac{size(S_{j1})}{size(X_i)} Entropia(S_{j1}) + \frac{size(S_{j2})}{size(X_i)} Entropia(S_{j2}) \quad (4.3)$$

onde S_{j1} e S_{j2} são dois conjuntos de exemplos, tal que S_{j1} é composto pelos exemplos cujos valores de X_i são menores ou iguais a pc_j e S_{j2} é composto pelos exemplos cujos valores de X_i são maiores que pc_j. A função $size$ retorna o número de exemplos pertencentes a uma coleção ou atributo.

O melhor ponto de corte pc_j a ser selecionado é aquele que minimiza o valor de $Info$. O menor valor que pode ser obtido é zero e ocorre quando o ponto de corte separa os exemplos de X_i em dois conjuntos, e todos os exemplos de cada um desses conjuntos pertencem a uma única classe.

Após o primeiro ponto de corte ser selecionado, o processo é aplicado recursivamente para cada um dos conjuntos de exemplos S_{j1} e S_{j2}, de maneira a encontrar novos pontos de corte caso necessário. O critério de parada utilizado baseia-se no Princípio da Descrição de Menor Tamanho – MDLP (*Minimum Description Length Principle*) – e compara se o ganho de informação obtido com a criação de um novo ponto de corte compensa a criação de um novo intervalo.

4.4 AVALIAÇÃO DA DISCRETIZAÇÃO

Para avaliar os diferentes métodos de discretização, é necessário que seja definido um conjunto de características dos dados discretizados comuns a todos esses métodos. São

consideradas aqui três características principais que podem ser utilizadas para avaliar os métodos de discretização. Essas características são utilizadas para avaliar os métodos selecionados que são comparados com o proposto nesse trabalho.

Número de valores distintos: A discretização visa definir intervalos para os atributos contínuos que serão mapeados para valores discretos. Quando os intervalos são maiores, tem-se consequentemente um número menor de valores discretos. Quanto menor o número de valores discretos, mais simples é a representação do atributo. Porém, existe um limite dessa generalização tal que seu aumento causaria impacto no número de inconsistências no Sistema de Informação e uma excessiva perda de informação. A discretização não deve permitir o enfraquecimento da capacidade de discernibilidade do conjunto de dados do Sistema de Informação.

Número de inconsistências na Base de Dados: Dois exemplos são ditos inconsistentes caso seus atributos possuam os mesmos valores, porém estão rotulados com classes diferentes. Inconsistências podem ser causadas pelos pontos de corte definidos pelo método de discretização utilizado. Seja o exemplo dado na Tabela 4.1, que apresenta um SI contendo 8 objetos, com os atributos de condição $\{a_1,a_2\}$ e de decisão $\{d\}$ correspondente a classe desses objetos. Nessa tabela os atributos de condição são contínuos.

Tabela 4.1 – SI com atributos contínuos.

U	a_1	a_2	d
x_1	2,3	10,1	1
x_2	8,7	4,9	2
x_3	7,1	8,3	0
x_4	9,4	0,8	2
x_5	6,1	7,5	0
x_6	1,8	9,4	1
x_7	0,3	3,6	0
x_8	10,4	6,2	2

Considere os seguintes pontos de corte, definidos manualmente, para os atributos a_1 e a_2:

$$se\ a_1 < 5{,}0 => a_1 = A$$
$$se\ a_1 \geq 5{,}0 => a_1 = B$$
$$se\ a_2 < 5{,}0 => a_2 = A$$
$$se\ a_2 \geq 5{,}0 => a_2 = B$$

A Tabela 4.2 apresenta os atributos discretizados de acordo com as faixas definidas.

Tabela 4.2 – SI com atributos discretizados.

U	a_1	a_2	d
x_1	A	B	1
x_2	B	A	2
x_3	B	B	0
x_4	B	A	2
x_5	B	B	0
x_6	A	B	1
x_7	A	A	0
x_8	B	B	2

A discretização causou inconsistências (ou contradições) entre os objetos x_3, x_5 e x_8, conforme é mostrado pelas linhas em destaque na Tabela 4.2. Qualquer método de discretização pode causar inconsistências que devem ser identificadas e podem assim avaliar a qualidade da discretização.

Precisão da predição: Após a discretização de um SI, novos exemplos devem ser apresentados com o objetivo de predizer a respectiva classe. O ideal é que a classificação seja melhor ou sem diferença estatisticamente significante em relação ao classificador induzido com os exemplos do SI original.

4.5 CONSIDERAÇÕES FINAIS

Os métodos de discretização apresentados são todos univariados. Realizam a discretização considerando um atributo contínuo por vez. Os métodos mais simples como o *Equal-with Binning* e *Equal-frequency Binning*, além de serem univariados são também não-supervisionados. Resultados com a utilização desses métodos podem ser encontrados em Dougherty (1995), Liu et al. (2002) e Voltoni (2006).

O método de discretização proposto nesse trabalho é estático, multivariado e supervisionado. Isso significa que a discretização ocorre antes da extração de padrões, é feita considerando simultaneamente todos os atributos do SI. Este método é apresentado no próximo capítulo.

Capítulo 5

DISCRETIZAÇÃO DE ATRIBUTOS CONTÍNUOS EM SISTEMAS DE INFORMAÇÃO UTILIZANDO ALGORITMOS GENÉTICOS

5.1 INTRODUÇÃO

O Capítulo 4 apresentou a tarefa de discretização e alguns métodos utilizados para execução dessa tarefa. Neste capítulo, será apresentado um método de discretização de atributos contínuos utilizando Algoritmos Genéticos.

Uma proposta semelhante é apresentada por Dai (2004) que utiliza um AG em conjunto com a TCA para a tarefa de discretização. Apesar dos bons resultados obtidos, nesse caso o AG inicia a busca após a realização de um pré-processamento do Sistema de Informação. A população inicial é criada a partir do cálculo de pontos de corte candidatos para cada atributo. Para esse cálculo, todos os atributos devem ser ordenados de forma crescente. Os indivíduos foram representados por *strings* binárias de tamanho igual ao número de pontos de corte definidos.

O Algoritmo Genético proposto, que será apresentado neste capítulo, realiza uma discretização estática, supervisionada e multivariada. A busca é iniciada sem qualquer tipo de pré-processamento do Sistema de Informação, tal como ordenação crescente ou cálculo de pontos de corte candidatos. A população inicial é gerada aleatoriamente e os indivíduos são representados por números inteiros que definem os pontos de corte para cada atributo. O método de avaliação definido em conjunto com os operadores genéticos, permitem que esses indivíduos evoluam ao longo das gerações para soluções válidas.

Esse capítulo apresenta também as estruturas de dados utilizadas para representação dos indivíduos. As funções de avaliação implementadas são detalhadas passo a passo. Ao final, o programa desenvolvido é apresentado mostrando suas funcionalidades.

5.2 REPRESENTAÇÃO DE INTERVALOS

Na discretização de um atributo contínuo, após os pontos de corte serem encontrados, é necessário que seja realizada uma alteração no atributo que está sendo discretizado de maneira a transformar seus valores numéricos em valores discretos. Nesse trabalho, os atributos contínuos são substituídos por letras (A,B,C,...) as quais representam intervalos numéricos que serão definidos automaticamente pelo AG. Esses intervalos são utilizados para discretizar e classificar novos exemplos de acordo com os pontos de corte definidos.

Seja, por exemplo, o atributo *a* de um SI que tem valores contínuos variando de -10,5 a 23,8. Dados os pontos de corte -3,5, 5,5 e 15,5 será gerada a seguinte informação:

Atributo: a

*> * e <= -3,5 então faixa = A*
>-3,5 e <= 5,5 então faixa = B
>5,5 e <=15,5 então faixa = C
*>15,5 e < * então faixa = D*

O primeiro intervalo tem como limite inferior o -∞, representado pelo caractere "*". No último intervalo esse caractere representa +∞.

5.3 O ALGORITMO GENÉTICO

5.3.1 REPRESENTAÇÃO DOS INDIVÍDUOS

Para o cálculo dos pontos de cortes, o primeiro aspecto considerado é a forma de representação da população de indivíduos ou *cromossomos* que, nesse caso, serão responsáveis por armazenar informações dos pontos de corte de todos os atributos contínuos de um SI.

Tabela 5.1 – SI com atributos contínuos.

U	a_1	a_2	***d***
x_1	2,3	10,1	1
x_2	8,7	4,9	2
x_3	7,1	8,3	0
x_4	9,4	0,8	2
x_5	6,1	7,5	0
x_6	1,8	9,4	1
x_7	0,3	3,6	0
x_8	10,4	6,2	2

Seja o exemplo da Tabela 5.1 onde existem os atributos de decisão contínuos a_1 e a_2. Um exemplo da estrutura de um indivíduo da população com os pontos de corte para discretizar esses dois atributos é mostrado na Figura 5.1. Nesse caso os atributos a_1 e a_2 serão discretizados em duas faixas "A" e B". A faixa A do atributo a_1 agrupa valores de 0 a 30%, enquanto que a faixa B agrupa valores de 30 a 100%. Já para o atributo a_2 a faixa A agrupa valores de 0 a 75%, enquanto a faixa B agrupa valores de 75% a 100%.

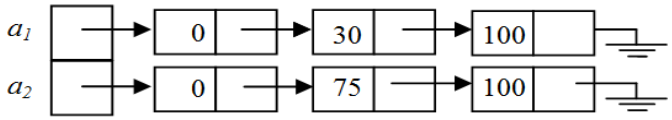


Figura 5.1 – Estrutura do indivíduo contendo pontos de corte para os atributos a_1 e a_2 .

A definição da faixa é feita com a normalização do valor contínuo de acordo com a porcentagem de corte definido para a faixa. No exemplo anterior, o menor valor do atributo a_1 é 0,3 e o maior valor 10,4. Então os valores desse atributo que forem menores ou iguais que 3,03 (30%) serão mapeados para a faixa A. Os maiores que 3,03 serão mapeados para a faixa B. O mesmo processo ocorre para o atributo a_2.

Os pontos de corte definidos na Figura 5.1 definem então as faixas a seguir para os atributos da Tabela 5.1:

Atributo: a1
*> * e <= 3,03 então faixa = A*
*>3,03 e <= * então faixa = B*
Atributo: a_2
*> * e <= 7,78 então faixa = A*
*>7,78 e <= * então faixa = B*

A estrutura de dados utilizada, para armazenar os pontos de corte de cada atributo, consiste em uma *lista ligada* ordenada de forma crescente pelo valor do corte. Cada indivíduo da população, por sua vez, é um vetor de tamanho igual ao número de atributos de condição que serão discretizados. Em cada posição desse vetor, é armazenado o endereço da lista ligada (contendo os cortes). E, finalmente, a população é também outro vetor de tamanho *I* que tem em suas posições o endereço dos indivíduos. A Figura 5.2 mostra uma abstração dessa estrutura de dados com uma população de tamanho *I* e número máximo de faixas igual a quatro. No caso são considerados os atributos a_1 e a_2 da Tabela 5.1.

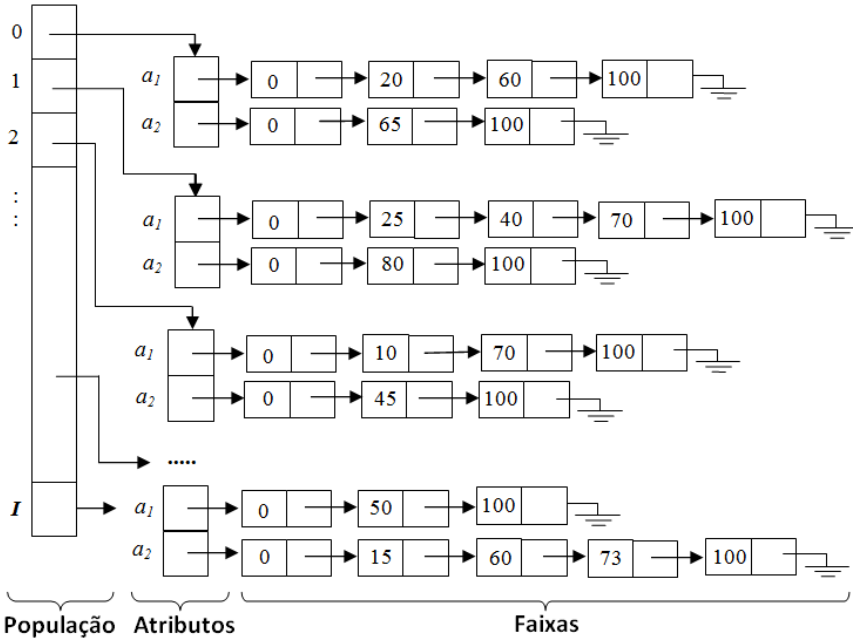


Figura 5.2 – Estrutura de dados da população.

5.3.2 POPULAÇÃO INICIAL

São duas as formas de geração da população inicial, mais especificamente em relação a definição do número de faixas (intervalos) criadas para discretizar cada atributo do SI. Para qualquer uma das formas utilizadas, é necessário definir o número máximo de faixas que cada atributo do SI poderá ter ao ser discretizado pelo AG.

5.3.2.1 Definição aleatória do número de faixas

Neste modo, fica a cargo do AG definir a quantidade de faixas (respeitando o valor máximo definido) bem como os valores de seus limites iniciais. Esses limites são definidos aleatoriamente pelo AG. A Figura 5.2 é um exemplo de uma população inicial definida aleatoriamente. No caso, para o atributo a_1 do indivíduo 0, foram definidas 3 faixas: de 0 a 20% para a faixa A, de 20 a 60% para a faixa B e de 60 a 100% para a faixa C. No entanto, para o mesmo atributo a_1 do indivíduo 1, foram definidas 4 faixas: de 0 a 25% para a faixa A, 25 a 40% para a faixa B, 40 a 70% para a faixa C e de 70 a 100% para a faixa C. Nesse caso, o número máximo de faixas permitido é igual a quatro sendo esse valor definido pelo usuário.

O AG poderá, ao longo das gerações, inserir ou remover faixas para cada atributo em função da atuação dos operadores de cruzamento e mutação.

5.3.2.2 Definição do número de faixas pelo Método de Scott

O Método de Scott foi inicialmente proposto para a definição automática do número de intervalos discretos de um histograma na definição de suas barras. Segundo Scott (1979), se o número de intervalos for muito alto, o histograma se apresentará muito acidentado. Caso o número de intervalos seja muito pequeno, o histograma se apresentará muito suave. Em ambos os casos, o histograma gerado não representará de maneira satisfatória a distribuição dos dados. Scott propõe um método automático que, a partir da análise dos dados, é definido o número de intervalos discretos a serem gerados. Voltolini (2006) utiliza esse método com sucesso na definição do número de intervalos discretos depois de avaliar outros métodos para esse fim. Esse método pode ser representado pela Equação 5.1, onde dado um atributo numérico a_i, pode ser calculado o número de intervalos a serem gerados.

$$Scott(a_i) = \frac{\max(a_i) - \min(a_i)}{3{,}5\sqrt{var(a_i)}N^{-\frac{1}{3}}} \tag{5.1}$$

onde $max(a_i)$ e $min(a_i)$ são, respectivamente, o valor máximo e mínimo do atributo a_i, $var(a_i)$ é a variância do atributo a_i e N o número de valores distintos do atributo a_i.

Esse método pode então ser utilizado pelo AG. Quando isso ocorre, cada atributo do SI terá seu número de faixas definido pela Equação 5.1, mas respeitando o limite máximo definido pelo usuário. A Figura 5.3 mostra o cálculo do número de faixas para os atributos da Tabela 5.1. O número de faixas corresponde ao valor truncado obtido pelo método de Scott mais 1 (um).

Atributos	a_1	a_2
Máximo	10,4	10,1
Mínimo	0,3	0,8
Variância	14,69	9,85
Scott(a_i)	1,51	1,69
Número de faixas	2	2

Figura 5.3 – Cálculo do número de Faixas

Quando esse método de definição do número de faixas for utilizado, o AG irá definir aleatoriamente apenas os valores dos limites de cada faixa. Com isso, seu objetivo consiste em encontrar apenas bons valores para os limites de cada faixa. Ao longo das gerações, o número de faixas para cada atributo não será alterado.

