UNIVERSIDADE FEDERAL DE SÃO CARLOS

CENTRO DE CIÊNCIAS EXATAS E DE TECNOLOGIA

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

SISTEMAS *FUZZY* GENÉTICOS BASEADOS EM REGRAS: UM OPERADOR DE SELEÇÃO COM FOCO NA DIVERSIDADE DAS SOLUÇÕES

EDUARDO FERNANDO VELLUDO PRADO

ORIENTADOR: PROFA. DRA. HELOISA DE ARRUDA CAMARGO

São Carlos - SP

Fevereiro/2017

UNIVERSIDADE FEDERAL DE SÃO CARLOS

CENTRO DE CIÊNCIAS EXATAS E DE TECNOLOGIA

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

SISTEMAS *FUZZY* GENÉTICOS BASEADOS EM REGRAS: UM OPERADOR DE SELEÇÃO COM FOCO NA DIVERSIDADE DAS SOLUÇÕES

EDUARDO FERNANDO VELLUDO PRADO

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de São Carlos, como parte dos requisitos para a obtenção do título de Mestre em Ciência da Computação, área de concentração: Inteligência Artificial.

Orientador: **PROFA. DRA. HELOISA DE ARRUDA CAMARGO**

São Carlos - SP

Fevereiro/2017

UNIVERSIDADE FEDERAL DE SÃO CARLOS

Centro de Ciências Exatas e de Tecnologia
Programa de Pós-Graduação em Ciência da Computação

Folha de Aprovação

Assinaturas dos membros da comissão examinadora que avaliou e aprovou a Defesa de Dissertação de Mestrado do candidato Eduardo Fernando Velludo Prado, realizada em 17/02/2017:

Profa. Dra. Heloisa de Arruda Camargo
UFSCar

Prof. Dr. Ricardo Cerri
UFSCar

Prof. Dr. Adinovan Henriques de Macedo Pimenta
FATECE

A todos que me ajudaram de alguma forma

AGRADECIMENTO

Agradeço aos meus familiares e amigos por todo o apoio. Agradeço aos colegas de laboratório e em especial a minha orientadora Heloisa de Arruda Camargo por todo o suporte durante esse tempo de mestrado.

Obrigado a todos.

“Embora ninguém possa voltar atrás e fazer um novo começo, qualquer um pode começar agora e fazer um novo fim.”

Chico Xavier

“Começar de novo não necessariamente resulta em um novo fim”

Maria do Carmo Nicoletti

RESUMO

A geração e otimização automática de Sistemas *Fuzzy* é uma área muito importante, pois gerar e otimizar manualmente é uma tarefa muito trabalhosa, e pode exigir um ou mais especialistas. Mesmo com a ajuda de especialistas o trabalho pode ser impossível dependendo das variáveis envolvidas e da complexidade do problema. Por isso uma das propostas existentes para a realização deste trabalho é a utilização de sistemas inteligentes. Os Sistemas *Fuzzy* Genéticos Multiobjetivos são um exemplo de sistemas inteligentes que fazem o uso de Algoritmos Genéticos Multiobjetivos. Os Algoritmos Genéticos Multiobjetivos foram desenvolvidos com o propósito de gerar soluções que atendam a mais de um objetivo ao mesmo tempo. Cada Algoritmo Genético Multiobjetivo utiliza um método diferente para selecionar os indivíduos que serão usados para gerar uma nova população. Esse trabalho propõe um método de seleção como alternativa ao método utilizado pelo Algoritmo Genético Multiobjetivo (2+2) M-PAES. Esse método busca melhorar a dispersão das soluções, mas acabou melhorando a acurácia dos resultados e mantendo a interpretabilidade dos Sistemas *Fuzzy*.

Palavras-chave: Algoritmo Genético Multiobjetivo; Sistema *Fuzzy* Genético Multiobjetivo; Seleção de soluções; (2+2) M-PAES;

ABSTRACT

The automatic generation and optimization of Fuzzy Systems is an important area because the manually generation and optimization is very hard and may require one or more specialists. Even with the help of specialists the work may be impossible according to the variables involved and the complexity of the problem. One of many possibilities to help solve this problem is the use of Inteligent Systems. The Multiobjective Genetic Fuzzy Systems are one example of Intelligent Systems that use Multiobjetive Genetic Algorithm. The propose of the Multiobjetives Genetics Algorithms is generate solutions that satisfies more than one objective at once. The selection of the individuals that will generate the next offspring depends on the Multiobjetive Genetic Algorithm used, each one of them uses a different method. The propose of this work is to implement an alternative selection method for the Multiobjective Genetic Algorithm (2+2) M-PAES. This method aimed to improve the dispersion of the solutions but it has improved the accuracy of the results and has keeped the interpretability of the Fuzzy System.

Keywords: *Multiobjective Genetic Algorithm; Multiobjective Genetic Fuzzy Systems; Solutions Selection; (2+2)M-PAES.*

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

AE	Algoritmos Evolutivos
AG	Algoritmos Genéticos
AGMO	Algoritmo Genético Multiobjetivo
BC	Base de Conhecimento
BD	Base de Dados
BR	Base de Regras
CIG	*Computational Intelligence Group*
D.M.S	Distância Média entre Soluções
D.T.M	Distância Total Média
Inter.	Interpretabilidade
KEEL	*Knownlegde Extraction based on Evolutionary Learning*
Max.	Máximo
Min.	Mínimo
MOGA	*Multiobjective Genetic Algorithm*
N.Sol.	Número de Soluções
NPGA	*Niched Pareto Genetic Algorithm*
NSGA	*Nondominated Sorting Genetic Algorithm*
NSGA-II	*Nondominated Sorting Genetic Algorithm II*
PAES	Pareto *Archived Evolution Strategy*
PESA	*Pareto Envelope-based Selection Algorithm*
PESAII	*Pareto Envelope-based Selection Algorithm II*
Qu	Quartil
RWGA	*Random Weighted Genetic Algorithm*
SF	Sistemas *Fuzzy*
SFBR	Sistemas *Fuzzy* Baseados em Regras
SFGBR	Sistemas *Fuzzy* Genéticos Baseados em Regras
SFGMO	Sistema *Fuzzy* Genético Multiobjetivo
SPEA	*Strength Pareto Evolutionary Approach*
SPEA2	*Strength Pareto Evolutionary Approach 2*
UFSCar	Universidade Federal de São Carlos
WBGA	*Weight-based Genetic Algorithm*

SUMÁRIO

Capítulo 1

INTRODUÇÃO

1.1 Contextualização e Motivação

Sistemas Baseados em Regras são comumente encontrados na construção de modelos de inteligência artificial, particularmente em modelos *Fuzzy* (KIM; OH; PEDRYCZ, 2017).

A união entre os Sistemas Baseados em Regras com os modelos *Fuzzy* formam os Sistemas *Fuzzy* Baseados em Regras (SFBR). Os SFBR são um tipo de Sistemas *Fuzzy* (SF) usados com sucesso nas áreas de controle, classificação, mineração de dados, etc. Essas aplicações podem ser vistas, dentre outros, nos trabalhos de Caridá, Morandin e Tuma (2015), Riid e Preden (2017) e Riza et. al (2017).

Entretanto, a geração manual de um SFBR é trabalhosa, pois necessita-se de um ou mais especialistas no domínio de um dado problema que entendam sobre *Fuzzy*. Outro problema ocorre quando os especialistas podem não estar de acordo com os parâmetros definidos para a configuração de um sistema. Além disso, o tempo necessário para construção e configuração manual desse sistema pode ser alto de acordo com a complexidade do problema.

Tendo isso em vista, uma solução encontrada para a automatização da criação dos SFBR foi a utilização dos Algoritmos Evolutivos (AE). Os Algoritmos Genéticos (AG), que são uma classe dos algoritmos evolutivos, podem ser citados como exemplo de técnica para esse fim, como pode ser visto nos trabalhos de Cordón et al. (2004) e Elhag et al. (2015). A combinação entre os SFBR e os AG formam os chamados Sistemas *Fuzzy* Genéticos Baseados em Regras (SFGBR).

A aplicação dos AG nos SFBR corrobora com o aumento do desempenho desses sistemas na medida em que possibilitam a geração de bases de regras que sejam adequadas para a resolução de diversos problemas.

A aplicação dos AG nos SFBR gerara bons resultados como pode ser observado nos trabalhos de Cordón et al. (2004) e Abadeh, Mhamadi e Habibi (2011).

Todavia, os AG foram inicialmente concebidos para explorar o espaço de soluções na busca por apenas uma solução que fosse a melhor para resolver um problema. Especificamente nos SFGBR, a princípio, considerava-se a busca por soluções que apresentavam resultados voltados somente para o objetivo relacionado à acurácia. Segundo Fazzolari et al. (2013b), objetivos relacionados à interpretabilidade do sistema foram negligenciados até meados da década de 1990. Essa interpretabilidade é definida, segundo Gacto, Alcalá e Herrera (2011), como a capacidade de um sistema em expressar o seu comportamento de uma forma compreensível, sendo esse, um objetivo que depende da pessoa que faz a avaliação.

Para suprir as carências dos AG quando da aplicação em problemas que exigem mais de um objetivo surgiram então os Algoritmos Genéticos Multiobjetivos (AGMO) e junção desses nos SF formam os chamados SFGMO.

Geralmente, em SFGMO considera-se, para o processo de otimização, os objetivos de acurácia e interpretabilidade. Esses objetivos são conflitantes, ou seja, o aumento de um leva à diminuição do outro. Nesse caso, é obtido, não apenas uma, mas um conjunto de soluções com esses objetivos balanceados entre si.

Uma forma possível de lidar com objetivos conflitantes é com o uso da Técnica de Pareto. Essa técnica foi criada pelo economista Franco-Italiano Vilfredo Pareto em 1896 e possibilita, no contexto dos SFGMO, a otimização, em conjunto, dos objetivos acurácia e interpretabilidade. Essa otimização é realizada por intermédio de uma ponderação entre os objetivos, encontrando um conjunto de soluções. As melhores soluções ficam dispostas na chamada de Fronteira de Pareto.

Em alguns casos relacionados aos SFGMO, é necessário escolher uma ou mais soluções para a resolução dos problemas. Nesse sentido uma fronteira onde as soluções se encontram mais dispersas umas das outras possibilita uma melhor tomada de decisão.

Tendo isso em vista, esse trabalho tem como foco propiciar uma melhor tomada de decisão no processo de escolha das soluções em um problema multiobjetivo levando em consideração, para isso, a dispersão das soluções dispostas na Fronteira de Pareto. Essa melhor dispersão será realizada por intermédio de alterações no operador de seleção de uma versão do AGMO chamado de Pareto *Archived Evolution Strategy* (PAES).

1.2 Justificativa

Em um AGMO, a depender da forma que o operador de seleção é modelado, pode-se obter uma Fronteira de Pareto com maior ou menor dispersão. Essa dispersão impacta diretamente na tomada de decisão em relação à qual solução será a escolhida para a resolução de um dado problema. Isso se dá pelo fato de que, quanto menos dispersa a fronteira, mais difícil se torna a escolha de uma entre um conjunto de soluções que se encontram, por ventura, próximas umas das outras.

Tendo isso em vista, estudos vem sendo desenvolvidos e realizam avanços na forma com que a Fronteira de Pareto é obtida. Trabalhos como os de Zitzler, Laumanns e Thiele (2001) e Deb et al. (2002) podem ser citados como exemplo para esse fim.

Além disso, o laboratório CIG - *Computational Intelligence Group* - do Departamento de Computação da UFSCar – Universidade Federal de São Carlos realiza trabalhos voltados para melhorias e aplicações de SFGMO. Pode-se citar os trabalhos de Cintra, Monard e Camargo (2012) e Pimenta e Camargo (2015) entre outros.

1.3 Objetivo

Como objetivo geral pretende-se propor uma abordagem para propiciar uma melhor tomada de decisão no processo de escolha das soluções em um problema multiobjetivo a partir de um método que está pautado na dispersão das soluções dispostas na Fronteira de Pareto por intermédio da seleção direcionada de pais no algoritmo (2+2) M-PAES.

Como objetivos específicos tem-se:

- Implementação do (2+2) M-PAES proposto em Cococcioni et al. (2007);
- alteração da seleção de pais do (2+2) M-PAES;
- implementação de um SFBR;
- aplicação da métrica de dispersão de fronteira proposta em Deb et al. (2002);
- experimentos com a alteração proposta;
- comparação dos resultados dos testes.

1.4 Organização do Trabalho

Para mostrar a validação da hipótese e atingir o objetivo a dissertação está dividida da seguinte forma:

No Capítulo 2 é apresentada a fundamentação teórica com temas relacionados a Lógica *Fuzzy* e Algoritmos Genéticos e Algoritmos Genéticos Multiobjetivos.

O Capítulo 3 trata do estado da arte com apresentação de artigos relacionados a Sistemas *Fuzzy* Baseados em Regras, Algoritmos Genéticos Multiobjetivos na construção de Sistemas *Fuzzy*, Operadores Genéticos dentre outros.

No Capítulo 4 é apresentada a proposta de um sistema *Fuzzy* genético baseado em regras onde um operador de seleção é desenvolvido para prover diversidade das soluções encontradas.

No Capítulo 5 é apresentada a implementação da proposta destacado o algoritmo genético multiobjetivo modificado desenvolvido e os recursos utilizados.

No Capítulo 6 são apresentados os experimentos realizados e os resultados obtidos;

Por fim, no Capítulo 7 é apresentada a conclusão e as perspectivas para os trabalhos futuros

Capítulo 2

FUNDAMENTAÇÃO TEÓRICA

2.1 Sistemas *Fuzzy*

Sistemas *Fuzzy* são sistemas fundamentados na Lógica *Fuzzy*, que ao contrário da lógica clássica, conseguem representar informações incertas e imprecisas utilizando a teoria de conjuntos *Fuzzy*. A necessidade de representações incertas e imprecisas se dá ao fato de muitas vezes não haver modelos matemáticos precisos para descrição de problemas ou do modelo existente possuir um alto custo computacional.

A seguir serão explicados brevemente alguns dos termos que serão utilizados no decorrer deste capítulo. Para um aprofundamento é aconselhável a consulta de trabalhos específicos como Pedrycz e Gomide (1998) e Nicoletti (2004).

A teoria dos conjuntos *Fuzzy* permite que um elemento esteja classificado como pertencendo a mais de um conjunto simultaneamente de acordo com um grau. Esse grau é conhecido como pertinência, e é calculado de acordo com a função de pertinência, e o conjunto que permite essa caracterização em graus é conhecido como conjunto *Fuzzy*. Outros dois termos importantes são o de variáveis linguísticas e termos linguísticos. Variável linguística é uma variável que recebe um valor na linguagem natural, como por exemplo, cor, peso, distância, temperatura. E termos linguísticos são o rótulo que também são expressos em linguagem natural por exemplo, longe, leve, pesado.

2.2 Sistemas *Fuzzy* Baseado em Regras

Sistemas *Fuzzy* Baseados em Regras (SFBRs) são o tipo de SF sobre o qual o algoritmo da proposta funciona, e por isso serão explicados a seguir importantes termos e funcionamentos deles.

Os SFBRs possuem dois componentes, um mecanismo de inferência e uma base de conhecimento (BC). O mecanismo de inferência recebe valores de entrada que são inferidos em valores de saída, já a BC pode ser entendida contendo duas partes, a base de regras (BR) e a base de dados (BD).

A base de regras é constituída de regras *Fuzzy* que são utilizadas no processo de inferência. Essas regras possuem o formato a seguir:

Se <antecedente> então <consequente> (2.1)

O número de antecedentes e consequentes pode variar de acordo com a regra e eles são formados com as informações guardadas na BC.

A base de dados possui os conjuntos *Fuzzy*, os termos e variáveis linguísticos. Uma regra *Fuzzy* expressa com termos linguísticos X_{n} e variáveis linguísticas $Y_{\mathrm{n/n}}$ onde n é o índice que distingue os diferentes termos e variáveis possíveis é representada da seguinte forma:

$$SE \quad X_1 \text{ é } Y_{1/1} \quad E \quad X_2 \text{ é } Y_{2,2} \quad ENT\tilde{A}O \quad X_{\mathrm{n}} \text{ é } Y_{\mathrm{n/n}} \qquad (2.2)$$

A Figura 1 representa um SFBRs com os componentes descritos acima.

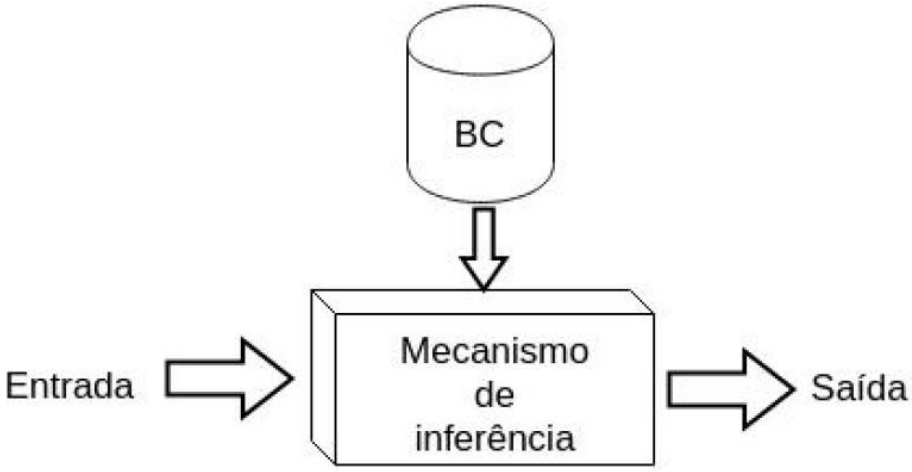

Figura 1 - Representação de um SFBRs

2.3 Algoritmos Genéticos

Os algoritmos genéticos (AG) foram inventados por John Holland nos anos 60 e desenvolvidos por ele e seus estudantes na *University of Michigan* nos anos 60 e 70. O motivo da criação do AG, segundo Mitchell e Melanie (1996), é estudar o fenômeno de adaptação que ocorre na natureza e importa-lo para sistemas computacionais.

Para representação desse fenômeno de adaptação, muitos termos foram importados para os sistemas computacionais (cromossomo, população, indivíduo, gene, *fitness*, entre outros). Um cromossomo pode ser descrito como um vetor em que cada célula é um gene e pode representar uma característica. Cada cromossomo representa um indivíduo e o conjunto de indivíduos forma uma população. O *Fitness* é uma função que representa o quanto o indivíduo é adaptado ao meio, ou seja, quão bem ele soluciona um problema. A forma mais simples de AG utiliza 3 operadores:

- Seleção – Operador que seleciona os indivíduos de uma população que irão compor uma nova população.
- Cruzamento – Realiza o cruzamento entre os indivíduos.
- Mutação – Modifica aleatoriamente genes de indivíduos.

Segundo Mitchell e Melanie (1996), um AG simples possui o seguinte ciclo:

Algorithm 1: Algoritmo Genético

```
CrossoverProbabilidade ← 0.5;
MutaçãoProbabilidade ← 0.5;
População ← geraPopulaçãoAleatória();
while Condição do
    atribuiFitness(População);
    População ← selecionaIndivíduos(População);
    if (CrossoverProbabilidade < probabilidadeAleatória()) then
        População ← crossover(População);
    if (MutaçãoProbabilidade < probabilidadeAteatória()) then
        População ← mutação(População);
    Condição ← atualizaCondição(Condição);
```

2.4 Algoritmos Genéticos Multiobjetivo

Algoritmos Genéticos Multiobjetivo (AGMO) surgiram de uma necessidade de resolver problemas na área de otimização multiobjetivo. Nessa área há problemas que não possuem uma solução ótima e sim um conjunto de soluções que possuem diferentes níveis de satisfação para cada objetivo que se quer atingir. Uma característica dos problemas multiobjetivos é a existência de objetivos conflitantes, isto é, a medida em que um objetivo é satisfeito, outro objetivo passa a obter piores níveis de satisfação (FONSECA; FLEMING, 1993). Quando um objetivo de uma solução é melhorado e simultaneamente outro objetivo é piorado, dizemos que há um *trade-off*. Para compreender melhor o que é um problema, os objetivos, soluções e *trade-off*, podemos usar como exemplo a escolha de um novo celular.

Existem diversos tipos de celulares com especificações técnicas, valores e funcionalidades diferentes. Para esse exemplo digamos que uma pessoa deseja comprar um celular barato e que possua um ótimo desempenho. Nesse caso o problema é definido como qual celular comprar. Os objetivos são, ser um celular barato e ter um bom desempenho. E como quanto mais caro um celular maior o desempenho podemos dizer que há um *trade-off* na relação desses objetivos.

Um outro termo importante da otimização multiobjetivo é a Fronteira de Pareto. Fronteira de Pareto é o melhor limite possível do equilíbrio entre os diversos objetivos. A busca por essa fronteira nos AGMO é caracterizada por um aperfeiçoamento das soluções não dominadas. Em AGMO dizemos que uma solução domina a outra quando todos os valores de objetivos de uma solução são melhores que todos os valores de outra.

A Figura 2 representa o problema da escolha do celular descrito acima.