5.3.3 FUNÇÃO DE AVALIAÇÃO

O AG proposto tem como objetivo a realização de uma discretização multivariada no qual são considerados simultaneamente todos os atributos do SI. Para isso é verificada a consistência da discretização de cada objeto, levando em conta a classe a que pertencem.

Cada indivíduo da população propõe uma discretização diferente para o SI. A função de avaliação determina a aptidão de cada indivíduo. Nesse caso, é obtida em função do número de inconsistências geradas pela proposta de discretização de cada indivíduo da população.

A seguir, será mostrado um exemplo de cálculo da aptidão de um indivíduo. A Tabela 5.2 apresenta a discretização da Tabela 5.1 com base no primeiro indivíduo ($I = 0$) da população da Figura 5.2. Assim o atributo a_1 foi mapeado em três faixas (“A”, “B” e “C”) e o atributo a_2 para duas faixas (“A” e “B”).

Tabela 5.2 – Discretização da Tabela 5.1

U	a_1	a_2	d
x_1	B	B	1
x_2	C	A	2
x_3	C	B	0
x_4	C	A	2
x_5	C	B	0
x_6	A	B	1
x_7	A	A	0
x_8	C	B	2

A avaliação desse indivíduo é obtida a partir da tabela discretizada e pode ser calculada de duas formas: Pelo coeficiente de qualidade da TCA ou em função das classes majoritárias.

5.3.3.1 Avaliação pelo coeficiente de qualidade da TCA

O coeficiente de qualidade $\gamma_B(X)$, apresentado no Capítulo 2, caracteriza numericamente a qualidade de uma aproximação. Sua equação é apresentada novamente a seguir:

$$\gamma_B(X) = \left|\underline{B}(X)\right|/|U|$$

O conjunto B refere-se ao conjunto de atributos de condição. Como o objetivo é verificar todos os atributos simultaneamente, sempre serão considerados na aproximação todos os atributos de condição existentes no SI, no caso do exemplo em questão, a_1e a_2. A relação de indiscernibilidade para esse caso é então:

$$U/IND_S(B) = \{\{x_1\}, \{x_2, x_4\}, \{x_3, x_5, x_8\}, \{x_6\}, \{x_7\}\}$$

O Conjunto X é obtido através das informações dos atributos de B e os respectivos valores para o atributo de decisão. Nesse caso:

$$X = \{\{x_1\}, \{x_2, x_4\}, \{x_3, x_5\}, \{x_6\}, \{x_7\}, \{x_8\}\}$$

Os objetos da aproximação inferior $\underline{B}(X)$ são classificados, com certeza, como membros de X e é obtido pela equação:

$$\underline{B}(X) = \{x \in U | U/IND_S(B) \subseteq X\}$$

A aproximação inferior para o exemplo será então:

$$\underline{B}(X) = \{x_1, x_2, x_4, x_6, x_7\}$$

E o coeficiente de qualidade pode então ser calculado:

$$\gamma_B(X) = \frac{|\{x_1, x_2, x_4, x_6, x_7\}|}{|\{x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8\}|} = \frac{5}{8} = 0{,}625$$

O valor 0,625 é a aptidão do indivíduo. Isso significa que existem inconsistências na discretização. De fato elas ocorrem entre os objetos x_3, x_5 e x_8, considerados exemplos inconsistentes no SI discretizado por apresentarem os mesmos valores para os atributos de condição, sendo rotulados com classes diferentes.

Quando o coeficiente de qualidade para uma discretização aplicada for igual a 1, significa que não existem inconsistências nessa discretização. Esse é o valor máximo da função de avaliação que é buscado pelo AG ao longo das gerações.

Como mostrado, esse cálculo é feito a partir da tabela discretizada e foi implementado via linguagem de consulta *SQL* (*Structured Query Language*). Seus passos são detalhados a seguir:

1. Montar uma tabela com a seguinte consulta: seleção dos atributos de condição e decisão, agrupados por esses mesmos atributos criando um novo campo *CNT* para armazenar o número de objetos iguais que foram encontrados.

2. Montar uma nova tabela a partir da tabela obtida no passo 1: seleção agrupando apenas pelos atributos de condição, criando um novo campo *XCNT* para armazenar o somatório do campo *CNT* e um novo campo *NOC* para armazenar o número de objetos iguais que foram encontrados.

3. Calcular o somatório de todas as ocorrências *XCNT* onde *NOC* seja maior que 1.

O resultado da aplicação desses passos no exemplo da Tabela 5.2 são mostrados nas Tabelas 5.3 e 5.4

Tabela 5.3 – Cálculo de inconsistências – Passo 1

U	***a₁***	***a₂***	***d***	***CNT***
x_1	B	B	1	1
x_2, x_4	C	A	2	2
x_3, x_5	C	B	0	2
x_6	A	B	1	1
x_7	A	A	0	1
x_8	C	B	2	1

Tabela 5.4 – Cálculo de inconsistências – Passo 2

U	***XCNT***	***NOC***
x_1	1	1
x_2, x_4	2	1
x_3, x_5, x_8	3	2
x_6	1	1
x_7	1	1
Total	3	

O passo 3 resultará no valor do erro, nesse caso igual a 3. Esse valor é obtido pela soma de todos os valores do campo *XCNT*, somente onde o campo *NOC* é maior que 1. Quando *NOC* é maior que 1, indica que esses objetos são classificados em classes diferentes o que é interpretado, nesse caso, como inconsistências. O Anexo I apresenta a consulta *SQL* para cálculo do número de inconsistências.

5.3.3.2 Em função das classes majoritárias

A avaliação utilizando o coeficiente de qualidade considera que todos os objetos que estão na Região de Fronteira são inconsistentes, no exemplo os objetos x_3, x_5 e x_8. No entanto, essas inconsistências podem ser calculadas considerando que os objetos inconsistentes são apenas os que pertencem a classes minoritárias. No exemplo, existem dois objetos x_3 e x_5 pertencentes a mesma classe, que é então a classe majoritária, enquanto que o objeto x_8 pertence a uma classe distinta, que é então a classe minoritária. Nesse caso será considerado então como inconsistente apenas o objeto x_8.

Esse cálculo é realizado em três passos que são mostrados a seguir:

1. Montar uma tabela com a seguinte consulta: seleção dos atributos de condição e decisão, agrupados por esses mesmos atributos criando um novo campo *CNT* para armazenar o número de objetos iguais que foram encontrados.

2. Montar uma nova tabela a partir da tabela obtida no passo 1: seleção agrupando apenas pelo conjunto de atributos de condição, criando um novo campo *XCNT* para armazenar o número de objetos iguais encontrados e um novo campo *OCMAX* para armazenar o número máximo do campo *CNT* desses objetos.

3. Calcular o somatório de todas as diferenças entre total de ocorrências *XCNT* e máximo das ocorrências *OCMAX*.

O resultado da aplicação desses passos no exemplo da Tabela 5.2 são mostrados nas Tabelas 5.5, 5.6 e 5.7.

Tabela 5.5 – Cálculo de inconsistências – Passo 1

U	a_1	a_2	***d***	***CNT***
x_1	B	B	1	1
x_2, x_4	C	A	2	2
x_3, x_5	C	B	0	2
x_6	A	B	1	1
x_7	A	A	0	1
x_8	C	B	2	1

Tabela 5.6 – Cálculo de inconsistências – Passo 2

U	*XCNT*	*OCMAX*
x_1	1	1
x_2, x_4	2	2
x_3, x_5, x_8	3	2
x_6	1	1
x_7	1	1

Tabela 5.7 – Cálculo de inconsistências – Passo 3

U	*XCNT*	*OCMAX*	*Erro*
x_1	1	1	0
x_3, x_4	2	2	0
x_3, x_5, x_8	3	2	1
x_6	1	1	0
x_7	1	1	0
Total			1

O Passo 3 resultará no valor do erro, nesse caso igual a 1. Isso significa que apenas um objeto é considerado como inconsistente. O Anexo II apresenta a consulta *SQL* para cálculo do número de inconsistências.

5.3.4 CRITÉRIO DE RENOVAÇÃO E SELEÇÃO

O critério de renovação e seleção implementado nesse trabalho é o da reprodução em conjunto com o método da Roleta. Na reprodução, são copiados indivíduos para a próxima geração de acordo com sua aptidão. Indivíduos com alto valor de aptidão contribuirão com um ou mais descendentes exatamente iguais para a próxima geração.

Considere o exemplo da Tabela 5.8 que mostra uma população $I = 4$ com o valor da função de avaliação de cada indivíduo $f_1 = 0{,}01$, $f_2 = 0{,}36$, $f_3 = 0{,}46$ e $f_4 = 0{,}80$ na população corrente. A partir dessa avaliação, é calculada a avaliação relativa do indivíduo na geração dada pela Equação 5.2

$$AR_i = \frac{f_i - min(f)}{max(f) - min(f)} \tag{5.2}$$

Onde f_i é a avaliação do indivíduo i, $max(f)$ e $min(f)$ são, respectivamente, o valor máximo e mínimo das avaliações de todos os indivíduos na geração atual.

A avaliação relativa garante notas altas a indivíduos mais bem adaptados mesmo quando a nota média da população é quase a nota máxima. Indivíduos, com notas acima da média nesse caso, são os melhores avaliados com chances maiores de reprodução.

A aptidão parcial de cada cromossomo é calculada com base no somatório da avaliação relativa. No caso é igual a $0/2 = 0$, $0{,}44/2 = 0{,}22$, $0{,}56/2 = 0{,}28$, e $1/2 = 0{,}5$, respectivamente. A partir da aptidão parcial, é calculado o número de descendentes esperados de cada cromossomo na próxima geração. No exemplo, para cada cromossomo, os descendentes esperados são $0{,}0 \times 4 = 0$, $0{,}22 \times 4 = 0{,}88$, $0{,}28 \times 4 = 1{,}12$ e $0{,}5 \times 0{,}4 = 2{,}00$ respectivamente.

Tabela 5.8 – Critério de Seleção

I	$f_i(x)$ Avaliação	$AR_i(x)$	Aptidão Parcial	Descendentes Esperados	Indivíduos Reproduzidos
0	0,01	0,00	0,00	0,00	0
1	0,36	0,44	0,22	0,88	0
2	0,46	0,56	0,28	1,12	1
3	0,80	1,00	0,50	2,00	2
Max	0,80				
Min	0,01				
Soma	1,63	2	1,00	4	3

O número de indivíduos efetivamente reproduzidos na próxima geração é dado pela parte inteira do número de descendentes esperados de cada cromossomo. Logo, para o exemplo temos uma reprodução do indivíduo 2 e duas reproduções do indivíduo 3 totalizando a reprodução de três indivíduos.

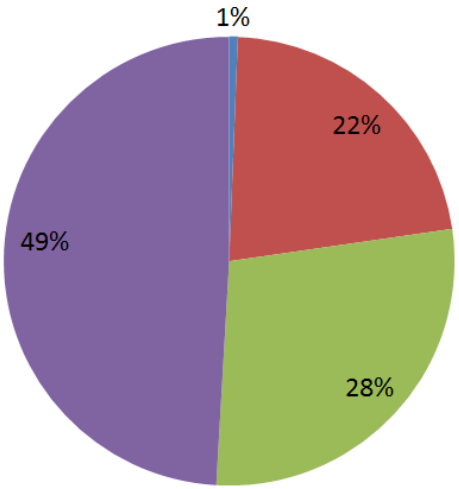


Figura 5.4 – Representação de Roleta

A seleção de mais um indivíduo para completar a população de 4 indivíduos é feita através do método da roleta onde qualquer um dos indivíduos poderá ser escolhido para a nova geração, pois todos estarão na roleta com uma partição proporcional a sua avaliação f_i. A representação da roleta é mostrada na Figura 5.4. No exemplo, para completar a população, a roleta será "girada" apenas uma vez.

5.3.5 OPERADORES GENÉTICOS

Após a avaliação de cada indivíduo, esses são submetidos aos operadores genéticos de cruzamento (*crossover*) e mutação. O princípio básico dos operadores genéticos é transformar a população por meio de sucessivas gerações, estendendo a busca até chegar a um resultado satisfatório.

5.3.5.1 Cruzamento

O operador de cruzamento toma dois indivíduos "pais" selecionados e secciona os seus cromossomos em uma posição escolhida aleatoriamente. Os genes são trocados de acordo com o ponto de corte escolhido gerando dois descendentes. Ambos os descendentes herdam características de seus pais. O operador de cruzamento é aplicado com uma probabilidade de cruzamento Pc ($0 \leq Pc \leq 1$).

No AG proposto, o cruzamento pode ocorrer de duas formas. Se a população inicial foi criada utilizando a definição aleatória para o número de faixas, a população terá indivíduos com estruturas e tamanho diferentes. Se a população inicial foi criada utilizando o método de Scott, todos os indivíduos terão o mesmo tamanho. Foi desenvolvido então um operador de cruzamento para cada caso.

Cruzamento com número de faixas aleatórias para cada indivíduo

A Figura 5.5 mostra dois indivíduos emparelhados antes do cruzamento. Como seus tamanhos são diferentes, o cruzamento ocorre a partir da seleção aleatória de um ponto de corte que corresponde a um índice no vetor de atributos. Nesse caso, é indicado pela linha tracejada no indivíduo 0 e 1 conforme mostra a Figura 5.5. Durante o cruzamento, os

atributos a partir desse ponto são trocados e, depois do cruzamento, dois novos indivíduos são formados.

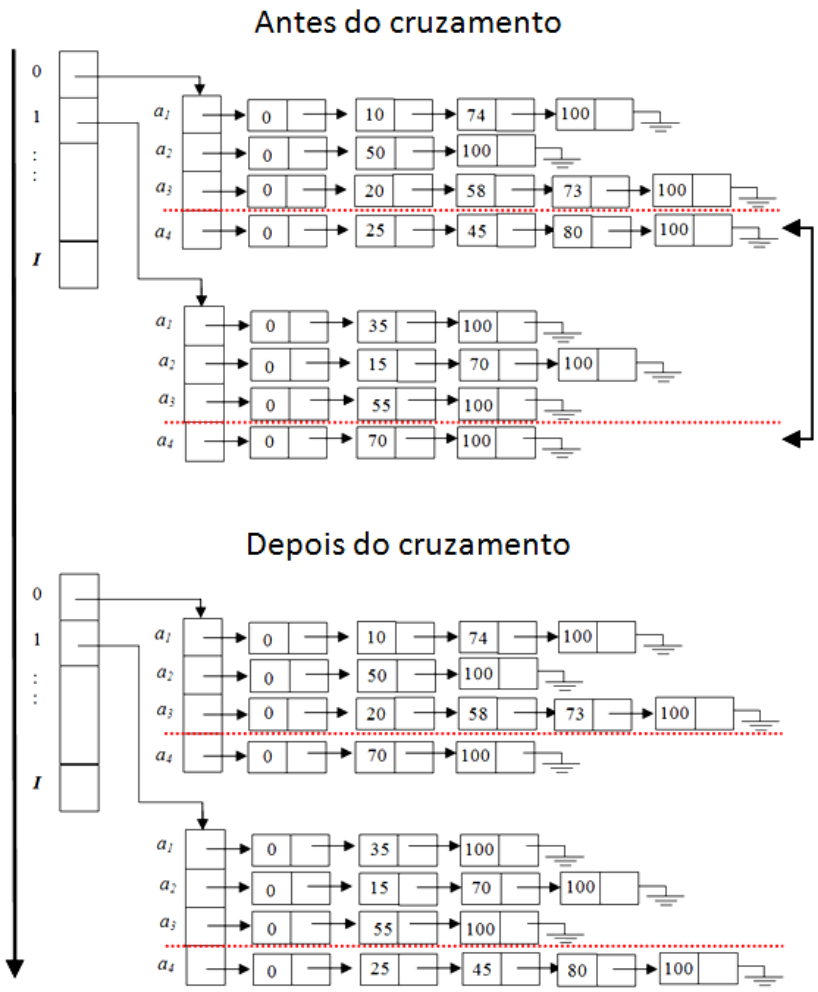


Figura 5.5 – Cruzamento de atributos entre indivíduos com estruturas diferentes

Cruzamento com número de faixas determinadas pelo método de Scott

Quando uma população é criada utilizando o método de Scott, o número de faixas é o mesmo para todos os indivíduos da população. Ao longo das gerações, o número de faixas não é alterado pelos operadores de cruzamento e mutação. Assim, o cruzamento pode ocorrer a um nível mais "baixo", cruzando entre os indivíduos pais seus atributos, um a um.

A Figura 5.6 mostra dois indivíduos emparelhados antes do cruzamento. A seleção

aleatória de um ponto de corte ocorre para cada atributo conforme indicado pelas linhas tracejadas nos indivíduos 0 e 1. Durante o cruzamento, as faixas a partir desse ponto são trocadas entre os atributos. Depois do cruzamento, dois novos indivíduos são formados.

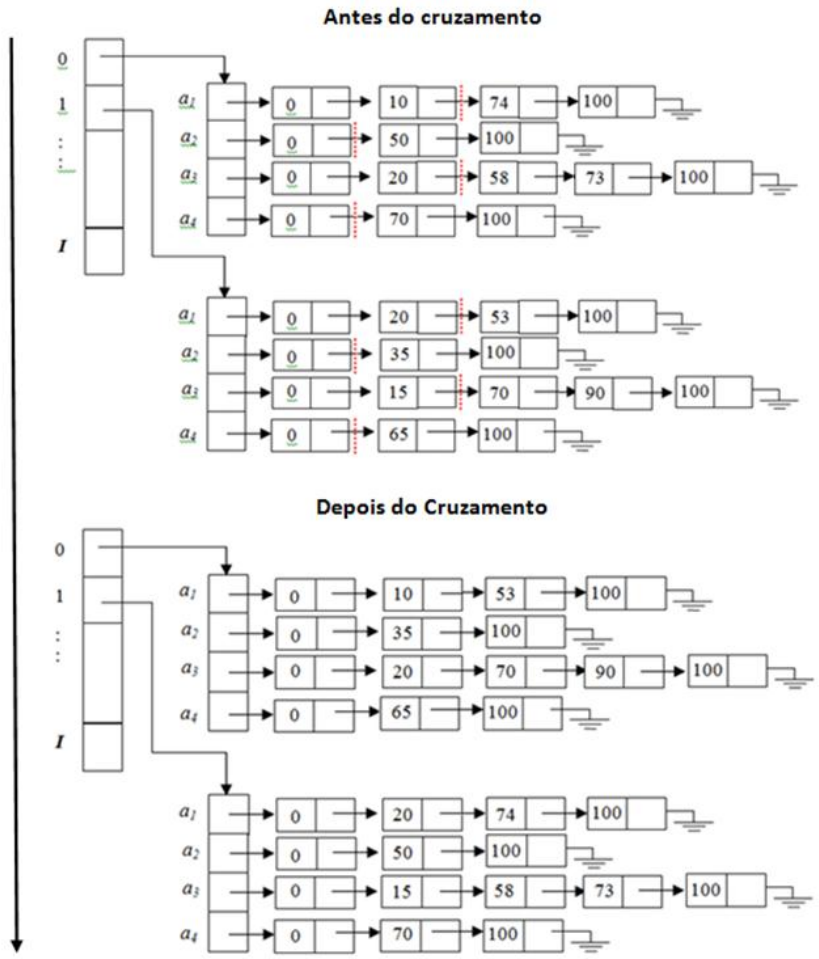


Figura 5.6 – Cruzamento de faixas entre indivíduos com estruturas iguais

Esse tipo de cruzamento não pode ser aplicado para indivíduos com estruturas diferentes, uma vez que o ponto de corte entre dois indivíduos é o mesmo. Poderia ocorrer, por exemplo, o emparelhamento entre dois indivíduos que possuem 5 e 2 faixas definidas para um determinado atributo. Não haveria como realizar o cruzamento se o ponto de corte selecionado for, por exemplo, o 3. O segundo indivíduo não teria faixas definidas para esse ponto de corte.

5.3.5.2 Mutação

Este operador é necessário para a introdução e manutenção da diversidade genética da população. Isso é feito alterando os limites das faixas de um atributo escolhido. Quando a população trabalha com indivíduos com estruturas diferentes, a mutação fornece meios para alterar, introduzir ou remover novos pontos de corte para os atributos. O operador de mutação é aplicado com uma probabilidade dada pela taxa de mutação P_m ($0 \leq P_m \leq 1$).

A Figura 5.7 mostra um exemplo de mutação ocorrida nos atributos a_1 e a_3.

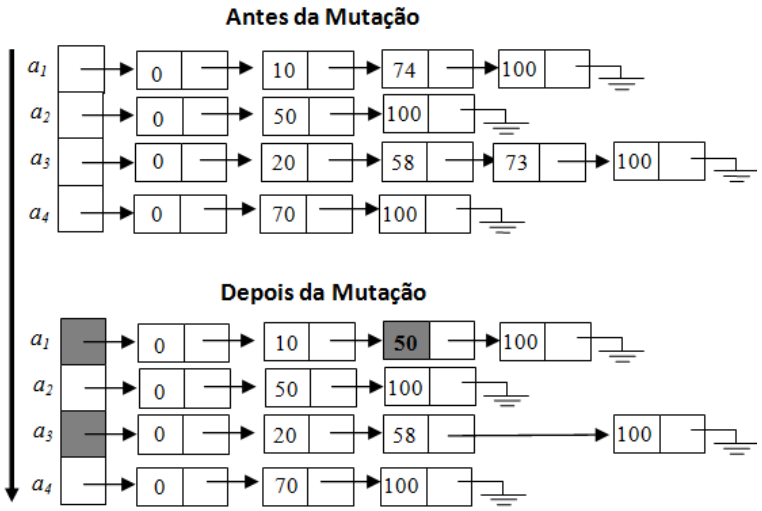


Figura 5.7 – Exemplo de Mutação

Para o caso de populações com o número de faixas definidas pelo método de Scott, ao longo das gerações, este operador de mutação irá promover somente alterações nos valores das faixas, não realizando remoções ou inserções de novas faixas.

5.4 PROGRAMA DESENVOLVIDO

Para a implementação do AG apresentados nas seções anteriores, foi utilizada a linguagem de programação Delphi® em conjunto com o banco de dados Microsoft Access®. A seguir serão mostradas as principais funcionalidades do programa sendo utilizado para discretização do SI exemplo, apresentado na Tabela 5.1.

Todas as funções do programa estão organizadas dentro de quatro “abas”. São elas:

Definição de Tabelas e Atributos: Nesta aba, o usuário seleciona um SI para ser discretizado (Figura 5.8), define quais são os atributos de condição e decisão e, dos atributos de condição, quais serão discretizados. Os objetos do SI selecionado são exibidos na parte inferior da janela.

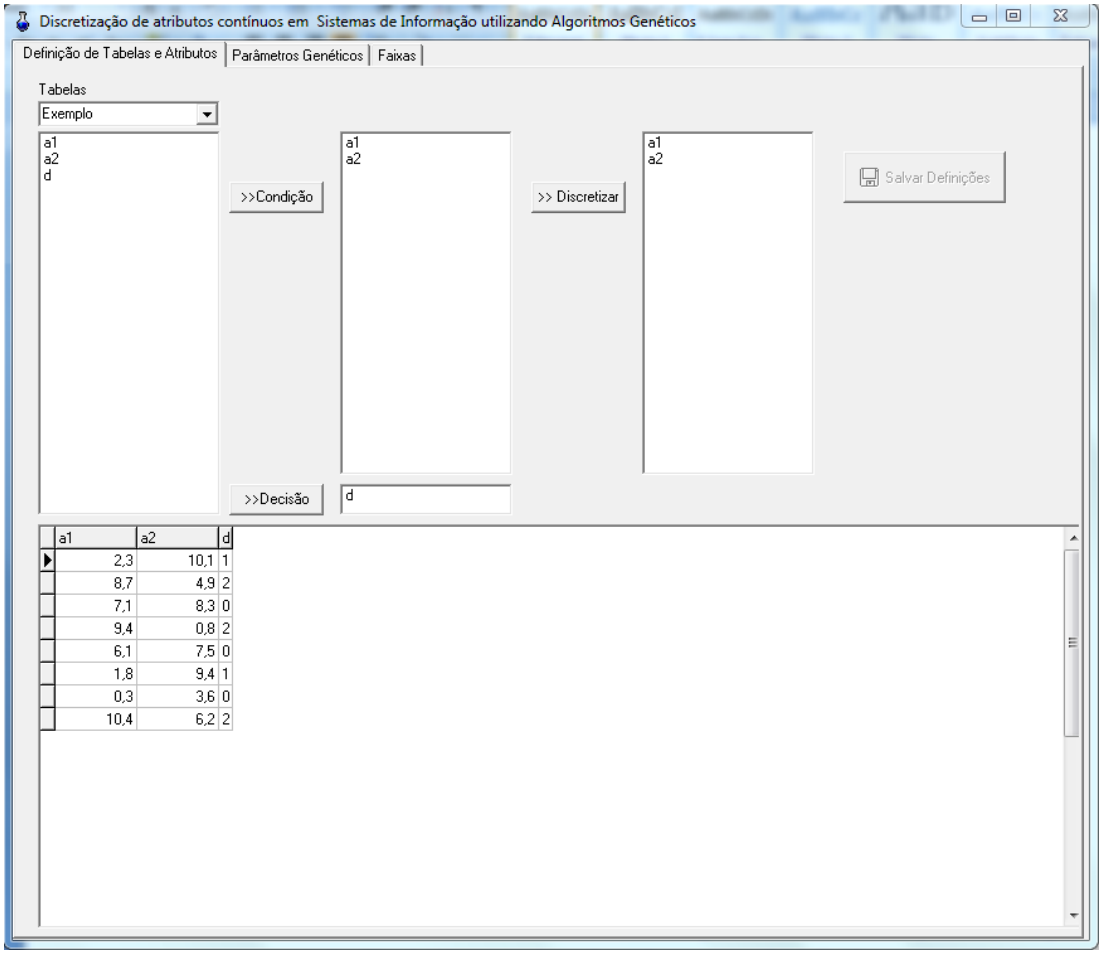


Figura 5.8 – Definição de Tabelas e Atributos

Parâmetros Genéticos: Nesta aba, todos os parâmetros genéticos são definidos. É definido também o critério de parada do AG e a função de avaliação que será utilizada. Pode-se observar na Figura 5.9 que foi definida uma população de 4 indivíduos. Nesse caso, o número de faixas para cada atributo foi calculado pelo método de Scott. O critério de parada escolhido foi o máximo de gerações e a função de avaliação utilizada pelo AG foi a Classe Majoritária.

Do lado direito da janela, é possível visualizar o histórico da busca enquanto o AG é executado. Para cada geração é informada a melhor nota obtida na geração, sendo 100

a nota máxima. São informados também os pontos de corte desse indivíduo para cada um de seus atributos. No exemplo, a nota 100 é a melhor nota das últimas gerações, o que significa que foram encontrados pontos de corte que não geram inconsistências.

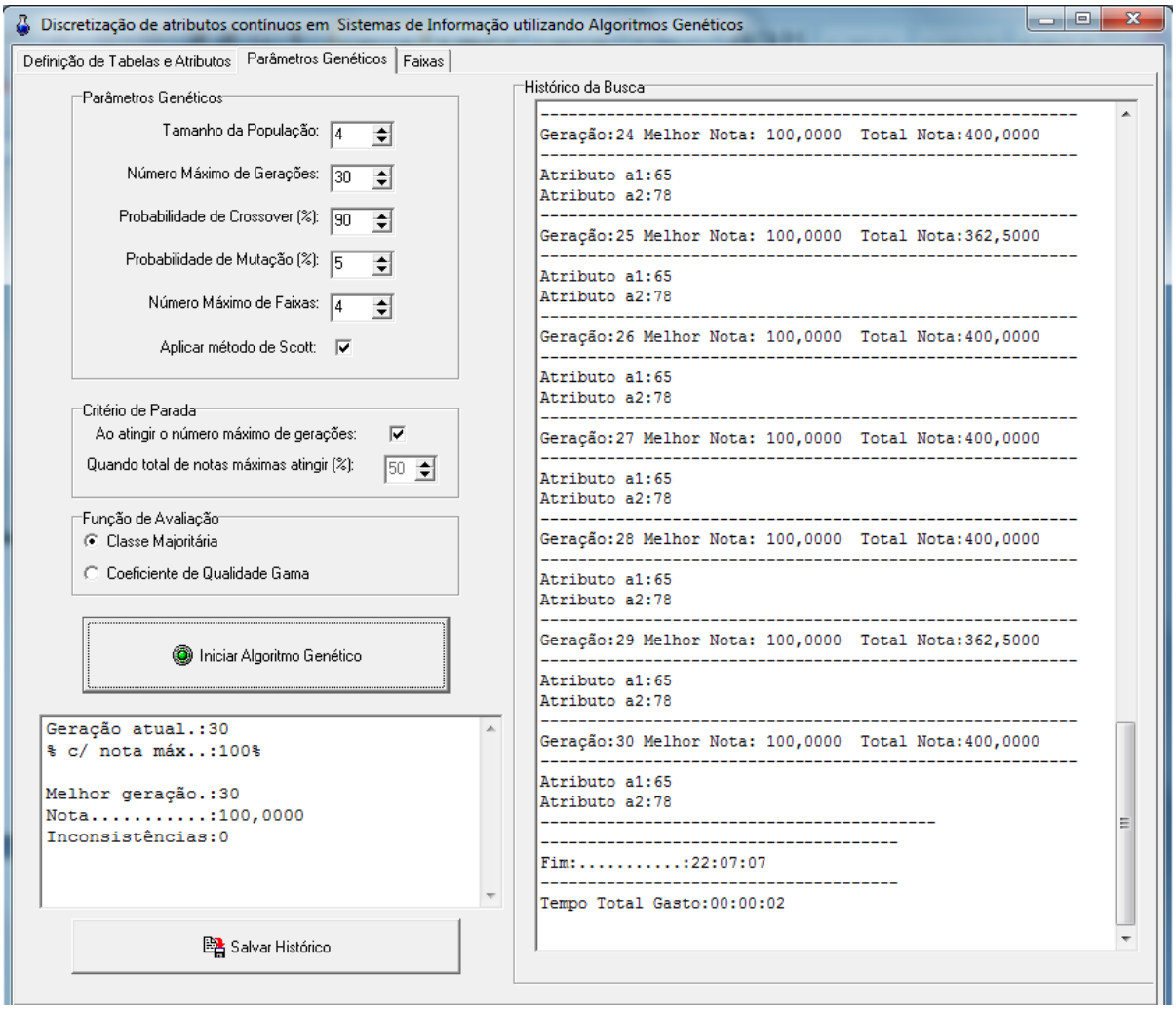


Figura 5.9 – Parâmetros Genéticos

Ainda nessa mesma aba, é exibido um resumo do histórico da busca com a porcentagem de indivíduos com nota máxima, a melhor geração e a nota correspondente ao melhor indivíduo dessa geração juntamente o número de inconsistências desse indivíduo. A Figura 5.10 destaca essas informações.