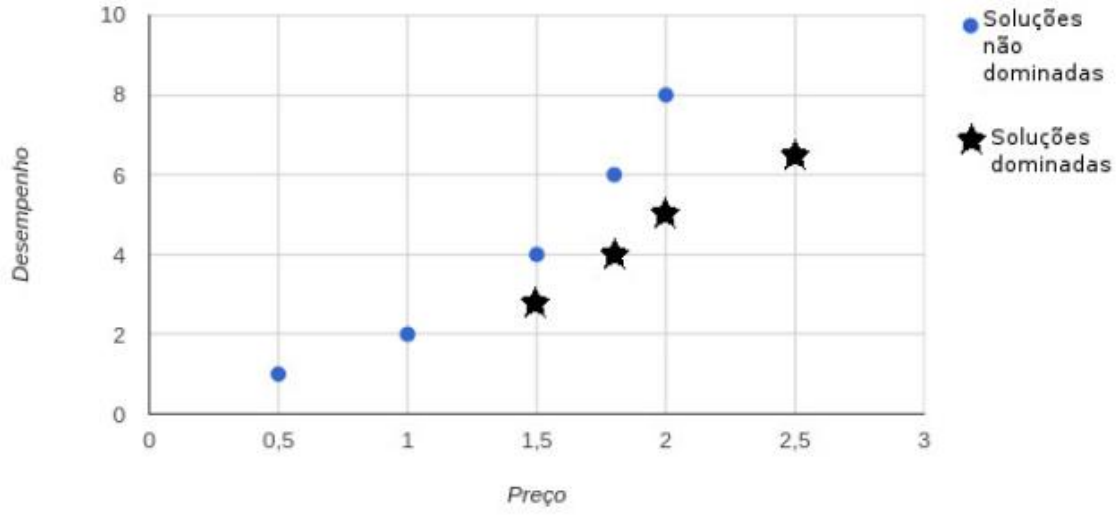

Figura 2 - Representação de um problema multiobjetivo

Na representação os eixos se referem aos dois objetivos e as soluções são os celulares. Se for observado as soluções que possuem o valor de preço 1,5, pode-se notar que mesmo tendo o mesmo preço há uma solução que possui o melhor desempenho, assim, pode-se dizer que a solução que possui o melhor desempenho domina a outra que também possui o valor de 1,5 mas que tem um desempenho inferior. A mesma comparação vale para as soluções que possuem o desempenho 4 e preços diferentes.

Segundo Zitzler e Thiele (1998), inicialmente eram escolhidos AG específicos e feito as alterações necessárias, como iterar as melhores soluções individualmente buscando melhoras, mas isso acabava gerando novos AG específicos.

De acordo com Deb e Kalyanmoy (2001), o primeiro AG capaz de encontrar soluções diversificadas, com objetivos conflitantes, sem que fosse necessária a otimização objetivo a objetivo, foi proposto por Schaffer (1985). Em Schaffer (1985), é proposta uma adaptação simples a um AG que era usado para tratar um problema com um só objetivo, mas trabalho não

recebeu muita atenção durante uma década pois os resultados acabavam sempre convergindo para um único ponto.

Após isso, começaram a surgir diversos AGMO, sendo que alguns serão explicados brevemente nas próximas seções. Na lista abaixo, os quatro primeiros itens são os que vão ser explicados.

· *Approximating the Nondominated Front Using the Pareto Archived Evolution Strategy* (PAES) (CORNE; KNOWLES; OATES, 2000)

· *An Evolutionary Algorithm for Multiobjective Optimization: The Strength Pareto Approach* (SPEA) (ZITZLER; THIELE, 1998)

· *SPEA2: Improving the Strength Pareto Evolutionary Algorithm* (SPEA2) (ZITZLER; LAUMANNS; THIELE, 2001)

· *A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II* (NSGA-II) (DEB et al., 2002)

· *Multi-objective Genetic Algorithm (MOGA)* (FONSECA; FLEMING, 1993)

· *Niched Pareto Genetic Algorithm* (NPGA) (HORN; NAFPLIOTIS; GOLDBERG, 2002)

· *Weight-based Genetic Algorithm* (WBGA) (HAJELA; LIN, 1992)

· *Random Weighted Genetic Algorithm* (RWGA) (MURATA; ISHIBUCHI, 1995)

· *Nondominated Sorting Genetic Algorithm* (NSGA) (SRINIVAS; DEB, 1994)

· *Pareto Envelope-based Selection Algorithm* (PESA)(ZITZLER; LAUMANNS; THIELE, 2001)

· *Region-based Selection in Evolutionary Multiobjective Optimization* (PESA-II) (CORNE et al., 2001)

Os AGMOs são divididos em primeira e segunda geração, essa divisão serve para assinalar se o algoritmo faz ou não o uso do conceito de elitismo. Utilizar o conceito de elitismo significa que o algoritmo possui meios de manter as melhores soluções separadas para que não se percam no passar das gerações. Um dos meios de se fazer isso é tendo um arquivo que guarda

as soluções que serão utilizadas para criar novas gerações. As próximas subseções vão apresentar os 4 primeiros algoritmos da lista de AGMO que foi mostrada anteriormente.

2.4.1 *Pareto Archived Evolution Strategy* (PAES)

O PAES é um algoritmo criado inicialmente com o intuito de resolver um problema de roteamento da área de telecomunicações, como obteve bons resultados, especialmente relativos a desempenho, ficou sendo um dos algoritmos importantes na área dos AGMO (KNOWLES; CORNE, 2000).

Como o algoritmo em sua forma mais simples (1+1) não possui fase de seleção, operador de crossover, torneio binário, isso faz dele um algoritmo rápido e agressivo (rápida convergência). Segundo os autores, esse algoritmo pode representar a forma mais simples e não trivial de se obter soluções próximas da fronteira de Pareto.

Serão apresentados na sequencia o algoritmo inicial e algumas de suas variações.

2.4.1.1 (1 + 1)-PAES

A expressão 1 + 1 indica que a partir de uma solução é gerada apenas uma outra solução. O algoritmo pode ser separado em 3 fases:

1. Geração de nova solução – Fase onde a nova solução é gerada;
2. Comparação e aceitação – Ocorre a comparação e aceitação da solução;
3. Arquivamento da solução – A solução é arquivada.

A fase geração de nova solução começa com a geração de uma solução aleatória i, que é colocada no arquivo de soluções não dominadas e considerada a solução atual. Esse arquivo possui um tamanho fixo e é configurado anteriormente. Após esse início a nova solução é gerada através de uma mutação na solução atual i para obter uma nova solução m, assim acaba a fase de geração de soluções e inicia a segunda fase.

Nessa fase, a solução m é comparada com a i e 3 possíveis cenários podem acontecer:

1. i dominar m – A solução m é descartada;

2. m dominar i – A solução i é descartada e o algoritmo segue para terceira parte;

3. não se dominarem – Nenhuma é descartada e o algoritmo segue para terceira fase de inserção no arquivo.

A terceira e última fase é iniciada verificando a dominância entre a solução candidata e as soluções do arquivo. Se a solução candidata for dominada por alguma do arquivo, ela é descartada e o algoritmo volta a primeira fase. Se a solução candidata domina alguma do arquivo, a solução do arquivo é removida e é inserida a solução candidata. Se elas não se dominarem e houver espaço a solução candidata é inserida e por último se elas não se dominarem e não houver espaço no arquivo é feita uma verificação de densidade. A verificação de densidade é feita utilizando um algoritmo chamado *Adaptive Grid*. Este algoritmo cria uma malha de n dimensões em que n é o número de objetivos e o alcance de cada eixo depende do domínio de cada objetivo. Os eixos dos objetivos são também bisseccionados. Cada solução então é inserida nesta malha de acordo o valor obtido em cada objetivo e a partir da quantidade de soluções por setor da malha é calculada a densidade da malha.

Algorithm 2: Testes de dominância

```
Input: Recebe a SoluçãoCandidata
Output: Configura a nova SoluçãoAtual
if SoluçãoCandidata domina SoluçãoAtual then
    SoluçãoAtual ← SoluçãoCandidata
    AdicionarAoArquivo(SoluçãoCandidata)
else
    if SolucãoCandidata e SoluçãoAtual não se dominam then
        AdicionarAoArquivo(SoluçãoCandidata)
        SoluçãoAtual ← ProcuraNovaSoluçãoAtual(SoluçãoAtual,
        SoluçãoCandidata, Arquivo)
```

Algorithm 3: AdicionarAoArquivo

```
Input: Recebe a SoluçãoCandidata
if SoluçãoCandidata domina alguma Solução do arquivo then
    RemoverSoluçãoDominada(Arquivo)
    Adicionar(SoluçãoCandidata)
else
    if Solução do arquivo domina SoluçãoCandidata then
        return False
    else
        if Se arquivo está cheio then
            Elimina Solução em área de maior densidade
        Adicionar(SoluçãoCandidata)
```

2.4.1.2 (1+λ)-PAES

A expressão 1 + λ indica que a partir de uma solução i atual ocorrem mutações que resultam em λ novas soluções. Gerar λ soluções acarreta na maior dificuldade de escolher qual solução é aceita como a nova solução i atual. Para resolver isso as soluções recebem um valor de *fitness* de acordo com a relação de dominância e densidade no grid.

Se a mutação domina alguma solução do arquivo, então ela recebe +1 para cada solução dominada e se ela é dominada por alguma recebe -1. Ao final a que possui maior valor é a escolhida e se houver mais de uma com o melhor valor a que se encontraria em uma região menos densa é a escolhida. O restante da fase de arquivamento da solução acontece como no (1 + 1)-PAES.

2.4.1.3 (μ+λ)-PAES

A letra grega μ indica a quantidade de soluções que são usadas inicialmente para gerar as λ mutações. As μ soluções são selecionadas do arquivo via torneio binário usando os valores de *fitness* que são obtidos da mesma maneira explicada no (1 + λ)-PAES. O restante do algoritmo segue como o anterior.

2.4.2 *Strong Pareto Evolutive Algorithm* (SPEA)

O SPEA foi desenvolvido por Zitzler e Thiele (1998) com a intenção de integrar as melhores técnicas de AEs existentes em um só algoritmo. De acordo com Zitzler e Thiele (1998), os AEs traziam melhores resultados que as técnicas existentes de busca cega, mas havia o problema de existirem muitos AEs, cada um com uma característica diferente. Em consequência disso, era difícil localizar o algoritmo que melhor solucionava os problemas, e muitas vezes o resultado final gerava um novo AE específico.

O algoritmo funciona partindo de uma população inicial que pode ser dada ou gerada aleatoriamente e com o arquivo vazio. A partir da população inicial as soluções não dominadas são coletadas e adicionadas a um arquivo externo que possui o tamanho especificado anteriormente. São aplicados métodos de redução desse arquivo externo com a finalidade de eliminar soluções duplicadas e, também soluções não dominadas com o auxílio de uma técnica de agrupamento. O próximo passo é selecionar as soluções que vão sofrer a mutação e o crossover. Isto é feito utilizando torneio binário, que usa a função de *fitness* como parâmetro. É importante observar que as soluções do arquivo têm mais chances de serem selecionadas que as soluções da população, pois irão possuir melhor valor de *fitness*. Depois de selecionada a população, ocorrem operações de crossover e mutações para se obter a população da nova geração (Figura 3).

O cálculo de *fitness* é feito de maneira diferente para as soluções não dominadas (arquivo) e da população restante. Para o cálculo de *fitness* das soluções do arquivo, considere a uma solução do arquivo, s o número de soluções da população que a domina e n a quantidade de soluções da população. O *fitness* de a pode ser expresso com a fórmula 2.3.

$$fitness(a) = \frac{\mathrm{S}}{\mathrm{n}+1} \qquad (2.3)$$

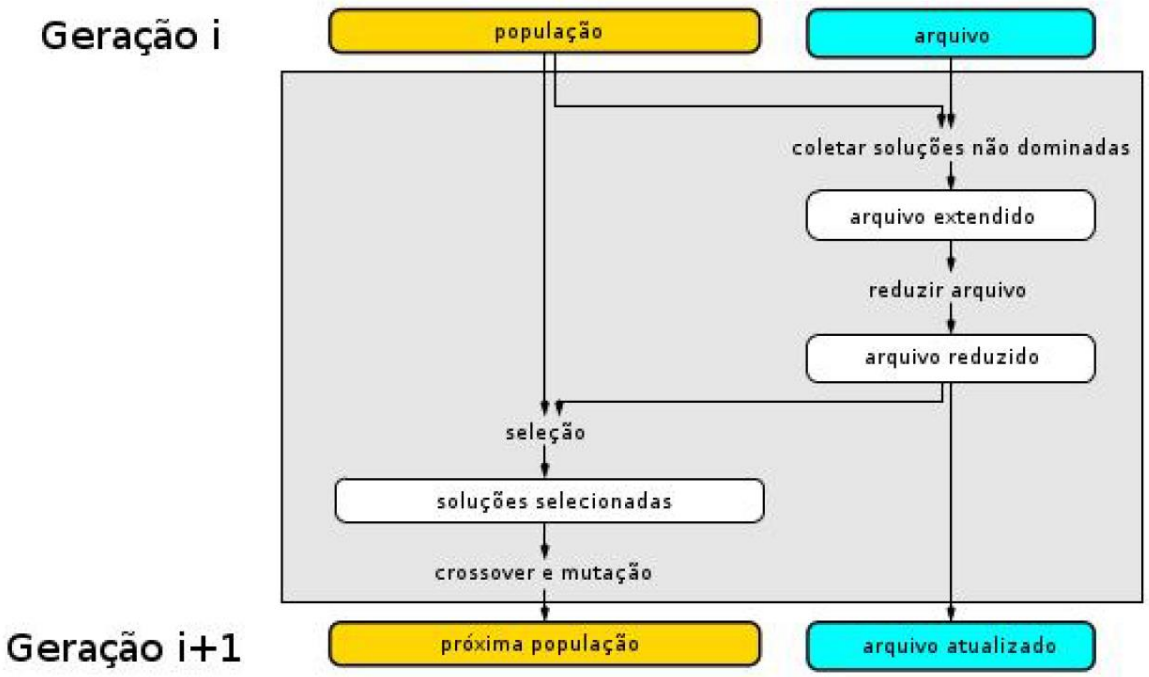

Figura 3 - Principal iteração do algoritmo SPEA. Adaptação de Zitzler e Thiele (1998)

Ou seja, o *fitness* de uma solução do arquivo é a quantidade de soluções da população que ela domina, dividido pelo número de indivíduos da população acrescentado em 1.

Para o cálculo de *fitness* de soluções p da população separaremos em 3 passos.

1. Somar os s das soluções a que o dominam,
2. Acrescentar a soma o valor de n acrescido à um,
3. Dividir a soma por n acrescido à um.

Para exemplificar o cálculo de *fitness*, observe a Figura 4.

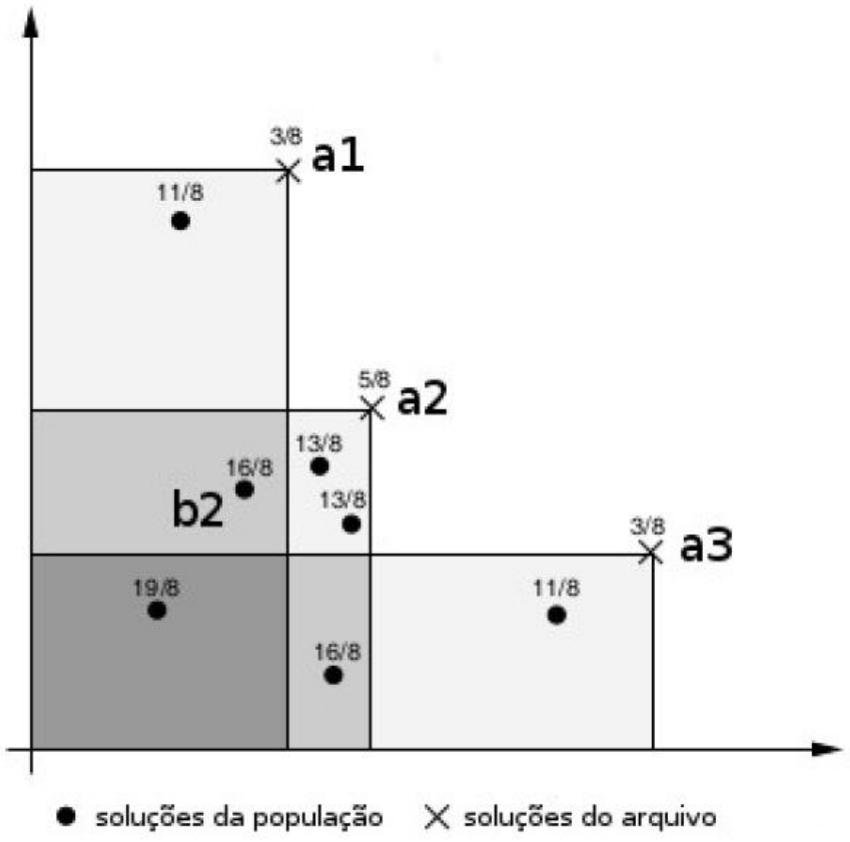

Figura 4 - Cálculo do *fitness* no SPEA. Adaptação de Zitzler e Thiele (1998)

Da equação 2.3 e da Figura 4, para a1 s = 3 (domina 3 soluções) e n = 7 portanto o *fitness* de a1 é 3/8.

Para fazer o cálculo de *fitness* do b2, seguindo os passos:

1. A soma do s de a1 com o s de a2 é 8,

2. 8 + (n + 1) = 16,

3. 16/(n + 1) = 16/8.

A técnica de agrupamento utilizada no SPEA, que foi citada anteriormente, é chamada de "*average linkage method*". Ela funciona inicialmente considerando cada solução do arquivo como um grupo. Depois disso, a cada iteração, dois a dois os grupos são juntados até que um número dado de grupo seja obtido. Essas junções são feitas levando em consideração o vizinho mais próximo. Caso seja necessário juntar dois grupos que possuem mais de um elemento, a distância é a média de distância entre duas soluções de cada grupo. O novo arquivo é escolhido

obtendo os centroides de cada grupo. O centroide é a solução com a menor média de distância entre ela e as outras soluções do grupo.

2.4.3 *Strong Pareto Evolutive Algorithm 2* (SPEA2)

SPEA2 foi sugerido por Zitzler, Laumanns e Thiele (2001) com o principal objetivo de atualizar o SPEA, descrito na seção 2.3.2. Ele leva em consideração novos estudos na área de AGMO que mostraram quais as principais características responsáveis pelo sucesso dos AGMO (LAUMANNS; ZITZLER; THIELE, 2001).

As principais diferenças entre o SPEA2 e o SPEA são:

- Mudança no método de calcular o *fitness* das soluções;
- Adição de densidade do vizinho mais próximo;
- Novos métodos de montagem do arquivo.

Essas mudanças ocorreram, pois, em cada um dos pontos citados foram encontrados pontos fracos. O novo cálculo de *fitness* tenta resolver o problema que acontece se o arquivo do SPEA conter apenas 1 solução e todas as outras soluções receberem o mesmo valor de *fitness*. Isso levaria o algoritmo a agir como algoritmos de busca aleatórios.

A adição de um método de densidade que considera a proximidade de vizinhos foi incorporada à população. Já existia um método de agrupamento que era usado no arquivo que diminuía e orientava a busca da fronteira de Pareto usando uma função de densidade. Porém, o mesmo não ocorria com os indivíduos da população que, quando não se dominavam, dificultava a classificação dos mesmos.

As últimas melhoras feitas pelo SPEA2 foram em relação à montagem do arquivo. Uma delas foi em como é escolhida a solução a ser excluída caso o arquivo esteja cheio, pois quando era selecionada a média dos grupos obtidos, algumas soluções que contribuíam para a dispersão na fronteira de Pareto eram descartadas.

Outra mudança na montagem do arquivo foi a adição de soluções dominadas para completar o tamanho de soluções possíveis. No algoritmo SPEA só eram adicionados ao arquivo e mantidas nele as soluções que não eram dominadas.

Além de todas as alterações feitas em pontos considerados fracos do SPEA, foi alterada, também, a maneira que as soluções são selecionadas para que ocorra o crossover e mutação. No SPEA2 são utilizadas somente as soluções do arquivo para favorecer as soluções não dominadas.

O algoritmo SPEA2 funciona em 6 passos, inicialização, atribuição do *fitness*, seleção do arquivo, término, seleção de indivíduos, recombinação e mutação.

A inicialização é feita gerando uma população inicial e o arquivo vazio.

A atribuição do *fitness* é feita nas soluções da população inicial e na população do arquivo. Ela pode ser dividida em duas partes, sendo que na primeira são calculadas as relações de dominância entre as soluções e na segunda é calculada o grau de densidade que as soluções possuem.

Para o cálculo da primeira parte, as soluções não dominadas recebem o valor 0 e as soluções dominadas recebem um valor calculado dependendo do número de soluções que elas dominam e que são dominadas por ela. Para calcular o *fitness* de uma solução dominada s, consideremos d a quantidade de soluções que a dominam, n a quantidade de soluções dominadas, e m o maior valor de *fitness* dentre as soluções que dominam s. Sendo assim, a seguinte fórmula mostra como é feito a primeira parte do cálculo de *fitness*:

$$P1 = (d + n - 1) + m \qquad (2.4)$$

A única observação a ser feita é que se m for igual a 0, então m recebe o valor 1. Para um exemplo prático a Figura 5 é observada.

Figura 5 - Cálculo do *fitness* SPEA2. Adaptação de Zitzler, Laumanns e Thiele (2001)

Para se calcular o valor de *fitness* da solução S1, é necessário contar a quantidade de soluções que ela domina (d = 3), somar com a quantidade de soluções que a dominam (n = 2), subtrair 1 e em seguida adicionar o maior valor de *fitness* de uma solução que a domina (m = 5). Assim obtemos o valor 9 de *fitness* para s1, e termina a primeira parte do cálculo.

A segunda parte, responsável por medir a densidade, utiliza uma técnica adaptada de Silverman (1996). A densidade de um certo ponto i é 1 dividida pela distância da solução mais próxima de i aumentada em 2. Considerando a distância da solução mais próxima sendo s, podemos calcular a segunda parte de acordo com a equação 2.5.

$$\mathrm{P2} = \frac{1}{\mathrm{s} + 2} \qquad (2.5)$$

Encerrando o passo de atribuição de *fitness*, basta que se some os valores obtidos na primeira parte com os valores obtidos na segunda parte para encontrar o valor final do *fitness*.

$$Final = P1 + P2 \tag{2.6}$$

A seleção do arquivo consiste em preencher o novo arquivo com soluções não dominadas da população e do arquivo anterior. No momento da inserção podem ocorrer 2 casos que precisam de 2 soluções diferentes.

1. As soluções não dominadas a serem inseridas são insuficientes para preencher completamente o novo arquivo. Nesse caso o arquivo é completado com soluções dominadas que são escolhidas de acordo com o valor de *fitness* delas.

2. As soluções não dominadas a serem inseridas ultrapassam o número de soluções máximas suportadas pelo arquivo. Quando isso ocorre, é necessário procurar quais são as duas soluções mais próximas uma da outra. Em seguida procura-se quais soluções são as mais próximas de ambas, e a que tiver a solução mais próxima de si é eliminada.