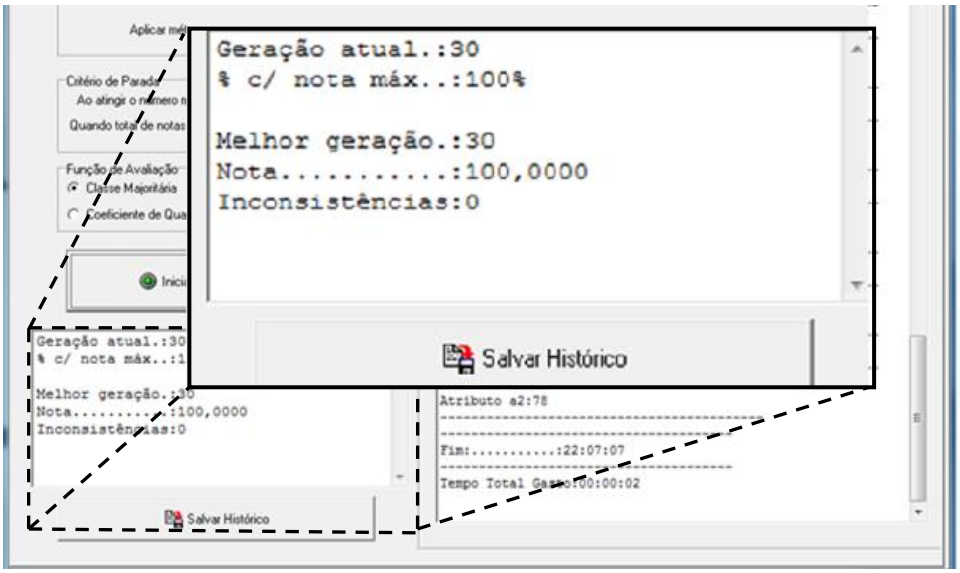


Figura 5.10 – Resumo da busca

Faixas: Nesta aba são apresentados os melhores pontos de corte obtidos em cada geração. O usuário pode selecionar uma geração e aplicar os respectivos pontos de corte na tabela original que terá seus atributos contínuos discretizados. No exemplo mostrado na Figura 5.11, foi aplicada na tabela os pontos de corte da geração 30. Os pontos de corte são 28 e 72. Os limites definidos por esses valores podem ser visualizados através do botão *Limites das Faixas* conforme mostra a Figura 5.12.

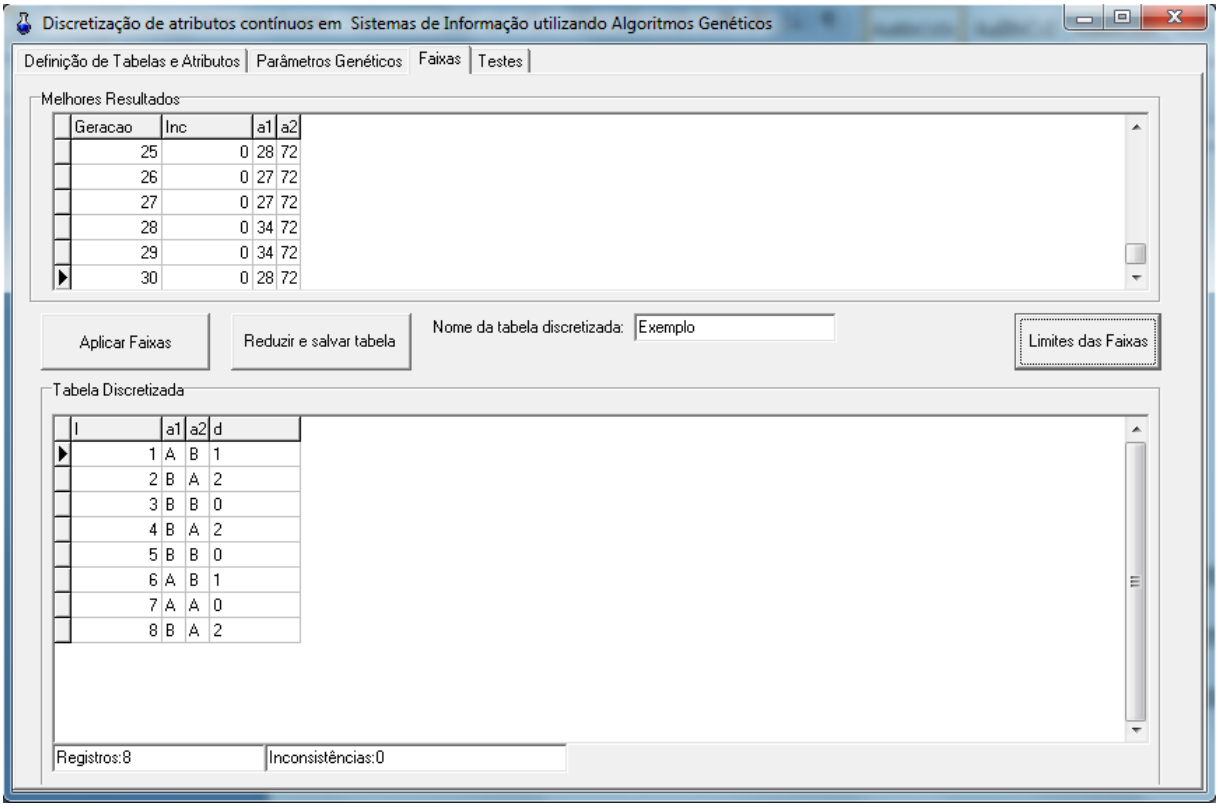


Figura 5.11 – Faixas

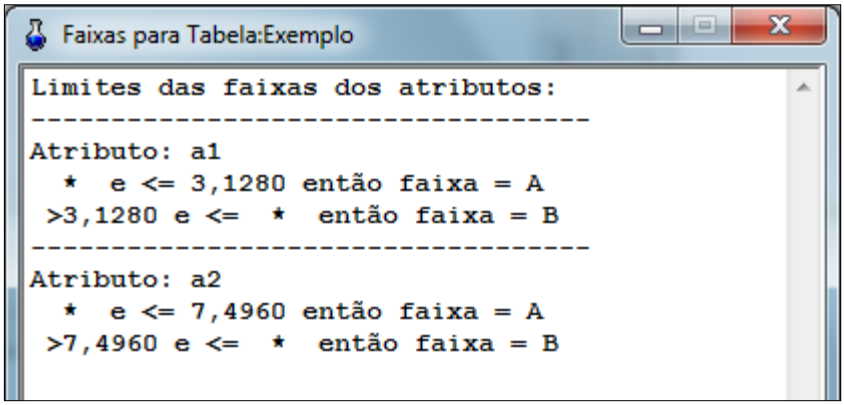
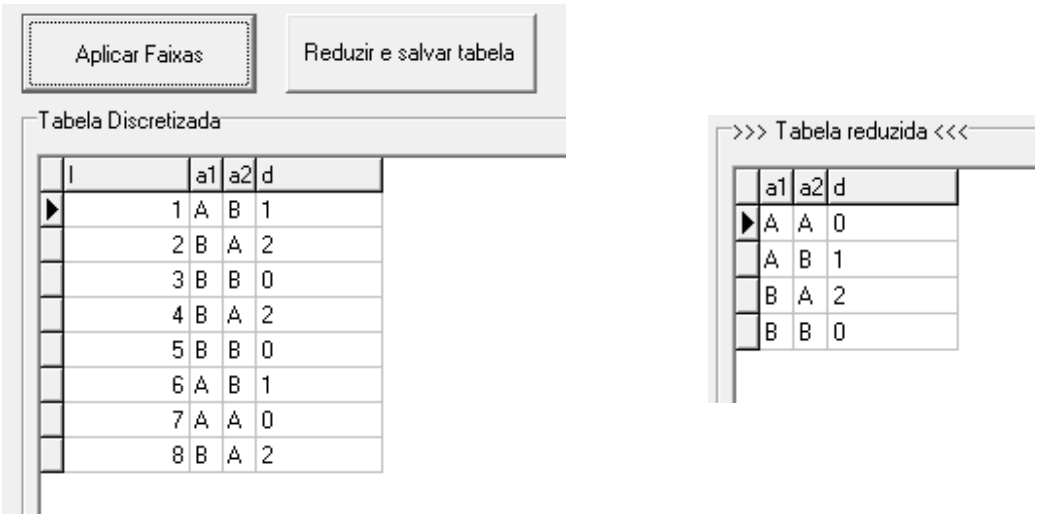


Figura 5.12 – Limites das faixas

Ainda nessa aba, a tabela discretizada será reduzida e gravada. Para isso o usuário deve utilizar o botão *Reduzir e salvar tabela*. A redução consiste em eliminar as repetições na tabela discretizada obtendo um conjunto mínimo de objetos. O nome dessa tabela também pode ser definido pelo usuário. A Figura 5.13 mostra o resultado da redução do exemplo em questão.



I	a1	a2	d
1	A	B	1
2	B	A	2
3	B	B	0
4	B	A	2
5	B	B	0
6	A	B	1
7	A	A	0
8	B	A	2

a1	a2	d
A	A	0
A	B	1
B	A	2
B	B	0

Figura 5.13 – Redução da tabela discretizada

Testes: Nesta aba, o usuário pode realizar testes de discretização. Para a realização dos testes, cada SI pode ser dividido em um conjunto de treinamento e um conjunto de teste. Os pontos de corte obtidos para o conjunto de treinamento são utilizados para discretizar o respectivo conjunto de teste. Os exemplos existentes nesse conjunto não foram utilizados pelo AG para encontrar os pontos de corte. Esses passos simulam a situação real na qual, após a criação do modelo com os exemplos disponíveis (conjunto de treinamento), novos exemplos devem ser classificados pelo modelo. A Figura 5.14 apresenta um conjunto de teste com quatro objetos.

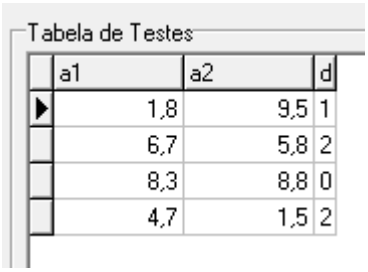


Figura 5.14 – Tabela de teste

A Figura 5.15 mostra a aba do programa onde uma tabela de teste pode ser selecionada. O teste é executado nos atributos definidos pelo usuário como reduto, no caso desse exemplo os atributos a1 e a2. O número de inconsistências é calculado e exibido obtendo-se, assim, informação utilizada na avaliação da discretização.

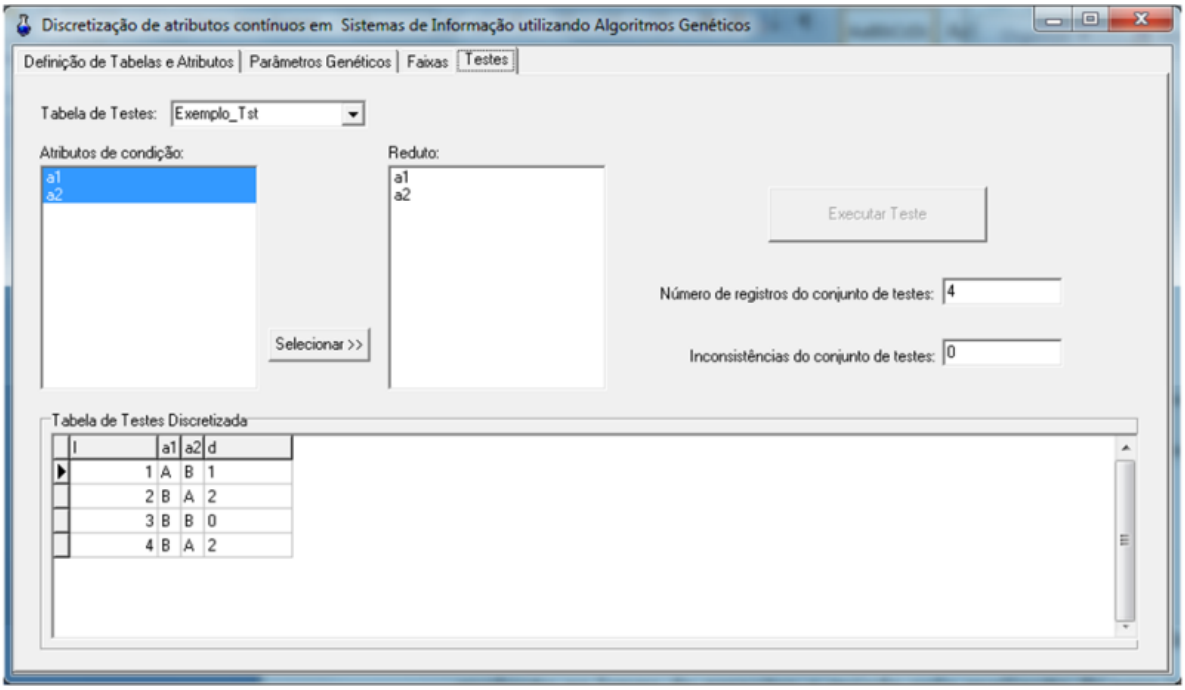


Figura 5.15 – Testes

5.5 CONSIDERAÇÕES FINAIS

Nesse capítulo, foi apresentado um AG capaz de realizar a discretização de atributos contínuos em Sistemas de Informação. Nenhum método determinístico foi utilizado para gerar pontos de corte para a população inicial. O único pré-processamento dos atributos contínuos, que pode ser realizado, é o cálculo do número de faixas por atributo através do método de Scott. Os pontos de corte são gerados ao acaso e, após gerações, evolui para soluções válidas.

A convergência do AG se dá a partir de uma representação que, para armazenar os pontos de corte, utiliza uma estrutura de lista ligada com alocação dinâmica de memória. Os operadores de seleção, cruzamento e mutação são executados sem que haja movimentação de dados na memória. Isto é feito apenas por ajustes de ponteiros de memória. Isso reduz consideravelmente o tempo de processamento e a quantidade de memória utilizada.

Um gargalo comum em qualquer AG é a função de avaliação. Nesse trabalho, a avaliação implica processar todo o SI transformando os atributos contínuos em faixas de acordo com a proposta sugerida por cada indivíduo da população para o cálculo do número de inconsistências. O SI é uma tabela que fica em memória secundária e relativamente lenta se comparada a memória principal. Contudo, através de *Queries SQL*, tanto a aplicação das faixas, bem como o cálculo das inconsistências, foram executadas em memória principal proporcionando assim o bom desempenho do AG.

O próximo capítulo apresenta alguns experimentos realizados com o método proposto com o objetivo de demonstrar sua eficiência.

Capítulo 6

EXPERIMENTOS

6.1 INTRODUÇÃO

No Capítulo 5 foi apresentado em detalhes o AG proposto para realizar discretizações de atributos contínuos em Sistemas de Informação. Foi apresentado também o programa desenvolvido que implementa esse AG.

Neste capítulo são apresentados experimentos realizados. A partir das discretizações realizadas pelo programa desenvolvido, os correspondentes redutos foram obtidos e o número de inconsistências calculado.

Os resultados obtidos com o AG podem ser comparados com os resultados obtidos pelo método de discretização não supervisionado *Equal-frequency Binning* (EFB) e pelo método supervisionado Entropia MDL. Para a discretização por esses dois métodos e cálculo de redutos, foi utilizado o programa ROSETTA (ROSETTA, 2009). Esse programa é uma ferramenta baseada na TCA e cobre todas as etapas do processo com opções para discretização, cálculo de redutos e geração de regras.

O objetivo dos experimentos apresentados é demonstrar a eficiência do AG proposto para a tarefa de discretização.

6.2 EXPERIMENTOS COM SISTEMAS DE INFORMAÇÕES REAIS

Para a realização dos testes, foram utilizadas cinco bases de dados que representam Sistemas de Informações. Seus dados referem-se a informações do mundo real, porém esses dados já sofreram pré-processamentos, muitas vezes eliminando-se erros, repetições e inconsistências.

As três primeiras bases utilizadas foram obtidas do repositório de dados da UCI (Asuncion, A., Newman, D., 2007). Elas foram selecionadas por apresentarem atributos

contínuos e tanto o número de atributos quanto o número de exemplos variam bastante. Além disso, essas bases são frequentemente referenciadas pela comunidade.

As bases de dados utilizadas são brevemente descritas a seguir:

Íris: essa base é composta por 50 exemplos de cada uma das três classes de flores Íris: Íris setosa, Íris versicolor e Íris virgínica. Os exemplos são classificados de acordo com os quatro atributos presentes na base: comprimento da sépala, largura da sépala, comprimento da pétala e largura da pétala;

Wdbc: o problema é predizer se uma amostra de tecido de mama obtida de uma paciente é benigna (B) ou maligna (M) baseada em dados histológicos.

Wine: essa base possui dados resultantes de análise química de vinhos produzidos em uma mesma região na Itália, mas, por diferentes vinicultores. A análise determinou 13 atributos encontrados que classificam três diferentes tipos de vinho.