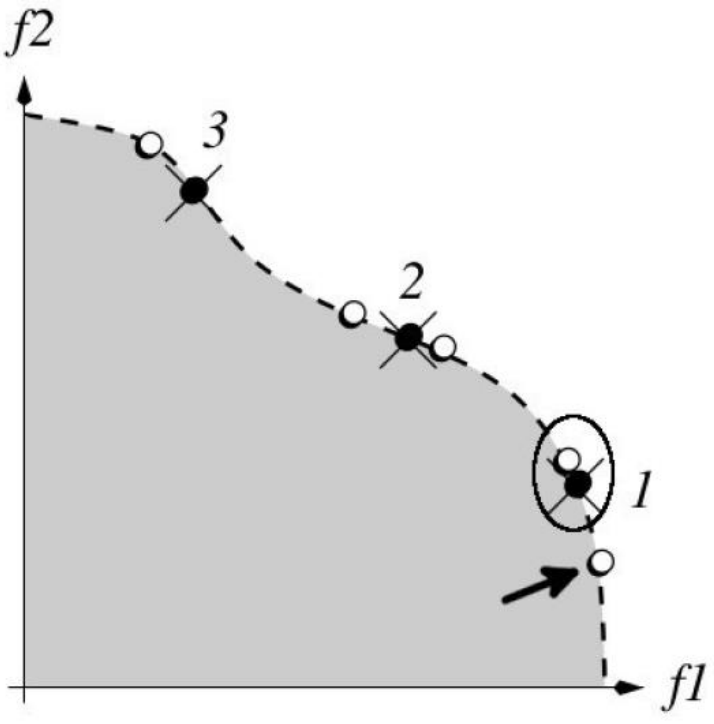

Figura 6 - Remoção de soluções no SPEA2. Adaptação de Zitzler, Laumanns e Thiele (2001)

A elipse representa as duas soluções mais próximas e a seta indica a solução que está mais perto de ambas. A solução dentro da elipse marcada com o X foi eliminada pois estava mais perto da solução mais perto de ambas soluções. Pode acontecer de a solução mais próxima

de ambas ter a mesma distância de ambas, caso isso aconteça a solução seguinte mais próxima de ambas é selecionada para as comparações.

O passo denominado de **término**, termina o algoritmo caso as condições de término sejam satisfeitas. As condições podem ser, depois tem um dado tempo de execução, obtenção de resultados satisfatórios, alcance do número de iterações máximo, ou outra de acordo com o objetivo buscado.

O passo de seleção de soluções para compor a formação de uma nova população acontece por meio de um torneiro binário, que leva em consideração o valor de *fitness* de cada solução.

Após o passo de seleção, segue-se com o passo da mutação e recombinação, que é o último passo da iteração principal do algoritmo. Ele consiste em compor uma nova população com as soluções adquiridas no passo anterior e com as soluções do arquivo. As soluções do arquivo são recombinadas e mutacionadas com as soluções selecionadas.

2.4.4 *Nondominated Sorting Genetic Algorithm II* (NSGA-II)

Assim como o SPEA2 veio para melhorar alguns aspectos do SPEA, o NSGA-II Deb et al. (2002) veio para atualizar o seu antecessor NSGA Srinivas e Deb (1994) com as novas descobertas na área de AGMO. (Laumanns, Zitzler e Thiele (2001).

De acordo com Deb et al. (2002) o algoritmo que inspirou o NSGA-II sofria críticas em relação à 3 principais características:

1. Alta complexidade – A complexidade do mecanismo de organização das soluções não dominadas era de O (MN 3) (onde M é o número de objetivos e N o tamanho da população);

2. Falta de elitismo – A falta de elitismo atrasava a convergência de soluções e provocava a perda de soluções que contribuíam para a diversificação de resultados.

3. Necessidade de especificar um parâmetro de compartilhamento – Esse parâmetro era responsável por aumentar a diversidade de soluções e de acordo com o autor era desejável diminuir o número de parâmetros.

Para organizar as soluções e usá-las posteriormente, o algoritmo utiliza o conceito de fronteira. Ele organiza as fronteiras de acordo com a relação de dominância entre as soluções, e busca também diminuir a complexidade do algoritmo NSGA.

Para diminuir a complexidade do algoritmo de $O(MN^3)$ para $O(MN^2)$, o autor cria 2 conjuntos para cada solução, um é chamado de n_p, e guarda a quantidade de soluções que dominam p, e outra de S_p, que é o conjunto de soluções dominadas por p. O algoritmo então itera uma primeira vez para as N, soluções preenchendo ambas as variáveis criadas. As que possuem n_p igual a 0 são colocadas em um vetor Q. Depois disso, inicia a iteração no vetor Q, e para cada solução s do vetor Q ele diminui do número das soluções S_p em 1.

Isso gera novas soluções que possuem n_p iguais a zero, que por sua vez são colocadas em Q. O loop é repetido $N - 1$ vezes, ou seja, até que todas as soluções em Q sejam verificadas e classificadas em fronteiras de acordo com a dominância. Uma representação de como é classificada as fronteiras de acordo com os passos explicados anteriormente pode ser observada na Figura 7.

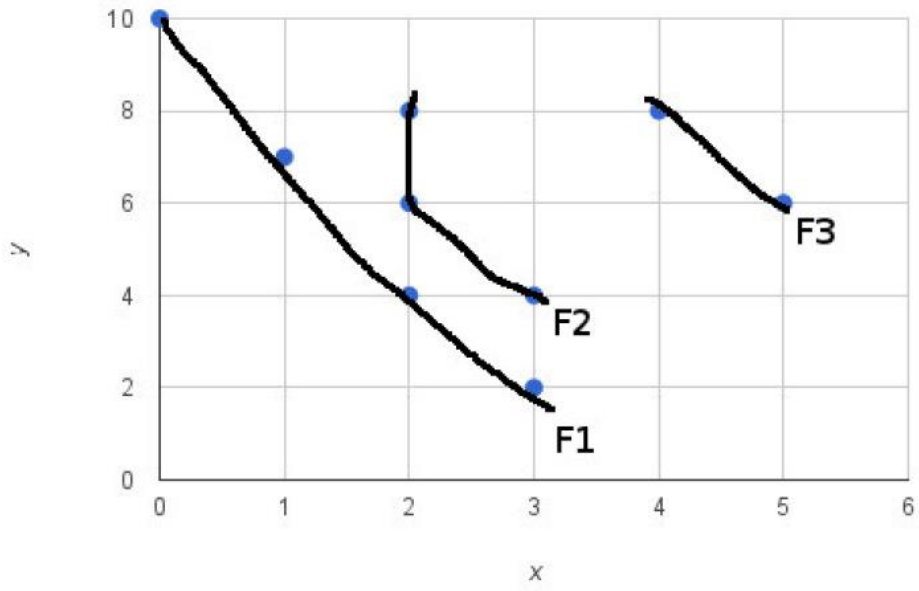

Figura 7 - Representação da classificação do NSGA-II em fronteiras

X e Y são os objetivos, os pontos são as soluções e a linhas em preto são um esboço das fronteiras F1, F2 e F3.

Para resolver o problema de precisar configurar um parâmetro de compartilhamento, foi proposto um operador que estima a densidade, e posteriormente é utilizado na classificação das soluções. O parâmetro de compartilhamento está relacionado com o tipo de métrica escolhida para se calcular a distância entre dois indivíduos da população. O cálculo de densidade é feito escolhendo uma solução i, e calculando a distância média de seus vizinhos no mesmo objetivo. Tendo os vizinhos de i como vértice é possível formar um cuboide que terá como perímetro a média de distância de i com seus vizinhos de cada objetivo. Quanto menor for o valor do perímetro do cuboide, mais densa é a região que ele está. Para uma representação do cuboide graficamente, tem-se a Figura 8.

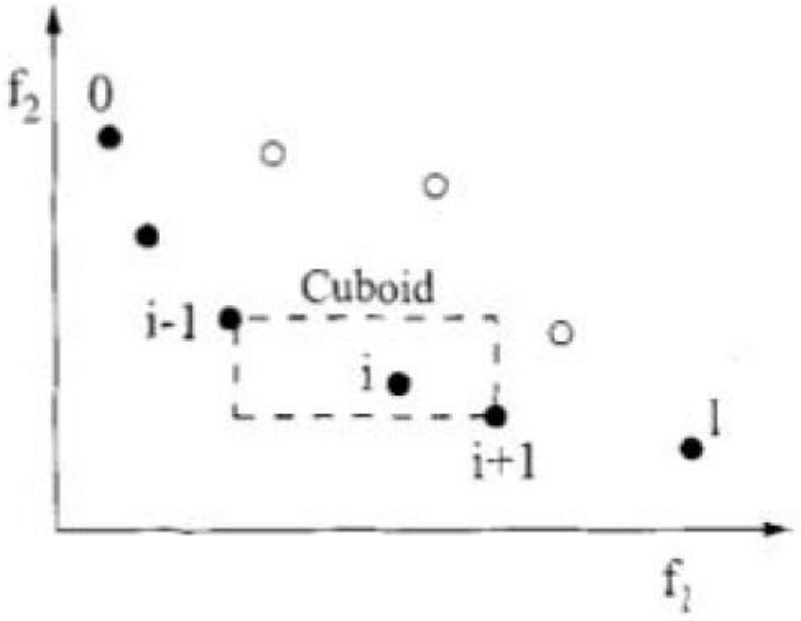

Figura 8 - Representação do cuboide do NSGA-II. Adaptação de Deb et al. (2002)

A iteração principal do algoritmo utiliza esses 2 operadores (separação em fronteiras e cálculo de densidade) para classificar as soluções que iram participar de uma geração futura.

Para iniciar a iteração principal do algoritmo, é criada uma população P_t onde t é o contador de iterações. Partindo da população P_t, cria-se Qt utilizando operadores genéticos de mutação e crossover que tem o mesmo número de indivíduos de Pt. Ambas populações são unidas para formar R, que é classificado de acordo com as fronteiras para formar a população inicial P_{t+1}. Como R tem o dobro de indivíduos de P, ele é dividido ao meio para formar a população P_{t+1}. Nessa divisão pode ocorrer a possibilidade de indivíduos da mesma fronteira serem deixados de fora da próxima geração, e essa escolha sobre qual indivíduo será descartado

é feita a partir da medida de densidade, isto é, os indivíduos da mesma fronteira que tiverem maior valor de densidade serão descartados. Para exemplificação de todo este processo tem-se a Figura 9.

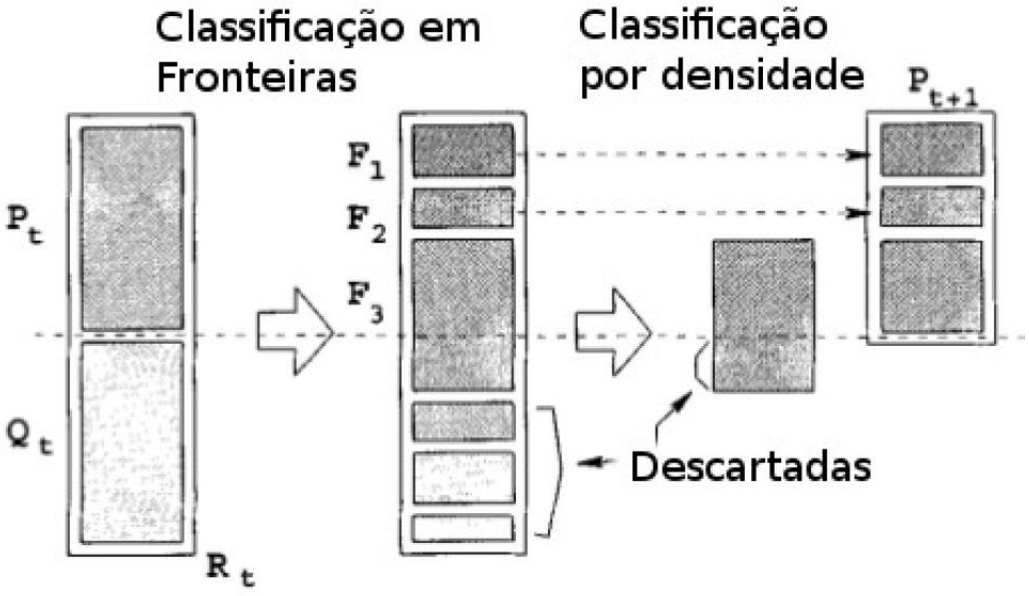

Figura 9 - Iteração principal do NSGA-II. Adaptação de Deb et al. (2002)

2.5 Sistemas *Fuzzy* Genéticos Multiobjetivo

Sistemas *Fuzzy* possuem aplicações em diversas áreas, como a de controle, classificação, regressão e mineração de dados. O modelo mais comum de SF são os sistemas *Fuzzy* baseados em regras (SFBRs). Como SFBRs podem ser trabalhosos de serem criados, surgiu necessidade de métodos que automatizem e otimizem essa tarefa. Com isso surgiram abordagens que utilizam os AG. Esse tipo de prática acabou sendo conhecido como Sistemas *Fuzzy* Genéticos.

No início dessa utilização, o objetivo principal era melhorar a acurácia, porém isso levava ao problema de que eram geradas regras não muito interpretáveis. O conceito de interpretabilidade está muito ligado a subjetividade, pois varia muito do entendimento de cada pessoa. Uma regra ou um conjunto de regras sendo observada por um especialista pode ser entendida mais facilmente, ao passo que para uma outra pessoa que não tem tanto conhecimento sobre o domínio pode ser difícil. Por isso surgiram trabalhos como Setnes, Babuska e

Verbruggen (1998), Oliveira (1999) e Guillaume (2001) que tentam sistematizar e entender o conceito de interpretabilidade utilizando algumas fórmulas.

Apesar de ainda não haver concordância sobre o que é menos ou mais interpretável, muitos autores dizem que a interpretabilidade está diretamente ligada ao número de antecedentes e consequente das regras, número de regras na base Fazzolari et al. (2013a).

A situação na qual a geração de SFBRs visa buscar bons índices de a acurácia e de interpretabilidade caracteriza um problema de otimização multiobjetivo. E como foi mostrado na seção 2.3, os AGMOs são adequados para tratar esse tipo de problema. Assim, SFBRs gerados a partir de AGMO recebem o nome de Sistemas *Fuzzy* Genéticos Multiobjetivos (SFGMO).

AGMOs não são utilizados somente para melhorar o balanceamento entre interpretabilidade e acurácia, (Fazzolari et al. (2013a) classificou a utilização deles conforme a Figura 10.

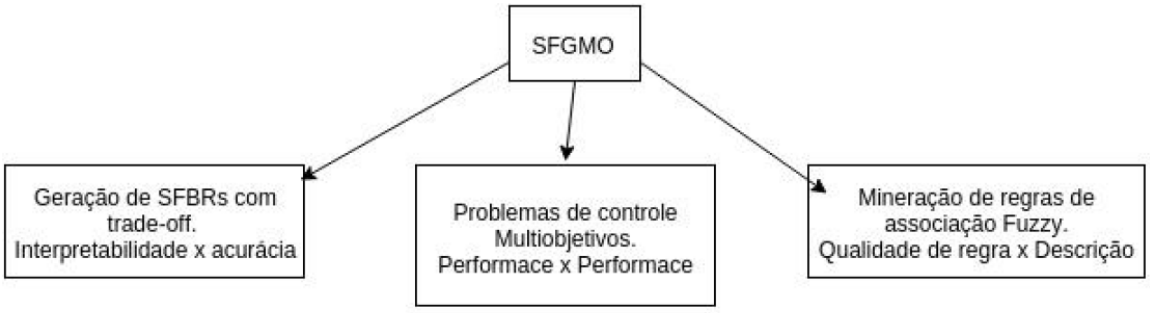

Figura 10 - Utilização de AGMOs nos SFBRs

Problemas de controle multiobjetivos envolvem a criação de controladores utilizando SFs, pois a tarefa a ser realizada pelos controladores não dispõe de modelos matemáticos precisos ou é uma tarefa realizada manualmente por humanos. AGMOs são utilizados pois para se realizar a tarefa de maneira eficaz. Geralmente há mais de um objetivo a ser satisfeito. SFGMOs desenvolvidos para essa finalidade tem como objetivo principal a acurácia, e fazem uso de duas estratégias, melhorar a função de pertinência e seleção de regras, aprender a base de regras.

Mineração de regras de associação *Fuzzy* diz respeito à mineração automatizada de conhecimento de bases de dados em busca de regras de associação. Regras de associação

descrevem a possibilidade de um conjunto de regras estarem diretamente ou indiretamente relacionado a outro, ou seja, o quanto o aparecimento de uma regra na base de regras pode influenciar ao aparecimento de outra. As metas principais são a predição, descrição ou uma combinação de ambas.

Quando a meta é predizer, a mineração de conhecimento da base de dado tem o objetivo de extrair conhecimentos que ajudem a prever valores explícitos, e nesse caso o foco principal é na medida de classificação. Na predição é utilizada a abordagem de aprendizado supervisionado, que faz uso de um conjunto de treinamento, e o objetivo principal é acertar qual o valor de saída de acordo com o valor de entrada. A predição é muito usada em classificações, regressões e pode também ser aplicadas e problemas de controle.

Quando o que se deseja é descrição, o objetivo é entender o processo de geração de dados procurando por padrões. Ao contrário da predição, que utiliza aprendizado supervisionado, a descrição utiliza um processo não supervisionado.

Geração de SFBRs com balanceamento entre interpretabilidade e acurácia foi explicado anteriormente no começo desta seção. Os meios de atingir esse balanceamento são:

1. Melhora dos componentes do SFBR – Feito através da modificação de função de pertinência e dos parâmetros de inferência.

2. Aprendizado da base de conhecimento – Feito aprendendo a regra de seleção, a base de regras ou simultaneamente os componentes da base de conhecimento.

A Figura 11 ilustra os itens anteriores.

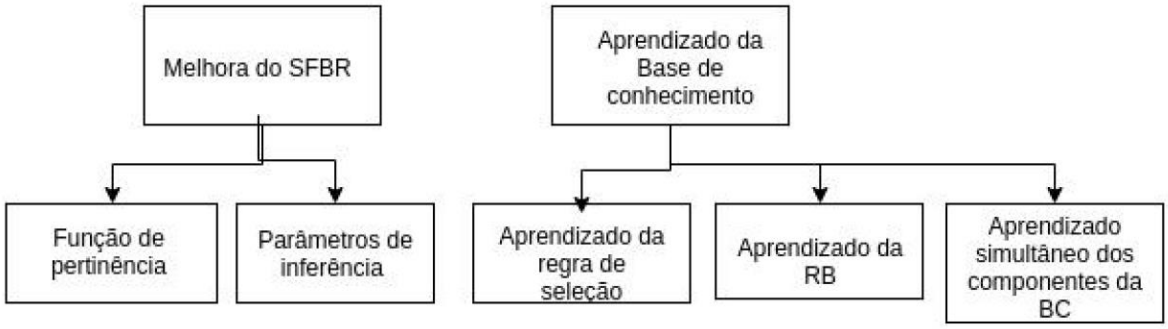

Figura 11 - Possíveis possibilidades de melhoramento na geração de SFBRs

Como o algoritmo proposto no trabalho utiliza somente o aprendizado da BC, não será abordado especificamente as partes de melhora na função de pertinência e nos parâmetros de inferências da Figura 11.

Aprendizado da regra de seleção – As primeiras contribuições feitas no aprendizado de regras de seleção foram realizadas nos trabalhos de Ishibuchi, Murata e Turksen (1997) e Ishibuchi e Nojima (2006). Nelas, foram utilizados AGMO clássicos que não possuíam elitismo. Esses trabalhos usam o AGMO para gerar um conjunto de soluções possíveis e depois selecionar as que satisfazem melhor os objetivos de interpretabilidade e acurácia.

Aprendizado da BR – Essa abordagem utiliza uma BD existente e a partir dela gera novas BR, obtendo uma nova BC. Apesar de os dois objetivos básicos serem a interpretabilidade e a acurácia, é proposto em Cococcioni et al. (2007) um terceiro objetivo que gera um índice de mudança como terceiro objetivo, para que os resultados gerados não se disperse muito em relação a BR original.

Aprendizado simultâneo – A intenção dessa última abordagem é aprender tanto a BD quanto a RB ao mesmo tempo. De acordo com Fazzolari et al. (2013b), como o espaço de busca é muito amplo, essa tarefa é difícil de ser desempenhada até mesmo para os Algoritmos Evolutivos.

O item a seguir descreverá uma abordagem de SFGMO, usando o algoritmo (2+2) M-PAES, por ser um algoritmo de interesse para o trabalho sendo proposto.

2.6 Geração de regras *Fuzzy* por meio do algoritmo (2 + 2)M-PAES

Esse algoritmo é uma adaptação criada por (Cococcioni et al. (2017) do clássico (2+2)-PAES que foi descrito em Sec. 2.3.1 que tem como objetivo a geração de Sistemas *Fuzzy* Baseados em regras com um bom balanceamento entre acurácia e interpretabilidade.

Retomando o significado da expressão entre parênteses, o 2+2 significa que a partir de 2 soluções, 2 novas serão geradas. Nessa adaptação foi incluído o operador genético de crossover, pois segundo os autores foi verificado experimentalmente que o crossover ajuda a criar uma aproximação da fronteira de Pareto. E cada solução sofre 2 mutações.

Antes de entender o funcionamento do algoritmo com as alterações propostas, é importante entender como a base de conhecimento é codificada em um cromossomo. Para isso, consideremos Xn para representar o conjunto de variáveis linguísticas, e Na,i para representar o conjunto de termos linguísticos relativos as variáveis linguísticas. n é a quantidade e índice da variável linguística e i é o termo linguístico relativo a variável linguística. Por exemplo, o termo $X1$ possui o conjunto de variáveis linguísticas A1,i. O cromossomo responsável por representar a base de conhecimento é criado possuindo

Jxn genes, sendo que J é a quantidade de regras que o sistema possui. Cada gene n representa um termo linguístico e recebe o valor i que representa a variável linguística, caso ele receba o valor 0, significa que o termo não é relevante para aquela regra. Como exemplo, a Figura 13 é a base de regra que gera a Figura 12.