As outras duas bases utilizadas são apresentadas em Moraes (2006) e Carvalho (2000). Estas bases foram escolhidas por, além de conter dados reais, já terem sidos aplicados sobre elas a Teoria dos Conjuntos Aproximados para a redução desses sistemas e obtenção de regras. Nesse caso, a discretização dos atributos contínuos foi feita de forma manual e com base na experiência do especialista. Essas bases são descritas a seguir:

Copel: o objetivo é classificar as condições operativas de um sistema elétrico da COPEL (Companhia Paranaense de Energia) composto de linhas de transmissão e subtransmissão de energia elétrica.

IEEE118: o objetivo é também classificar as condições operativas em um sistema elétrico para vários níveis de carregamento. Essa base possui um grande número de atributos de condição. No caso 118 atributos são utilizados para a classificação.

A Tabela 6.1 mostra um resumo das características dos conjuntos de dados organizado do seguinte modo:

 # Objetos: número de objetos do conjunto de dados;
Classes %: nome e distribuição das classes em porcentagem;
 # Atributos de condição: número total de atributos juntamente com a discrição dos atributos.

Tabela 6.1 - Características dos SI utilizados

Sistema de Informação	#Objetos	Classes %	#Atributos de condição
Íris	150	*Iris-setosa* 33,33 *Iris-versicolor* 33,33 *Iris-virginica* 33,33	4 {C1,C2,C3,C4}
Wdbc	569	B 62,74 M 37,26	30 {C1,C2,C3,...,C30}
Wine	178	Classe 1 33 Classe 2 40 Classe 3 27	13 {C1,C2,C3,...,C13}
Copel	32	S 46,87 A 40,62 U1 9,38 U2 3,13	18 {V1,V2,V3,...,V18}
IEEE118	12	Alerta 33,33 Seguro 41,67 Inseguro 25,00	118 {G1,G2,G3,...,G118}

6.3 CONFIGURAÇÃO DOS TESTES

As bases Íris, Wdbc e Wine foram divididas em um conjunto de treinamento com 90% do total de objetos de cada classe e um conjunto de teste com os 10% dos objetos restantes. Para outras duas bases, Copel e IEEE118, não foram criados conjuntos de testes em função do pequeno número de objetos existentes.

Os pontos de corte obtidos para o conjunto de treinamento são utilizados para discretizar o respectivo conjunto de teste. É importante ressaltar que dessa maneira os métodos de discretização não utilizaram informações do conjunto de teste para a obtenção dos pontos de corte. Esses passos simulam a situação real na qual, após a criação do modelo com os exemplos disponíveis (conjunto de treinamento), novos exemplos devem ser classificados pelo modelo.

A partir da base discretizada, o programa ROSETTA foi utilizado para obter o reduto. Utilizando apenas os atributos definidos no reduto, o conjunto de teste foi discretizado para o cálculo da taxa de erro de classificação. A Figura 6.1 ilustra como os testes foram realizados.

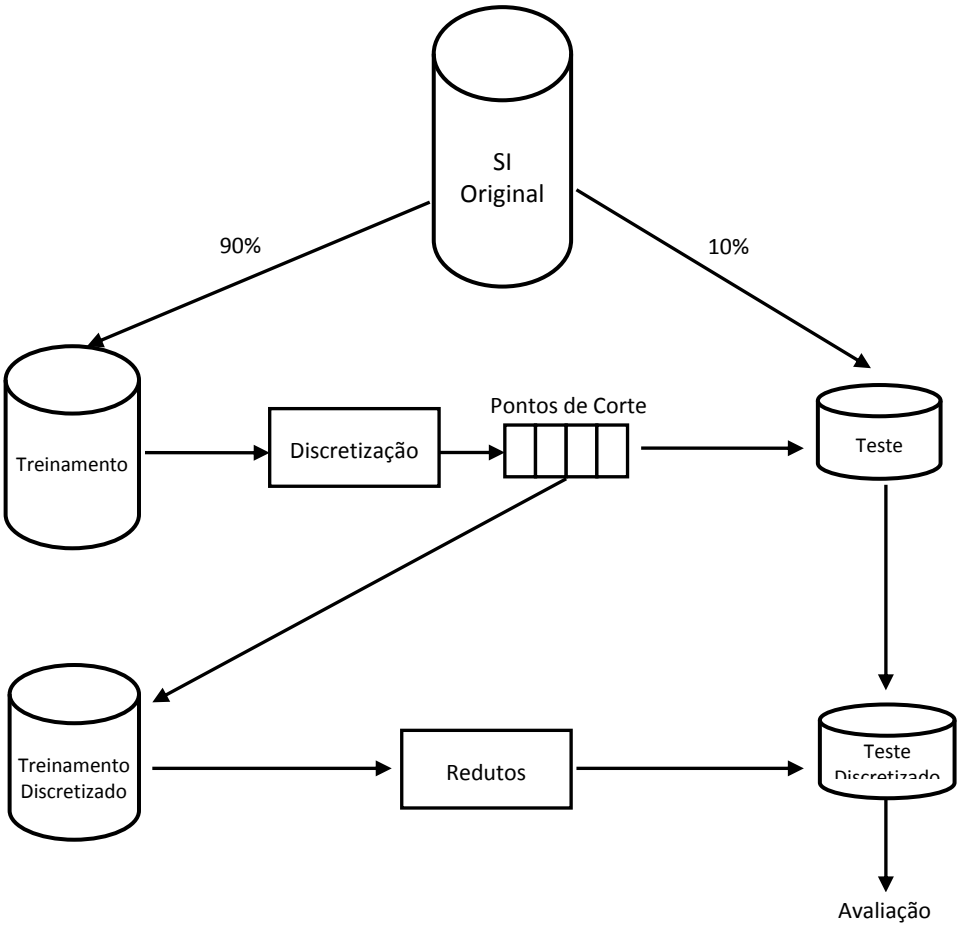


Figura 6.1 – Configuração dos testes

6.4 MEDIDAS UTILIZADAS PARA AVALIAÇÃO

As medidas consideradas para a avaliação foram:

Número de intervalos criados: Para o conjunto de treinamento é verificado o número de intervalos distintos criados. No caso do método proposto, o número de intervalos distintos poderá ser definido pelo próprio AG ou pelo método de Scott, conforme apresentado no Capítulo 5. No método *Equal-frequency Binning*, o número de intervalos deve ser definido pelo usuário. O método da Entropia MDL define automaticamente o número de intervalos para cada atributo do SI.

Número de inconsistências na Base de Dados: Será calculada a partir do conjunto de treinamento discretizado. Qualquer método de discretização pode causar inconsistências que devem ser identificadas podendo, assim avaliar a qualidade da discretização.

Precisão da predição Após a discretização de um SI, o conjunto de testes do respectivo SI será submetido a discretização com objetivo de predizer as respectivas classes.

6.5 RESULTADOS OBTIDOS

A Tabela 6.2 apresenta os resultados obtidos para as bases Íris, Wdbc e Wine. Os resultados das demais bases, Copel e IEEE118, são apresentados na Tabela 6.3.

São informados os resultados obtidos na discretização pelo Algoritmo Genético (AG), pelo método EFB e Entropia MDL. Para cada método, são informados os resultados das medidas consideradas para a avaliação.

6.5.1 Resultados das bases Íris, Wdbc e Win

Íris: Essa base foi discretizada pelo AG, criando 16 intervalos (ou faixas). A Figura 6.2 apresenta os pontos de corte definidos para cada atributo. A discretização obtida não gerou nenhuma inconsistência no conjunto de treinamento; o que permitiu a obtenção de um reduto com três atributos, no caso, C2, C3 e C4. Esses atributos são suficientes para classificar todos os objetos da base com precisão de 100% na predição, ou seja, nenhum objeto do conjunto de testes foi classificado incorretamente com a discretização.

Tabela 6.2 – Resultados Obtidos – Íris, Wdbc e Wine

Sistema de Informação	Método	Número de intervalos criados	Inconsistências no conjunto de treinamento (%)	Número de atributos do reduto	Precisão da predição para o Conjunto de Teste (%)
Íris	AG	16	0	3	100
	EFB	16	4	4	93
	Entropia MDL	35	0	3	100
Wdbc	AG	120	0	9	70
	EFB	120	0	7	100
	EntropiaMDL	2053	-	-	-
Wine	AG	39	0	9	100
	EFB	39	0	5	100
	Entropia MDL	174	-	-	-

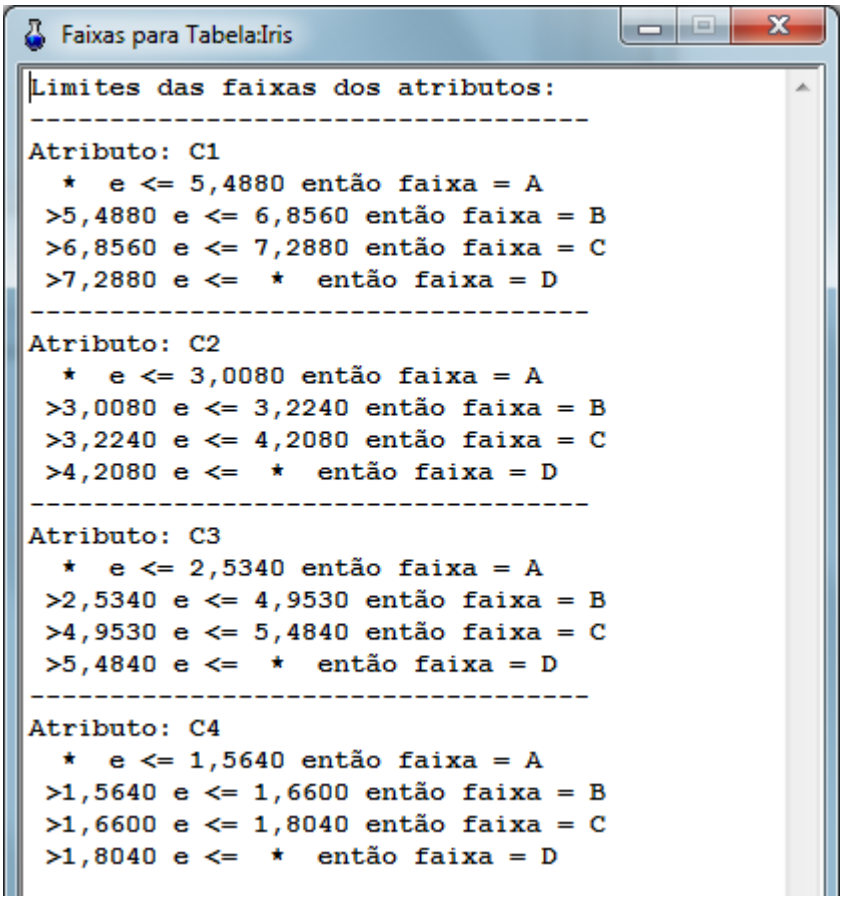


Figura 6.2 – Pontos de corte para os SI Íris

O Método EFB resultou em uma discretização com 4% de inconsistências, ou seja, os pontos de corte definidos pelo método geram inconsistências no conjunto de treinamento e também no conjunto de testes, em que a precisão de predição foi de 93%.

O método da Entropia MDL obteve uma discretização consistente para o conjunto de treinamento e teste com 100% de precisão na predição. O reduto obtido possui também três atributos, no caso, C1, C3 e C4. O número de intervalos gerados foi superior ao do AG e EFB que tiveram, definidos pelo usuário, o número máximo de 4 intervalos por atributo. Isso não ocorre no método da Entropia onde o número de intervalos é definido pelo seu próprio algoritmo.

Wdbc: Para essa base, o AG obteve uma discretização consistente. Para isso foram criados 120 intervalos, sendo 4 para cada atributo. A discretização obtida não gerou nenhuma inconsistência no conjunto de treinamento o que permitiu a obtenção de um reduto com 9 atributos. Os pontos de cortes obtidos são apresentados no Anexo III. Para o conjunto de testes a precisão foi de 70 % na predição.

O método EFB também conseguiu realizar uma discretização consistente com o mesmo número de intervalos gerados pelo AG com um reduto ainda menor atingindo 100% de precisão para o conjunto de testes.

O método da Entropia implementado no ROSETTA apresentou dificuldade na discretização dessa base. Foi criado um número excessivamente alto de intervalos (2053). Como citado anteriormente, nesse método o número de intervalos é definido pelo seu próprio algoritmo. Isso inviabilizou a utilização dessa discretização para obtenção de redutos e regras.

Wine: Para essa base, o AG obteve uma discretização com 39 intervalos sem inconsistências no conjunto de treinamento o que permitiu a obtenção de um reduto com 9 atributos. Para o conjunto de teste, a precisão da predição foi igual a 100%. Os pontos de cortes obtidos são apresentados no Anexo IV.

O método EFB também obteve uma discretização consistente para o conjunto de treinamento o que permitiu a obtenção de um reduto com 5 atributos e 100% de precisão de predição para o conjunto de testes.

Nesse caso, o método e Entropia MDL não pode ser avaliado devido ao número excessivo de intervalos criados.

6.5.2 Resultados das bases Copel e IEEE118

Para essas bases não foram criados conjuntos de testes em função do pequeno número de objetos existentes em cada uma. Assim, a tarefa de discretização, tem o objetivo apenas de encontrar bons pontos de corte que permitam a obtenção de redutos e regras de decisão. A Tabela 6.3 apresenta os resultados.

Tabela 6.3 – Resultados obtidos – Copel e IEEE118

Sistema de Informação	Método	Número de intervalos criados	Inconsistências no conjunto de treinamento (%)	Número de atributos do reduto
Copel	AG	54	0	3
	EFB	54	0	4
	Entropia MDL	47	0	3
IEEE118	AG	140	0	3
	EFB	209	0	3
	Entropia MDL	2141	-	-

Copel: Moraes (2006) e Carvalho (2000) discretizam essa base manualmente utilizando os mesmos limites para todos os atributos. Os valores definidos foram:

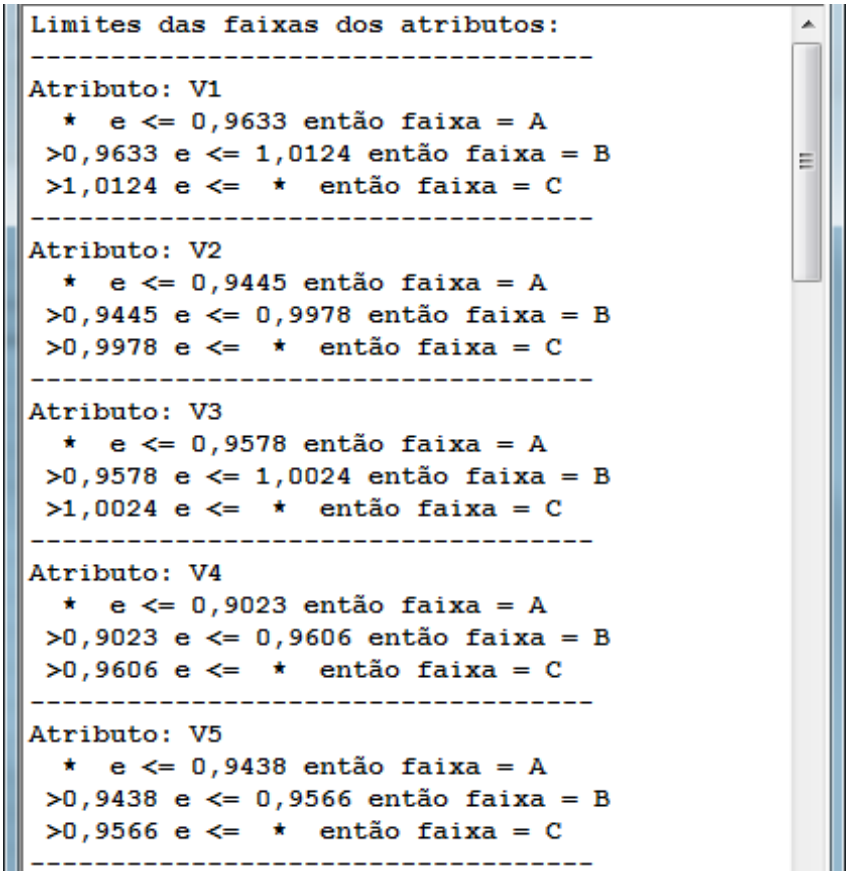
L - valores inferiores ou iguais a 0,98

M - valores entre 0,98 a 1,00 (inclusive)

H - valores superiores a 1,00

A partir dessa discretização foi possível a obtenção de redutos com, no mínimo, 5 atributos.

O AG, por sua vez, definiu automaticamente limites específicos para cada atributo do SI. Alguns desses limites são apresentados na Figura 6.3. Os outros dois métodos de discretização também encontraram discretizações consistentes para essa base.



Figuras 6.3 – Limites para os atributos da base Copel

IEEE118: Moraes (2006) utiliza os mesmos limites para discretizar os 118 atributos dessa base. Os limites definidos foram os mesmos definidos para a base Copel. Foram então definidos 3 intervalos para cada atributo o que resultou num total de 354 intervalos. Com essa discretização o menor reduto obtido possui 14 atributos.

O AG e método EFB nesse caso obtiveram uma discretização consistente e criaram relativamente poucos intervalos; uma vez que essa base possui 118 atributos de condição. Já o método da Entropia MDL não obteve sucesso gerando um número alto de intervalos e alguns de seus atributos não foram discretizados.

O AG discretizou essa base permitindo a obtenção de um conjunto mínimo de intervalos. O método de Scott definiu para a maioria dos atributos apenas dois intervalos o que resultou num total de 140 intervalos. Com essa discretização foi possível obter redutos com apenas 3 atributos.

6.6 EXPERIMENTO COM UM PEQUENO SISTEMA DE INFORMAÇÃO

O objetivo deste teste é mostrar que a discretização obtida pelo Algoritmo Genético proposto é capaz de proporcionar a descoberta de conhecimento implícito no Sistema de Informação pelas técnicas da TCA. Isso pode não ocorrer com a utilização dos métodos EFB e Entropia MDL porque são métodos do tipo univariados, ou seja, consideram um atributo contínuo por vez, não levando em conta a relação entre os atributos. Essa relação é descoberta pelo AG proposto uma vez que realiza a discretização de todas as variáveis simultaneamente verificando a consistência em relação a classe.

A Tabela 6.4 apresenta um SI extraído do trabalho de Moraes (2006), com exemplos de condições operativas de transformadores de energia elétrica de redes de distribuição. Todos os atributos dessa tabela foram discretizados da seguinte maneira: de 0 a 12 foi mapeado como “Baixo”, de 12 a 30 mapeado como “Médio” e de 30 a 45 KVA, como “Alto”. Assim, a Tabela 6.4 pode ser reescrita na forma da Tabela 6.5.

A partir da Tabela 6.5, foi obtido o reduto com dois atributos: Trafo C e Trafo D. Esses dois atributos classificam, no caso, a operação como Normal, Leve ou Carregado. Os atributos Trafo A e Trafo B foram identificados como supérfluos. Assim, eliminando-se os

atributos dispensáveis e agrupando-se os exemplos idênticos forma-se a Tabela 6.6.

Tabela 6.4 – Conjunto de Dados Inicias (Morais,2006)

Trafo A	Trafo B	Trafo C	Trafo D	Saída
10	34	33	18	Normal
21	32	8	15	Leve
22	8	37	18	Normal
23	10	35	15	Normal
25	9	36	17	Normal
25	24	10	35	Leve
32	32	25	8	Carregado
35	33	33	7	Carregado
5	35	8	17	Leve
5	35	35	7	Carregado
7	32	9	15	Leve
8	32	40	5	Carregado

Tabela 6.5 – Conjunto de Dados Iniciais Discretizados (Morais,2006)

Trafo A	Trafo B	Trafo C	Trafo D	Saída
Baixo	Alto	Alto	Médio	Normal
Médio	Alto	Baixo	Médio	Leve
Médio	Baixo	Alto	Médio	Normal
Médio	Baixo	Alto	Médio	Normal
Médio	Baixo	Alto	Médio	Normal
Médio	Médio	Baixo	Alto	Leve
Alto	Alto	Médio	Baixo	Carregado
Alto	Alto	Alto	Baixo	Carregado
Baixo	Alto	Baixo	Médio	Leve
Baixo	Alto	Alto	Baixo	Carregado
Baixo	Alto	Baixo	Médio	Leve
Baixo	Alto	Alto	Baixo	Carregado

Tabela 6.6 – Conjunto de Dados Iniciais sem Atributos Dispensáveis (Morais,2006)

Trafo C	Trafo D	Saída
Alto	Médio	Normal
Baixo	Médio	Leve
Baixo	Alto	Leve
Médio	Baixo	Carregado
Alto	Baixo	Carregado

A Tabela 6.6 gerou o conjunto de regras a seguir (Moraes, 2006):

Se TrafoC = Alto e	TrafoD = Médio	Então Saída = Normal
Se TrafoC = Baixo		Então Saída = Leve
Se	TrafoD = Alto	Então Saída = Leve
Se TrafoC = Médio		Então Saída = Carregado
Se	TrafoD = Baixo	Então Saída = Carregado

Ocorreu uma sensível redução na base de dados, sem perda de informação, pois todos os conhecimentos importantes da tabela original constam da Tabela 6.6 e também nas regras (Moraes, 2006).

Serão apresentados a seguir os resultados da discretização dessa base pelo AG e também pelo Método da Entropia MDL. O método EFB não foi utilizado por ser não supervisionado e, nesse caso, considerar a classe durante a discretização é fundamental.

A Figura 6.4 apresenta a tela do programa desenvolvido onde a discretização pelo AG foi realizada.

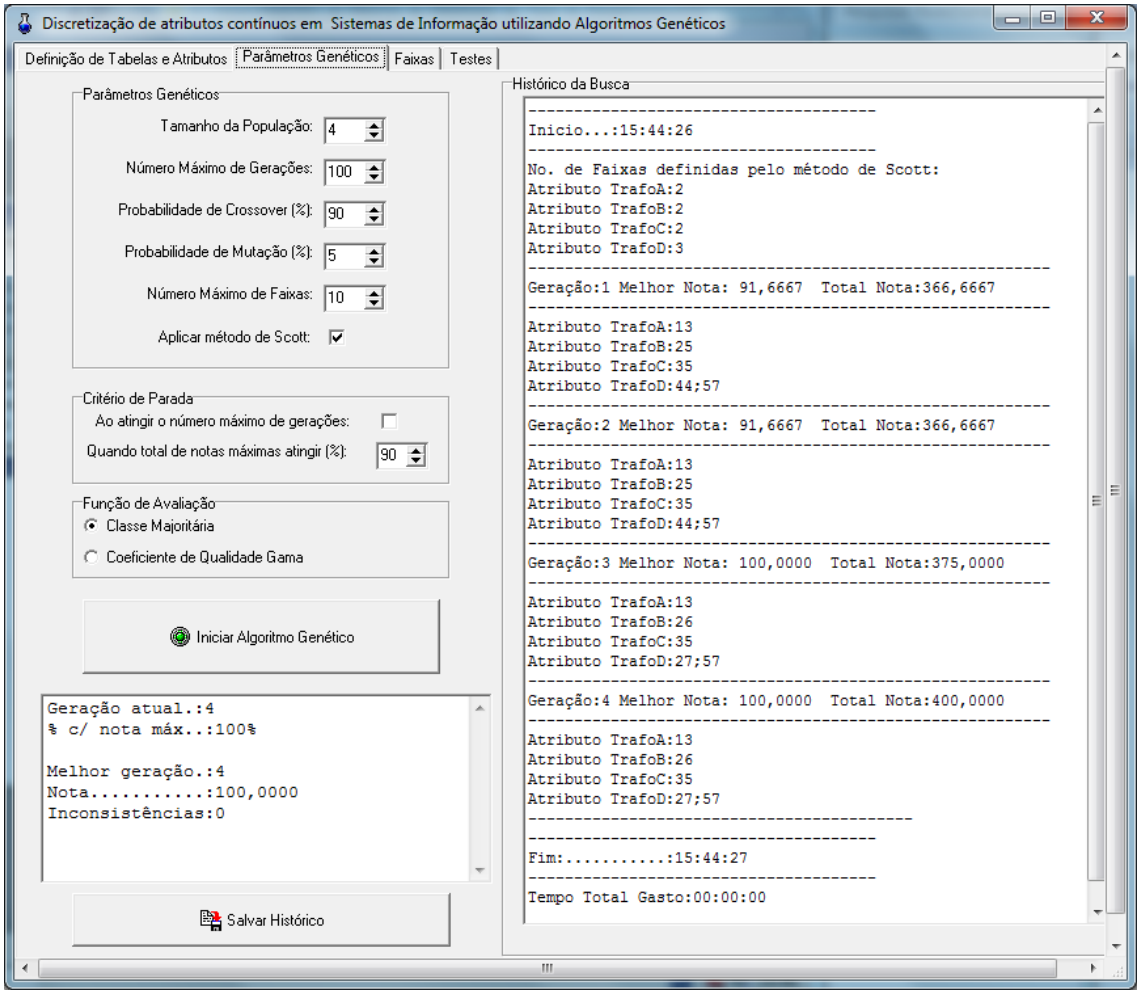


Figura 6.4 – Discretização utilizando o AG

As faixas obtidas pelo AG são mostradas na Figura 6.5.

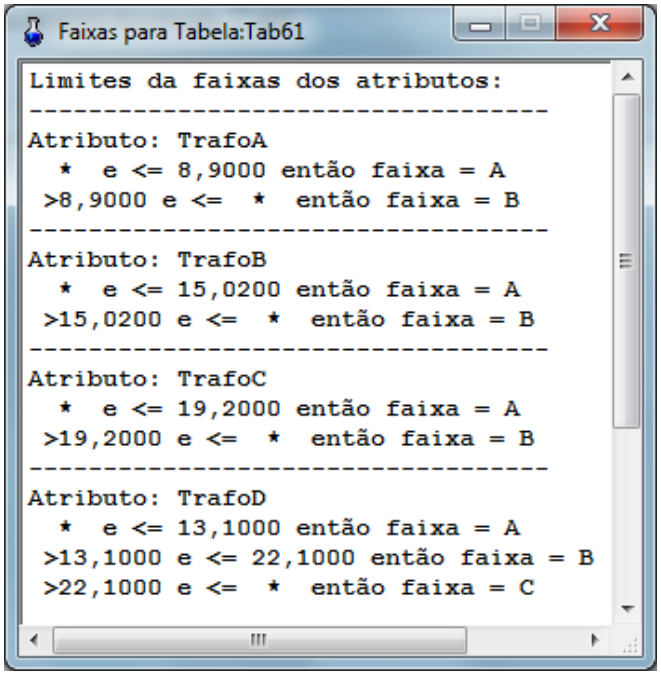


Figura 6.5 – Faixas definidas pelo AG

A Figura 6.6 mostra a discretização da Tabela 6.4 no programa de acordo com as faixas definidas pelo AG.

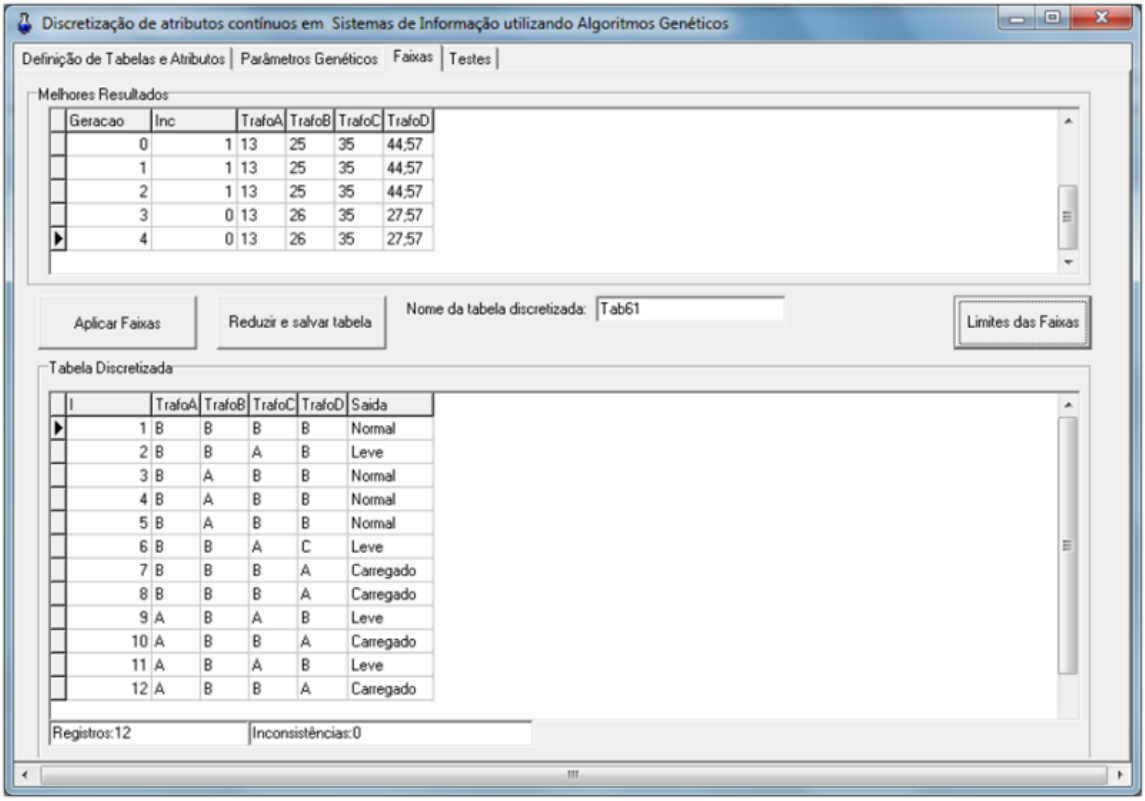


Figura 6.6 – Discretização da Tabela 6.4

A Figura 6.7 mostra a redução da tabela discretizada:


>>> Tabela reduzida <<<

TrafoA	TrafoB	TrafoC	TrafoD	Saida
A	B	A	B	Leve
A	B	B	A	Carregado
B	A	B	B	Normal
B	B	A	B	Leve
B	B	A	C	Leve
B	B	B	A	Carregado
B	B	B	B	Normal

Figura 6.7 – Tabela discretizada reduzida

Depois da redução da tabela, que eliminou as repetições, o programa ROSETTA foi utilizado para encontrar os redutos e regras. A Figura 6.8 mostra o reduto obtido para a tabela discretizada e as regras de decisão para esse reduto.