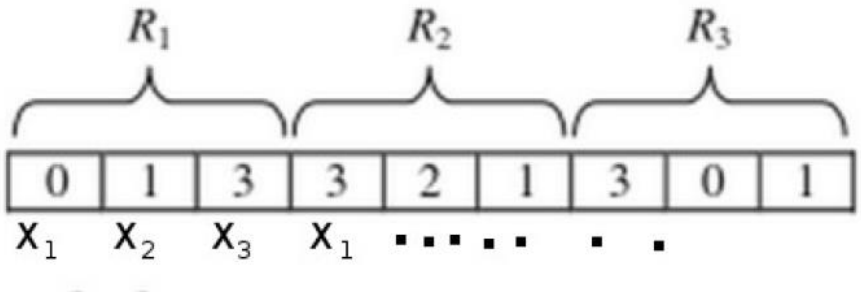

Figura 12 - Representação de um cromossomo no (2+2) M-PAES. Adaptação de Cococcioni et al. (2007)

A BR pode ser também facilmente expressa matricialmente. Cada linha representa uma regra e cada coluna uma variável linguística. Os valores representam os termos linguísticos. Sendo assim, a matriz que representa a base de regras da Figura 13 é:

$$R_1 : \textbf{IF } X_2 \textbf{ is } A_{2,1} \textbf{ THEN } X_3 \textbf{ is } A_{3,3}$$
$$R_2 : \textbf{IF } X_1 \textbf{ is } A_{1,3} \textbf{ and } X_2 \textbf{ is } A_{2,2} \textbf{ THEN } X_3 \textbf{ is } A_{3,1}$$
$$R_3 : \textbf{IF } X_1 \textbf{ is } A_{1,3} \textbf{ THEN } X_3 \textbf{ is } A_{3,1}$$

Figura 13 - Exemplo de base de regras do (2+2) M-PAES. Adaptação de Cococcioni et al. (2007)

$$\text{Matriz} = \begin{bmatrix} 0 & 1 & 3 \\ 3 & 2 & 1 \\ 3 & 0 & 1 \end{bmatrix}$$

Depois da explicação em como a base de regras é codificada em um cromossomo, podemos entender melhor como é aplicado os operadores de crossover e as mutações.

As 3 operações acontecem de acordo com uma probabilidade que varia dependendo da ocorrência de outros operadores. Primeiro é aplicado o operador de crossover com a probabilidade de 0.5. Após isso, caso tenha ocorrido o crossover, a probabilidade de também ocorrer as mutações é de 0.01. Se o crossover não ocorreu, então há uma chance de 0.55 de ocorrer a primeira mutação. A última mutação ocorre sempre que a primeira não ocorre.

O ponto de corte é um gene em comum entre as duas soluções. Esse gene é escolhido por um número gerado aleatoriamente entre Mnin e ρmin que posteriormente é multiplicado por (F + 1). Mmin é o número mínimo de regras que tem que estarem presentes na base de regras, ρmin é o número mínimo de regras entre as duas soluções e F são as variáveis de entrada.

A primeira mutação cria aleatoriamente γ regras que varia entre 1 e o número máximo que não ultrapasse o número de regras máximo que um cromossomo pode conter. Como as regras precisam dos antecedentes e do consequente, tudo é gerado aleatoriamente respeitando os limites de espaço das variáveis e termos linguístico, e a quantidade de antecedentes possíveis. É sorteado um número que serve para a quantidade de antecedentes e após isso são sorteados números de acordo com a quantidade de variáveis linguística que cada termo possui para preencher as regras.

A segunda mutação altera um número de genes aleatórios por outro número aleatório que corresponde à variável linguística.

O restante do algoritmo funciona como o PAES descrito na seção 2.4.1, só alterando que em toda nova geração são sorteadas 2 novas soluções do arquivo para serem as atuais.

Capítulo 3

PROPOSTA DO TRABALHO

3.1 Proposta do Trabalho

Todos os algoritmos descritos no Capítulo 2 possuem um método de seleção de soluções que receberão os operadores genéticos para a geração de uma nova população. Essa seleção é feita tentando escolher as soluções com as melhores características. A motivação dessa seleção está ligada ao elitismo, ou seja, soluções com as melhores características têm maior probabilidade de gerar melhores populações depois de receberem os operadores genéticos.

A seleção de soluções em AGMO é feita utilizando a métrica de relação de dominância entre as soluções. Essa métrica pode estar relacionada, por exemplo, a organização de fronteiras que se tornam prioritárias na hora da seleção ou na separação em arquivo de soluções não dominadas.

Quando os indivíduos selecionados por esses meios ultrapassam o tamanho máximo permitido para a próxima geração, os AGMO selecionam as soluções que serão descartadas utilizando mecanismos baseados em proximidade de soluções. Essa decisão é feita priorizando as soluções que estão mais afastadas uma das outras para melhorar a dispersão da fronteira.

O NSGA-II utiliza duas soluções vizinhas para medir a densidade da área e assim escolher as soluções que estão em áreas mais densas para serem descartadas. Já o SPEA2 procura na fronteira as duas soluções mais próximas, depois de identifica-las ele verifica qual das duas está mais perto de uma terceira solução e a que estiver mais próxima dessa terceira solução é descartada.

No caso do (2+2) M-PAES a seleção de soluções para a próxima geração é aleatória (sorteada), pois como o algoritmo só guarda as soluções não dominadas, não são necessárias métricas para se escolher quais soluções serão selecionadas. A ideia de escolher soluções não dominadas para ser descartada só ocorre quando é necessário adicionar uma nova solução e o limite máximo de soluções guardadas já foi alcançado. Caso isso ocorra o algoritmo também usa um mecanismo baseado em proximidade de soluções para escolher qual solução será descartada para a entrada da nova solução.

A Figura 14, retirada dos experimentos ilustra o conceito de amplitude e dispersão de soluções.

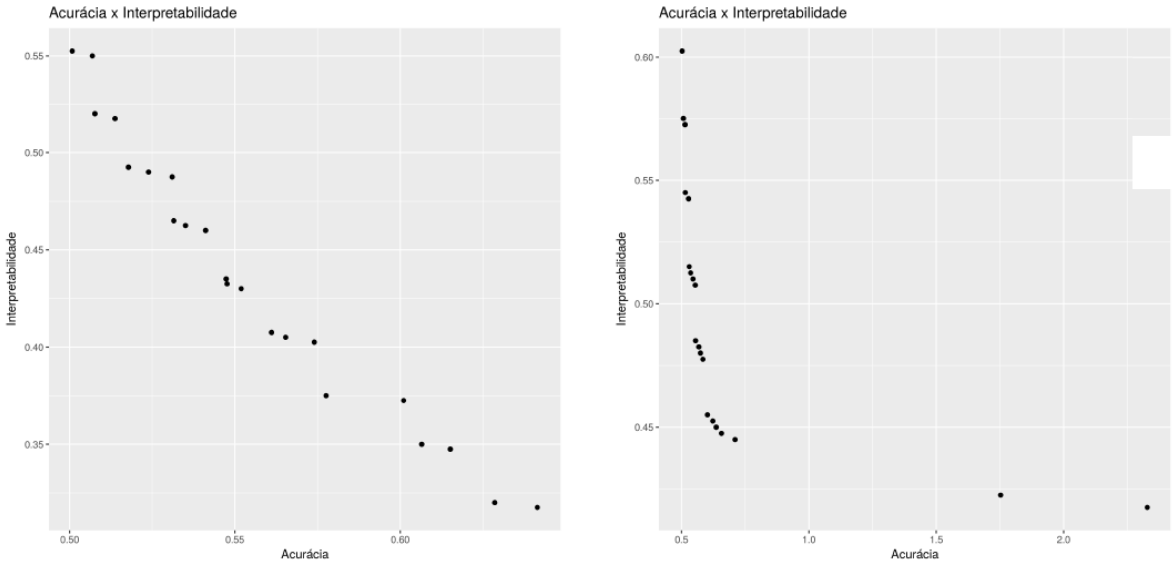

Figura 14 - Exemplo de fronteiras com diferentes dispersões

Na imagem da esquerda é possível observar que há uma uniformidade na fronteira, as soluções estão melhor espaçadas entre si e a fronteira cobre mais uniformemente o espaço. Na figura da direita podemos observar que são formados pequenos grupos de soluções e há um espaço entre os grupos. Caso seja necessário escolher uma solução de cada fronteira, a fronteira da esquerda oferece mais granularidade para a escolha pois não existem espaçamentos grandes entre as soluções.

Como o NSGA-II e o SPEA2 trabalham com o conceito de população, eles sempre acionam com mais frequência esse mecanismo que ajuda na dispersão das soluções.

No (2+2) M-PAES isso só é acionado quando não há espaço para uma solução nova não dominada que precisa ser guardada.

A seleção aleatória do (2+2)PAES pode gerar situações nas quais as soluções escolhidas estejam ao lado uma da outra, isso não é necessariamente ruim pois pode ser encontrada uma solução na região que domine as selecionadas. Entretanto isso também poderia acontecer selecionando soluções próximas, mas não vizinhas e ainda teria a vantagem de melhorar a dispersão das soluções. Para exemplificar na prática como a escolha aleatória pode desfavorecer a dispersão da fronteira observe a Figura 15 retirada dos experimentos.

Na Figura 15 da esquerda suponhamos que as soluções em vermelho tenham sido escolhidas, então temos um exemplo de soluções que são vizinhas e na figura da direita uma outra opção foi selecionada em vermelho que poderia não alterar tanto o resultado de gerar soluções não dominadas e ao mesmo tempo poderia contribuir para a dispersão da fronteira.

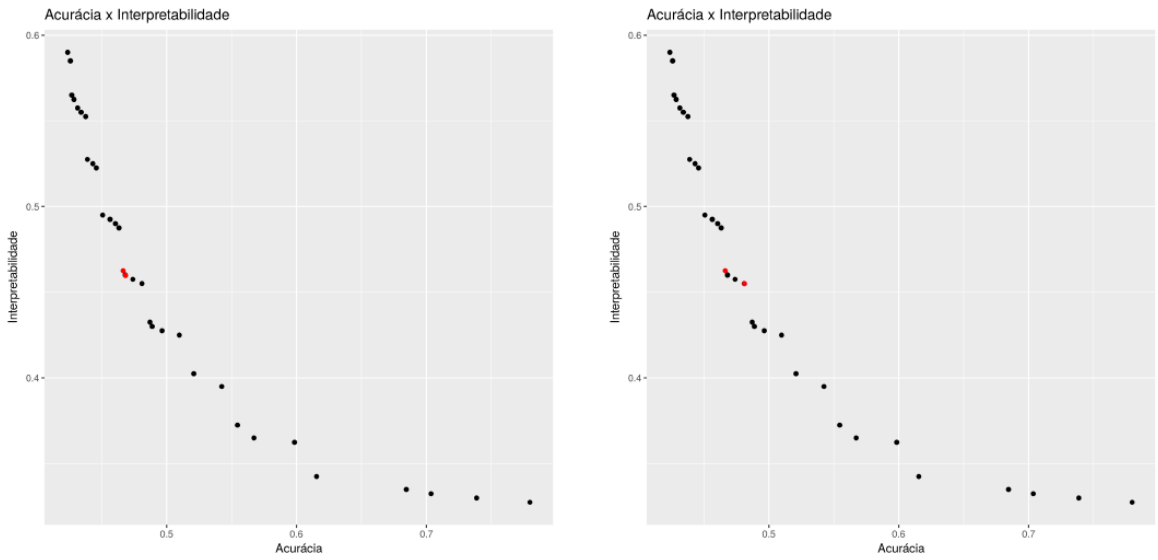

Figura 15 - Exemplo de escolha de soluções

Apesar da ideia de se evitar escolher soluções vizinhas, é importante também não selecionar poucas soluções para participar do sorteio pois a aleatoriedade na seleção das soluções é uma característica importante dos AG, portanto, a proposta apresentada neste trabalho considera a distância entre as soluções sem eliminar totalmente a escolha aleatória. Assim, o método de seleção proposto aqui consiste de duas etapas: seleção das soluções candidatas, com base na distância entre elas e seleção aleatória de duas soluções (sorteio) entre as candidatas selecionadas no passo anterior.