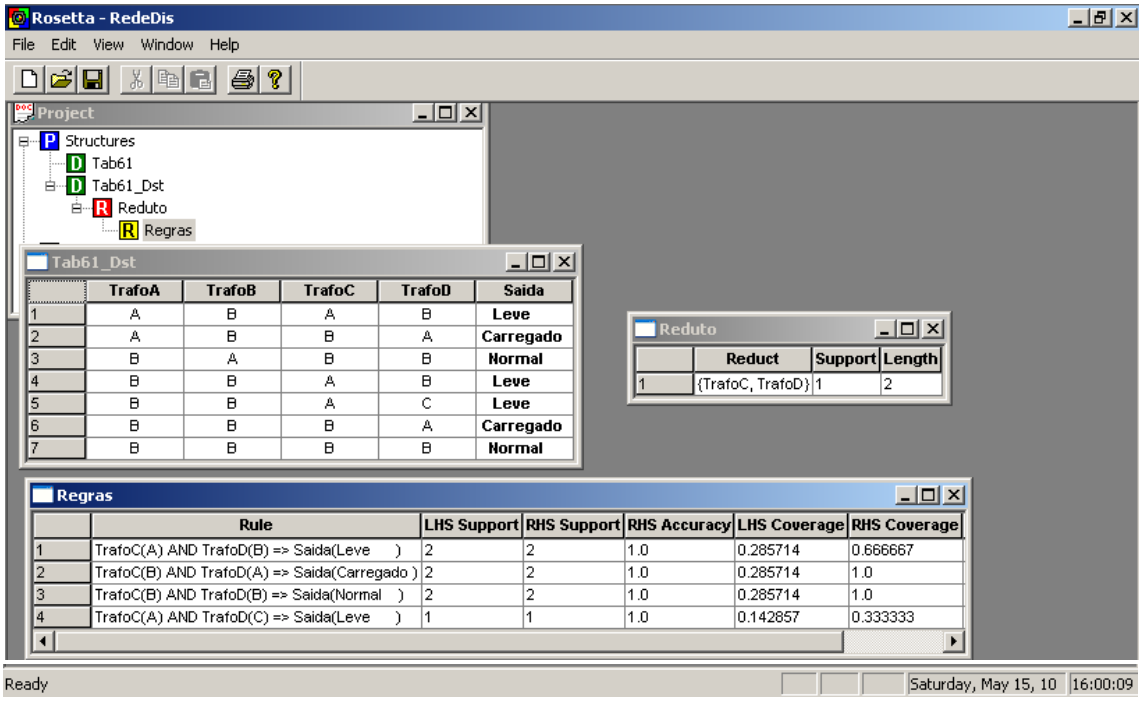


Figura 6.8 – Tela do ROSETTA com reduto e regras

Numericamente as regras obtidas são:

Se TrafoC <=19,2 Então Saída = Leve
Se TrafoC > 19,2 e TrafoD <= 13,1 Então Saída = Carregado
Se TrafoC > 19,2 e TrafoD > 13,1 e TrafoD <= 22,1 Então Saída = Normal

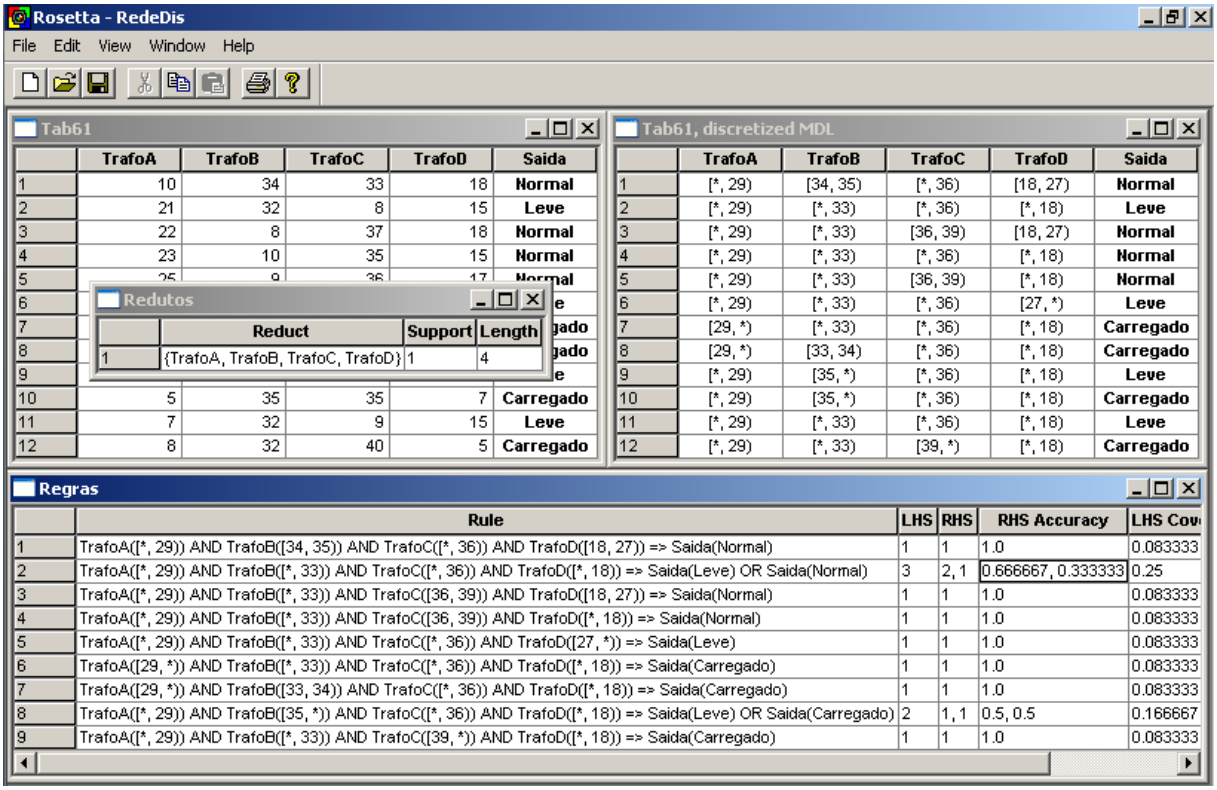
As regras obtidas representam o conhecimento retirado da Tabela 6.4 a partir da discretização realizada pelo AG. Na discretização feita manualmente foram definidos três intervalos para cada um dos atributos da tabela. Na discretização feito pelo AG, apenas o

atributo TrafoD necessitou de três intervalos. Para os demais atributos, dois intervalos foram suficientes para uma discretização consistente. Isso resultou na geração de um número menor de regras.

Em relação ao reduto obtido, tanto na discretização definida pelo especialista como na discretização definida pelo AG, os atributos que o compõem são os mesmos. Isso pode ser considerado como uma validação da discretização.

A seguir serão apresentados os resultados da discretização e obtenção de redutos e regras de decisão utilizando a discretização pelo método da Entropia MDL.

A Figura 6.9 mostra a discretização da Tabela 6.4 pelo programa ROSETTA. Na janela "*discretized MDL*" cada atributo foi discretizado pelo método da Entropia MDL. A janela "Redutos" mostra que não foi possível encontrar redutos para a tabela discretizada. As regras geradas mostram que a tabela foi discretizada gerando inconsistências. Isso pode ser visualizado na janela "Regras", linhas 2 e 9, onde para uma respectiva entrada existem duas saídas diferentes.



	TrafoA	TrafoB	TrafoC	TrafoD	Saida
1	10	34	33	18	Normal
2	21	32	8	15	Leve
3	22	8	37	18	Normal
4	23	10	35	15	Normal
10	5	35	35	7	Carregado
11	7	32	9	15	Leve
12	8	32	40	5	Carregado

	TrafoA	TrafoB	TrafoC	TrafoD	Saida
1	[*, 29)	[34, 35)	[*, 36)	[18, 27)	Normal
2	[*, 29)	[*, 33)	[*, 36)	[*, 18)	Leve
3	[*, 29)	[*, 33)	[36, 39)	[18, 27)	Normal
4	[*, 29)	[*, 33)	[*, 36)	[*, 18)	Normal
5	[*, 29)	[*, 33)	[36, 39)	[*, 18)	Normal
6	[*, 29)	[*, 33)	[*, 36)	[27, *)	Leve
7	[29, *)	[*, 33)	[*, 36)	[*, 18)	Carregado
8	[29, *)	[33, 34)	[*, 36)	[*, 18)	Carregado
9	[*, 29)	[35, *)	[*, 36)	[*, 18)	Leve
10	[*, 29)	[35, *)	[*, 36)	[*, 18)	Carregado
11	[*, 29)	[*, 33)	[*, 36)	[*, 18)	Leve
12	[*, 29)	[*, 33)	[39, *)	[*, 18)	Carregado

	Reduct	Support	Length
1	{TrafoA, TrafoB, TrafoC, TrafoD}	1	4

	Rule	LHS	RHS	RHS Accuracy	LHS Cov
1	TrafoA([*, 29)) AND TrafoB([34, 35)) AND TrafoC([*, 36)) AND TrafoD([18, 27)) => Saida(Normal)	1	1	1.0	0.083333
2	TrafoA([*, 29)) AND TrafoB([*, 33)) AND TrafoC([*, 36)) AND TrafoD([*, 18)) => Saida(Leve) OR Saida(Normal)	3	2, 1	0.666667, 0.333333	0.25
3	TrafoA([*, 29)) AND TrafoB([*, 33)) AND TrafoC([36, 39)) AND TrafoD([18, 27)) => Saida(Normal)	1	1	1.0	0.083333
4	TrafoA([*, 29)) AND TrafoB([*, 33)) AND TrafoC([36, 39)) AND TrafoD([*, 18)) => Saida(Normal)	1	1	1.0	0.083333
5	TrafoA([*, 29)) AND TrafoB([*, 33)) AND TrafoC([*, 36)) AND TrafoD([27, *)) => Saida(Leve)	1	1	1.0	0.083333
6	TrafoA([29, *)) AND TrafoB([*, 33)) AND TrafoC([*, 36)) AND TrafoD([*, 18)) => Saida(Carregado)	1	1	1.0	0.083333
7	TrafoA([29, *)) AND TrafoB([33, 34)) AND TrafoC([*, 36)) AND TrafoD([*, 18)) => Saida(Carregado)	1	1	1.0	0.083333
8	TrafoA([*, 29)) AND TrafoB([35, *)) AND TrafoC([*, 36)) AND TrafoD([*, 18)) => Saida(Leve) OR Saida(Carregado)	2	1, 1	0.5, 0.5	0.166667
9	TrafoA([*, 29)) AND TrafoB([*, 33)) AND TrafoC([39, *)) AND TrafoD([*, 18)) => Saida(Carregado)	1	1	1.0	0.083333

Figura 6.9 – Discretização pelo Método da Entropia MDL

O Algoritmo da Entropia MDL implementado no ROSETTA não foi capaz de encontrar uma discretização consistente para a tabela. Apesar dessa tabela possuir um pequeno número de objetos, na sua discretização, é necessário considerar a relação existente entre os atributos para que não sejam geradas inconsistências.

6.7 CONSIDERAÇÕES FINAIS

Nesse capítulo foram apresentados alguns experimentos com o método de discretização proposto. Os testes mostraram que o AG é capaz de obter uma discretização consistente para um Sistema de Informação, principalmente, quando existe relação entre seus atributos na definição da classe.

O número de intervalos definidos pelo AG pode ser calculado pelo método de Scott respeitando o máximo definido pelo usuário. Isso impediu a criação de muitos intervalos para as bases com um grande número de objetos. O AG buscou e encontrou uma discretização consistente, utilizando o número de intervalos definidos. Ao final da busca, várias opções de discretização são disponibilizadas pelo AG. O usuário pode testar cada uma seguindo algum critério para isso, como por exemplo, a discretização que gera o menor reduto.

No caso do SI Wdbc, o método EFB alcançou 100% na predição para o conjunto de testes enquanto que o AG alcançou apenas 70% (Tabela 6.2). Esse resultado é apresentado para ressaltar que o AG apresenta, entre outras, soluções que não são totalmente válidas. Uma solução válida poderia ser obtida com a realização de uma nova busca onde, possivelmente, melhores resultados fossem encontrados. Explorar diferentes espaços de solução é uma das características dos AG´s.

Os métodos de Entropia MDL e EFB mostraram-se eficientes na maioria dos experimentos. O problema é quando um desses métodos não realiza uma discretização consistente para um determinado SI. Nesse caso, ao serem executados novamente sobre o mesmo SI, não se obtém uma discretização diferente, persistindo, assim, as inconsistências. Diferentemente, o AG faz uma busca gerando uma "população" de possibilidades de discretização consistentes para um determinado SI.

Para o exemplo da Tabela 6.4, extraído do trabalho de Moraes (2006), o AG realizou a discretização de forma que regras pudessem ser obtidas automaticamente. Não foi necessário o conhecimento do especialista na etapa de discretização. O fato importante é que a relação

entre os atributos foi preservada de forma a obter a discretização consistente. Isso não foi possível com a utilização do método da Entropia MDL implementado no ROSETTA.

Capítulo 7

CONCLUSÕES

Essa dissertação de mestrado apresenta uma contribuição para o processo de discretização para extração de conhecimento em grandes bases de dados através da Teoria dos Conjuntos Aproximados (TCA). Essa teoria matemática tem por objetivo separar os eventos existentes em três grupos a saber: aqueles que fazem parte da solução, aqueles que não fazem parte da solução e aqueles que fazem parte de uma faixa intermediária que podem ser parte da solução. Esses três grupos dão origem aos dois conjuntos básicos dessa teoria: o conjunto de aproximação inferior e o conjunto de aproximação superior. O conjunto de aproximação inferior contém o grupo de eventos que "fazem parte da solução"; enquanto o conjunto de aproximação superior contém o grupo de eventos que "fazem parte da solução" e a faixa intermediária dos eventos que podem fazer parte da solução. Dessa forma, o conjunto de aproximação inferior é sempre um subconjunto do conjunto de aproximação superior.

Em um algoritmo de extração de conhecimento utilizando a TCA, busca-se encontrar um conjunto que possui todos os eventos que fazem parte da solução, ou seja, um conjunto que contém o conjunto dos eventos que "fazem parte da solução" mais aquele da faixa intermediária que também faz parte da solução, excluindo-se os eventos os quais estão na faixa intermediária e que não fazem parte da solução do problema. Assim, pode-se afirmar que o conjunto procurado é um superconjunto do conjunto de aproximação inferior e um subconjunto do conjunto de aproximação superior.

Logo, a idéia central do algoritmo de extração de conhecimento utilizado nessa dissertação é tentar expandir o conjunto de aproximação inferior colocando dentro dele mais eventos que fazem parte da solução e contrair durante sua execução o conjunto de aproximação superior excluindo os eventos que não fazem parte da solução. Isso pode ser feito de várias maneiras. A optada nesse trabalho divide o espaço de busca dos eventos de forma discreta, de modo que eventos que fazem parte de um sub-espaço discreto são incluídos ou excluídos da solução a cada passo do processo de extração de conhecimento. Assim, é

claro que a maneira como é feita essa discretização sensibiliza a qualidade das regras desse processo de extração de conhecimento de uma base de dados.

O processo de discretização dos eventos na TCA tem sido feita através da discretização de cada uma das variáveis de forma independente, criando assim um sub-espaço no espaço de busca. Essas variáveis são também chamadas de atributos e representam os itens importantes no processo de classificação de eventos. Além disso, esse processo de discretização dos atributos cria faixas dentro da região de observação, denominada, simplesmente nesse trabalho, como faixas.

Usualmente, o processo de discretização dos atributos, ou seja, de criação das faixas, é feito através de critérios que utilizam conhecimentos passados do usuário ou fazem a divisão segundo algum critério de agrupamento. Aqueles que utilizam conhecimentos dos usuários normalmente seguem faixas pré-determinadas, sejam elas operativas, de algum tipo de restrição do atributo ou de interesse do usuário.

Essa dissertação apresenta uma contribuição na segunda forma de discretização, ou seja, aquela que tenta agrupar os eventos disponíveis na base de dados através de um método de otimização global, no caso os Algoritmos Genéticos (AG), utilizando as características dos agrupamentos existentes nos eventos.

O AG apresentado mostrou-se capaz de realizar a discretização de atributos contínuos em Sistemas de Informação. Isso foi conseguido graças a sua característica de evolução e pela avaliação eficiente de seus indivíduos. Nenhum método determinístico foi utilizado para gerar pontos de corte para a população inicial, bem como qualquer outro tipo de pré-processamento do Sistema de Informação não discretizado. A população inicial é gerada aleatoriamente e, após gerações, evolui para soluções válidas.

Os operadores de *crossover* implementados permitem gerar descendentes melhores que herdam as características de seus pais. Isso graças ao método de reprodução utilizado, no qual, a cada geração, é calculada a avaliação relativa de cada indivíduo da população, garantindo que indivíduos mais bem adaptados sejam reproduzidos na próxima geração.

Os experimentos mostram que a consistência é mantida quando ocorre a discretização. O cálculo da consistência é feito pela função de avaliação onde é considerada a discretização simultânea de todos os atributos de um SI considerando a classe a que pertencem. O método de discretização proposto classifica-se como multivariado e supervisionado.

Os resultados obtidos nas discretizações são melhores ou iguais aos métodos comparados. Uma das vantagens do método proposto é a possibilidade do usuário determinar o número máximo de intervalos criados pela discretização. A partir das discretizações obtidas, foi possível submeter os Sistemas de Informações às técnicas de extração de conhecimento existentes na TCA, obtendo redutos e regras de decisão consistentes.

As discretizações obtiveram informações importantes tanto para as regras quanto para os pontos de corte definidos. Um exemplo foi a discretização da base IEEE118. As faixas definidas são equivalentes as definidas pela norma técnica.

Um ponto fundamental no sucesso do AG é que este faz uma busca gerando uma "população" de possibilidades de discretizações em paralelo que evoluem para soluções consistentes. Assim, elas puderam fazer a busca explorando diferentes áreas do espaço de solução.

Por fim, o método proposto não se apresenta como substituto de nenhum dos métodos tradicionais de discretização, mas é uma alternativa robusta a ser considerada quando os métodos convencionais não apresentarem resultados satisfatórios.

7.1 TRABALHOS FUTUROS

A realização desse trabalho e os bons resultados obtidos sugerem a realização de diversos trabalhos futuros. Alguns deles são:

- Implementação de um AG para obter redutos de um SI discretizado.
- Estudo da utilização das discretizações geradas pelo AG para construção de sistemas Fuzzy aplicados a controle ou previsão de valores.
- Combinar os resultados obtidos a partir das discretizações com a lógica Paraconsistente.
- Melhorar a implementação explorando o paralelismo inerente dos AGs. O modelo de Ilhas poderia obter resultados melhores ainda. Esse modelo consiste em dividir a população em subpopulações. A cada geração efetua-se o intercâmbio de informações entre as subpopulações promovendo migração de

indivíduos ou simplesmente valores de aptidão. Nesse caso, seria implementado um Algoritmo Genético não Convencional (Fernandes, 2003). Esse modelo permitiria que Sistemas de Informação de maior porte pudessem ser discretizados utilizando populações com um número adequado de indivíduos.

- Utilizar outros métodos evolutivos de classificação, tais como: inteligência de enxames, colônia de formigas ou evolução diferencial.
- Tentar criar a discretização realizando agrupamentos de atributos.

REFERÊNCIAS

Asuncion, A., Newman, D. UCI repository of machine learning datasets. `http://archive.ics.uci.edu/ml`, 2007.

Bay, S. D. Multivariate discretization of continuous variables for set mining. In: ACM Sigkdd International Conference on Knowledge Discovery and Dada Mining, 6, Boston, 2000. Proceedings Boston: ACM, 2000. p.315-319.

Carvalho, J. C. S. Aplicação de Conjuntos Aproximados Utilizando Banco de Dados Relacionais. *Dissertação (Mestrado)* — EFEI, Itajubá, 2000.

Dai, J. A Genetic Algorithm for Discretazation of Decision Systems. In Third Conference on Machine Learning and Cybernetcs, Shanghai, 2004.

Dougherty J., Kohavi R. e Sahami M. Supervised and unsupervised discretization of continuous features. In International Conference on Machine Learning, pages 194-202, 1995.

Fayyad, U. e Irani, K.. Multi-interval Discretization of Continuousvalued Attributes for Classification Learning. In *Thirteenth International Joint Conference on Artificial Intelligence (IJCAI'1993)*, pag. 1022–1027. Morgan Kaufmann.

Fernandes, A. M. R. Inteligencia Artificial: Noções Gerais. Visual Books, Florianópolis, 2003.

GoldBerg, D. E. *Genetic Algorithms in Search, Optmization, and Machine Learning.* Addison-Wesley, 1989.

Gu, X, et al. Rough Set Based on Modified Chimerge Algorithm and its Aplication. *In Proceedings of the Fourth International Conference on Machine Learning and Cybernetcis*, Guangzhou, 2005.

Holland, J. H. Adaptation in natural and artificial systems (The University of Michigam Press, Ann Arbor, 1975), reprinted by MIT Press, 1992.

Kerber, R.. ChiMerge: Discretization of Numeric Attributes. In *Ninth National Conference on Artificial Intelligence (AAAI'1992)*, pag. 123-128. AAAI Press, 1992.

Liu, H., Hussain, F., Tan, C. L., e Dash, M. Discretization: An Enabling Technique. Data Mining and Knowledge Discovery, (6):393–423, 2002.

Matsuura, J. P. Discretização para Aprendizagem Bayesiana: Aplicação no Auxílio à Validade de dados em Proteção ao Vôo. *Dissertação (Mestrado)* – ITA, 2003.

Moraes, C. H. V. Aplicação de Técnicas Inteligentes no Auxílio à Operação dos Centros de Controle. *Dissertação (Doutorado)* — UNIFEI, Itajubá, 2006.

Pawlak, Z. – "Rough sets", *International Journal of Computer and Information Sciences*, Vol. 11, No. 5, pp. 341-356, 1982.

Pawlak, Z., Andrzej Skowron, Rudiments of rough sets, *International Journal of Computer and Information Sciences*, Vol. 177, pp. 3–27, 2007.

Pawlak, Z. Rough sets: theoretical aspects of reasoning about data. London, Kluwer, 1991

Pinto, C. M. S. Algoritmos Incrementais para Aprendizagem Bayesiana. *Dissertação (Mestrado)* — Universidade do Porto, 2005.

Quinlan, J. R., *C 4.5*: *Programs for Machine Learning*, Morgan Kaufmann, Los Altos, California, 1993

Rezende, S. O. Sistemas Inteligentes: Fundamentos e Aplicações. Ed. Manole, São Paulo, 2005.

Rosetta - Rough Set Toolkit for Analysis of Data. Disponível em: `http://www.lcb.uu.se/tools/rosetta/downloads.php`, Pesquisado em: 01/09/2010.

Russel, S. J. & P. Norvig. *Inteligência Artificial.* Ed. Campus, Rio de Janeiro, 2004.

Voltolini, R. F. e Monard, M. C. Discretização e geração de gráficos de dados em aprendizado de máquina. Dissertação de Mestrado, ICMC-USP, 2006.

Scott, D. W. On Optimal and Data-Based Histograms. *Biometrika*, 66(3):605–610, 1979

ANEXO I

A seguir é apresentada a consulta SQL para calcular o número de inconsistências utilizando o Coeficiente de Qualidade da TCA na discretização da Tabela 5.1. São definidos um ponto de corte para cada atributo criando 2 faixas (A e B) para cada um.

```
SELECT SUM(XCNT) as ERRO
FROM (
    SELECT  sum(CNT) as XCNT,count (*) as NOC
    FROM (
        SELECT  a1,a2, count (*) as CNT
        FROM (
            SELECT IIF ( a1<= 7.976,"A","B") AS a1,
                IIF ( a2<= 3.125,"A","B") AS a2,
                d
            FROM Exemplo
        ) GROUP BY a1,a2,d
    ) GROUP BY a1,a2
) WHERE NOC >1
```

O valor do erro será igual a 5 o que significa o número de inconsistências na aplicação dos pontos de corte para a1 igual a 7.976 e a2 igual a 3.125.

ANEXO II

A seguir é apresentada a consulta SQL para calcular o número de inconsistências utilizando Classe Majoritária na discretização da Tabela 5.1. São definidos um ponto de corte para cada atributo criando 2 faixas (A e B) para cada um.

```
SELECT SUM(XCNT – OCMAX) as Erro

FROM   (

            SELECT SUM(CNT) as XCNT, Max(CNT) as OCMAX

            FROM  (

                        SELECT a1,a2, count (*) as CNT

                        FROM  (

                                    SELECT IIF ( a1<=5.148,"A","B") AS a1,

                                                IIF ( a2<=6.194,"A","B") AS a2,

                                                d

                                    FROM  Exemplo

                        )  GROUP BY a1,a2,d

            )   GROUP BY a1,a2

)
```

O valor do erro será igual a 3 o que significa o número de inconsistências na aplicação dos pontos de corte para a1 igual a 5.148 e a2 igual a 6.194.

ANEXO III

A seguir é apresentados pontos de corte para a discretização da base Wdbc obtida pelo AG.

Reduto: {C2, C5, C9, C16, C20, C22, C24, C25, C29}

Atributo: C1
  * e <= 13,9536 então faixa = A
 >13,9536 e <= 15,8552 então faixa = B
 >15,8552 e <= 17,7568 então faixa = C
 >17,7568 e <= * então faixa = D

Atributo: C2
  * e <= 18,2853 então faixa = A
 >18,2853 e <= 29,5219 então faixa = B
 >29,5219 e <= 31,2961 então faixa = C
 >31,2961 e <= * então faixa = D

Atributo: C3
  * e <= 108,9095 então faixa = A
 >108,9095 e <= 176,9232 então faixa = B
 >176,9232 e <= 185,6058 então faixa = C
 >185,6058 e <= * então faixa = D

Atributo: C4
  * e <= 544,2750 então faixa = A
 >544,2750 e <= 709,3000 então faixa = B
 >709,3000 e <= 803,6000 então faixa = C
 >803,6000 e <= * então faixa = D

Atributo: C5
  * e <= 0,0892 então faixa = A
 >0,0892 e <= 0,0992 então faixa = B
 >0,0992 e <= 0,1069 então faixa = C
 >0,1069 e <= * então faixa = D

Atributo: C6
  * e <= 0,1074 então faixa = A
 >0,1074 e <= 0,2411 então faixa = B
 >0,2411 e <= 0,2606 então faixa = C
 >0,2606 e <= * então faixa = D

Atributo: C7
  * e <= 0,1024 então faixa = A
 >0,1024 e <= 0,3799 então faixa = B
 >0,3799 e <= 0,4225 então faixa = C
 >0,4225 e <= * então faixa = D

Atributo: C8
  * e <= 0,0161 então faixa = A
 >0,0161 e <= 0,0302 então faixa = B
 >0,0302 e <= 0,0483 então faixa = C
 >0,0483 e <= * então faixa = D

Atributo: C9
  * e <= 0,1713 então faixa = A
 >0,1713 e <= 0,1852 então faixa = B
 >0,1852 e <= 0,2030 então faixa = C
 >0,2030 e <= * então faixa = D

Atributo: C10
  * e <= 0,0628 então faixa = A
 >0,0628 e <= 0,0808 então faixa = B
 >0,0808 e <= 0,0851 então faixa = C
 >0,0851 e <= * então faixa = D

Atributo: C11
  * e <= 1,2989 então faixa = A
 >1,2989 e <= 2,5969 então faixa = B
 >2,5969 e <= 2,7625 então faixa = C
 >2,7625 e <= * então faixa = D

Atributo: C12
  * e <= 0,6317 então faixa = A
 >0,6317 e <= 1,0389 então faixa = B
 >1,0389 e <= 1,3104 então faixa = C
 >1,3104 e <= * então faixa = D

Atributo: C13
  * e <= 4,7894 então faixa = A
 >4,7894 e <= 9,2462 então faixa = B
 >9,2462 e <= 10,5196 então faixa = C
 >10,5196 e <= * então faixa = D

Atributo: C14
  * e <= 135,2975 então faixa = A
 >135,2975 e <= 354,8107 então faixa = B
 >354,8107 e <= 386,9346 então faixa = C
 >386,9346 e <= * então faixa = D

Atributo: C15
  * e <= 0,0135 então faixa = A
 >0,0135 e <= 0,0282 então faixa = B
 >0,0282 e <= 0,0302 então faixa = C
 >0,0302 e <= * então faixa = D

Atributo: C16
  * e <= 0,0085 então faixa = A

>0,0085 e <= 0,0179 então faixa = B
>0,0179 e <= 0,0252 então faixa = C
>0,0252 e <= * então faixa = D

Atributo: C17
* e <= 0,1228 então faixa = A
>0,1228 e <= 0,1624 então faixa = B
>0,1624 e <= 0,1861 então faixa = C
>0,1861 e <= * então faixa = D

Atributo: C18
* e <= 0,0116 então faixa = A
>0,0116 e <= 0,0327 então faixa = B
>0,0327 e <= 0,0380 então faixa = C
>0,0380 e <= * então faixa = D

Atributo: C19
* e <= 0,0207 então faixa = A
>0,0207 e <= 0,0704 então faixa = B
>0,0704 e <= 0,0768 então faixa = C
>0,0768 e <= * então faixa = D

Atributo: C20
* e <= 0,0018 então faixa = A
>0,0018 e <= 0,0047 então faixa = B
>0,0047 e <= 0,0073 então faixa = C
>0,0073 e <= * então faixa = D

Atributo: C21
* e <= 15,8008 então faixa = A
>15,8008 e <= 18,6118 então faixa = B
>18,6118 e <= 20,2984 então faixa = C
>20,2984 e <= * então faixa = D

Atributo: C22
* e <= 16,8976 então faixa = A
>16,8976 e <= 35,6576 então faixa = B
>35,6576 e <= 37,9088 então faixa = C
>37,9088 e <= * então faixa = D

Atributo: C23
* e <= 106,6312 então faixa = A
>106,6312 e <= 227,1052 então faixa = B
>227,1052 e <= 239,1526 então faixa = C
>239,1526 e <= * então faixa = D

Atributo: C24
* e <= 714,1440 então faixa = A
>714,1440 e <= 958,2720 então faixa = B
>958,2720 e <= 1121,0240 então faixa = C

>1121,0240 e <= * então faixa = D

Atributo: C25
* e <= 0,1151 então faixa = A
>0,1151 e <= 0,1287 então faixa = B
>0,1287 e <= 0,1378 então faixa = C
>0,1378 e <= * então faixa = D

Atributo: C26
* e <= 0,1613 então faixa = A
>0,1613 e <= 0,6766 então faixa = B
>0,6766 e <= 0,7488 então faixa = C
>0,7488 e <= * então faixa = D

Atributo: C27
* e <= 0,4758 então faixa = A
>0,4758 e <= 1,1018 então faixa = B
>1,1018 e <= 1,1894 então faixa = C
>1,1894 e <= * então faixa = D

Atributo: C28
* e <= 0,0116 então faixa = A
>0,0116 e <= 0,0290 então faixa = B
>0,0290 e <= 0,0581 então faixa = C
>0,0581 e <= * então faixa = D

Atributo: C29
* e <= 0,2224 então faixa = A
>0,2224 e <= 0,3391 então faixa = B
>0,3391 e <= 0,3848 então faixa = C
>0,3848 e <= * então faixa = D

Atributo: C30
* e <= 0,0749 então faixa = A
>0,0749 e <= 0,1480 então faixa = B
>0,1480 e <= 0,1618 então faixa = C
>0,1618 e <= * então faixa = D

ANEXO IV

A seguir é apresentados pontos de corte para a discretização da base Wine obtida pelo AG.

Reduto: {C01, C02, C03, C05, C06, C08, C10, C12, C13}

Limites das faixas dos atributos:

Atributo: C01
  * e <= 12,2080 então faixa = A
 >12,2080 e <= 13,2720 então faixa = B
 >13,2720 e <= * então faixa = C

Atributo: C02
  * e <= 1,7894 então faixa = A
 >1,7894 e <= 4,4129 então faixa = B
 >4,4129 e <= * então faixa = C

Atributo: C03
  * e <= 1,6218 então faixa = A
 >1,6218 e <= 3,0991 então faixa = B
 >3,0991 e <= * então faixa = C

Atributo: C04
  * e <= 15,2560 então faixa = A
 >15,2560 e <= 25,5380 então faixa = B
 >25,5380 e <= * então faixa = C

Atributo: C05
  * e <= 78,2800 então faixa = A
 >78,2800 e <= 98,5200 então faixa = B
 >98,5200 e <= * então faixa = C

Atributo: C06
  * e <= 1,5310 então faixa = A
 >1,5310 e <= 2,3430 então faixa = B
 >2,3430 e <= * então faixa = C

Atributo: C07
  * e <= 1,9516 então faixa = A
 >1,9516 e <= 3,4210 então faixa = B
 >3,4210 e <= * então faixa = C

Atributo: C08
  * e <= 0,4109 então faixa = A
 >0,4109 e <= 0,5593 então faixa = B
 >0,5593 e <= * então faixa = C

Atributo: C09

* e <= 0,9172 então faixa = A
>0,9172 e <= 3,2630 então faixa = B
>3,2630 e <= * então faixa = C

Atributo: C10
* e <= 4,3272 então faixa = A
>4,3272 e <= 11,1248 então faixa = B
>11,1248 e <= * então faixa = C

Atributo: C11
* e <= 0,5661 então faixa = A
>0,5661 e <= 0,8982 então faixa = B
>0,8982 e <= * então faixa = C

Atributo: C12
* e <= 1,8706 então faixa = A
>1,8706 e <= 2,7169 então faixa = B
>2,7169 e <= * então faixa = C

Atributo: C13
* e <= 852,8200 então faixa = A
>852,8200 e <= 1245,3800 então faixa = B
>1245,3800 e <= * então faixa = C