Para escolher as soluções que participarão do sorteio, as soluções foram organizadas em um vetor ordenado de forma crescente em relação a um dos objetivos. Assim, a primeira solução

do vetor é a melhor no objetivo escolhido e a última solução é a melhor no outro objetivo. A partir disso escolhemos inicialmente a solução do final do vetor para participar do sorteio. O resto das soluções que vão participar do sorteio são escolhidas percorrendo o vetor e escolhendo uma sim e outra não alternadamente. O algoritmo 4 ilustra em maiores detalhes a proposta.

Para um exemplo mais prático considere uma fronteira que tenha 20 soluções. A última solução é escolhida, em seguida a solução 0 é escolhida e vai alternando entre escolhida e não escolhida conforme a Figura 16.

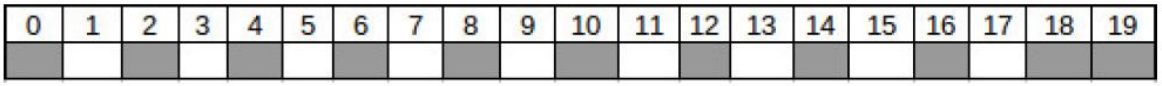

Figura 16 - Vetor de Exemplos

Algorithm 4: Seleção de soluções proposta

```
Input: Valor do arquivo escolhido para que comece a dividir pela metade as
       soluções selecionadas NumeroDeSoluções
Output: Lista com 2 soluções selecionadas na variável Selecionados
if TamanhoDaLista(Arquivo)>NumeroDeSoluções then
    ListaOrdenada ← Ordenar(Arquivo);
    ListaDeSorteio ← CriaLista();
    ListaDeSorteio.Adicionar(UltimoElemento(ListaOrdenada));
    Contador ← 0;
    while Contador<TamanhoDaLista(ListaOrdenada) do
        if Contador% 2 == 0 then
            ListaDeSorteio.Adicionar(ListaOrdenada[Contador]);
        Contador ← Contador + 1;
    Selecionados ← SelecionarAleatoriamente(ListaDeSorteio);
else
    Selecionados ← SelecionarAleatoriamente(Arquivo);
```

A Figura 17 como seriam selecionadas as soluções de um arquivo com 32 soluções.

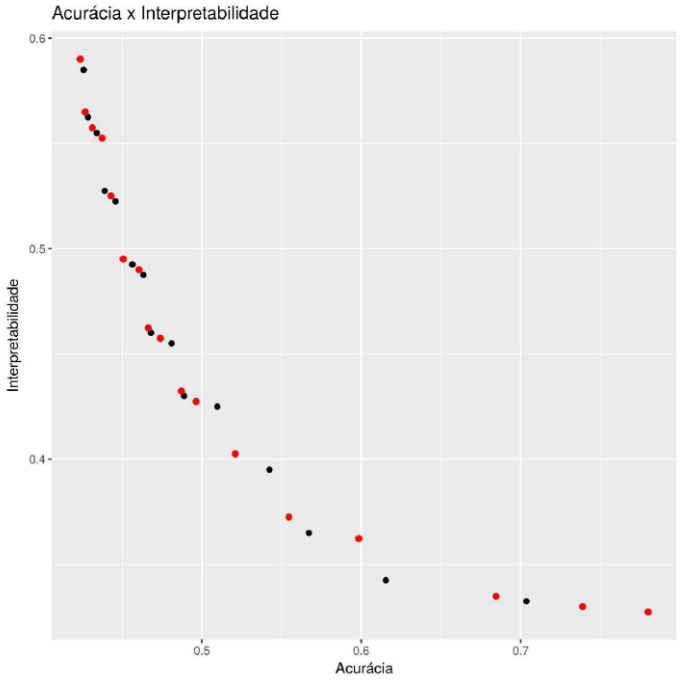

Figura 17 - Exemplo com as soluções selecionadas marcadas em vermelho

Esse método proposto evita que soluções vizinhas sejam escolhidas e ao mesmo tempo tenta manter a fronteira esparsa.

3.2 Recursos

A linguagem escolhida para a implementação, experimentação e testes foi o JAVA. Foi utilizada a framework jMetal já possui diversos AGMO (MOEAD, PAES, PESA2, NSGA-II, SPEA2 e outros) e também auxilia na experimentação, e contém alguns conjuntos de soluções para teste (DURILLO; NEBRO, 2011).

Para a construção de Tabelas e gráficos foi utilizada a ferramenta estatística R (KOHL, 2015; RAHLF, 2017).

Foi necessário também fazer uma grande adaptação no framework jMetal pois não havia suporte a um SF. A Figura 18 foi retirada do manual do jMetal e modificada para mostrar as alterações (em vermelho) que tiveram que ser feitas.

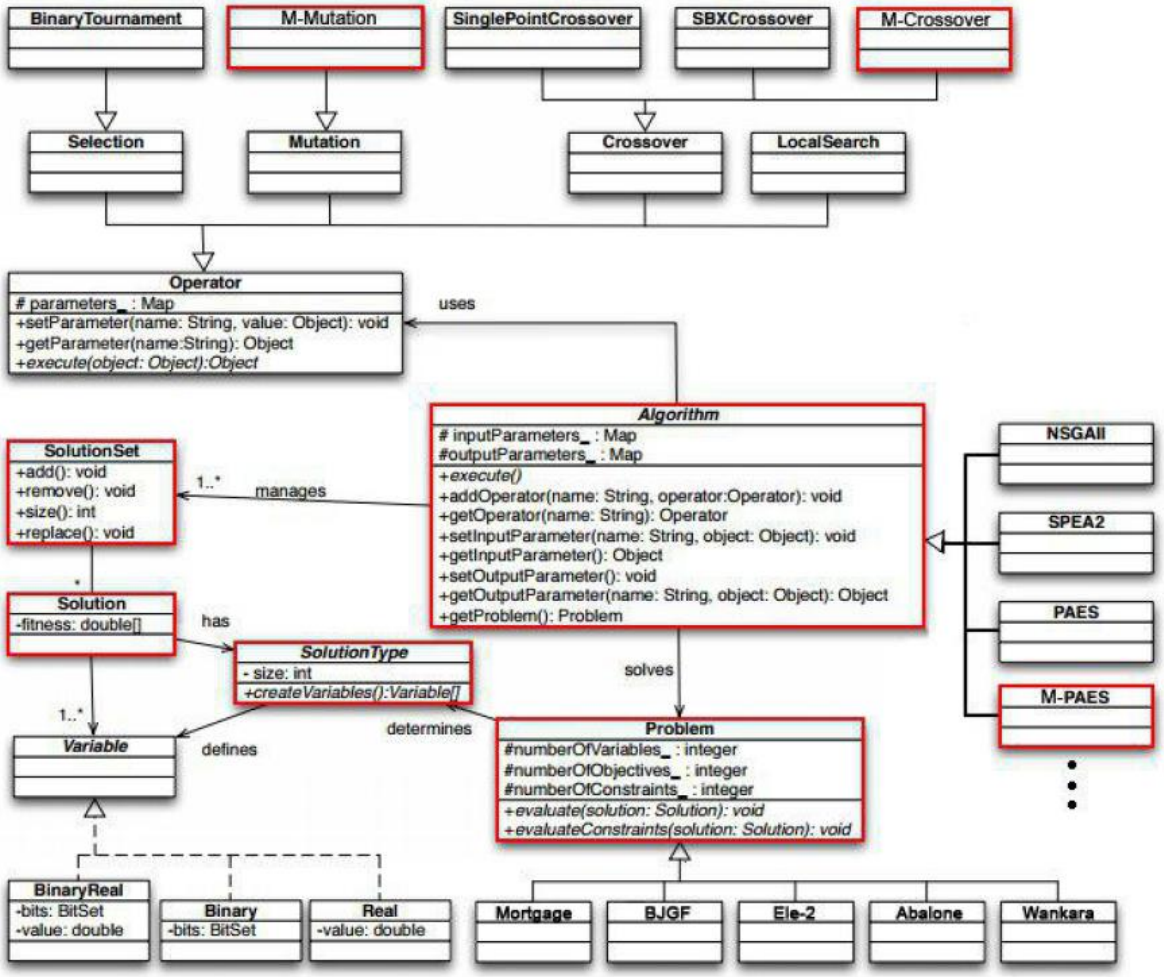

Figura 18 - Alterações feitas ao framework

A classe *M-Mutation* da Figura 18 na verdade é uma abstração das 2 mutações especificas que tiveram que ser criadas. Tanto a classe *M-Mutation* e a *M-Crossover* foram criadas especificamente para este trabalho. A framework facilitou essa parte pois a classe *Mutation* e Crossover são Interfaces, bastando apenas implementa-las. Os cuidados específicos que tiveram que ser tomados ao criar os operadores foram:

- O cruzamento não pode ocorrer no meio de uma Regra *Fuzzy*;
- Ao final de cruzamentos e mutações o cromossomo tem que ser inspecionado para remoção de regras que ocasionalmente tenham ficado iguais;

- Os operadores são responsáveis por atualizar as estruturas que foram implementadas na classe *Solution* para a contagem de antecedentes e regras que são usados para o cálculo da interpretabilidade.

A classe M-PAES é o algoritmo (2+2)M-PAES que foi totalmente implementado pois a framework não o disponibilizava.

A classe *Solution* teve que ser estendida para a adição de estruturas responsáveis por manter: o número máximo de regras *Fuzzy* possível, o número atual de regras *Fuzzy*, o número máximo de variáveis possível, o número de variáveis atuais, o mapeamento da quantidade de antecedentes que cada regra *Fuzzy* tinha atualmente. O problema de se estender essa classe é que todas as outras classes do *framework* que a recebiam tiveram também que ser modificadas para conseguirem receber e enviar esses parâmetros estruturais. Foram também implementadas classes paralelas à framework que tiveram a função de ler arquivos do tipo ARFF (utilizado para descrever instâncias e atributos de base de dados), extrair as informações e passa-las de forma já organizada para a classe *Fuzzy* e, uma classe *Fuzzy* que teve a função de receber os valores dos atributos, criar as partições *Fuzzy* e fazer as inferências.

3.3 Métricas

Para medir e validar as modificações feitas, foram aplicadas na acurácia e interpretabilidade as seguintes métricas: Valor mínimo, 1o quadrante, Mediana, média, 3o quadrante, valor máximo. E nas fronteiras de Pareto foram aplicadas as métricas: Distância total média (D.T.M), Distância média entre soluções (D.M.S) e Número de soluções (N.Sol.).

A Mediana é o valor que separa a amostra na metade, se a amostra tiver um número par de elementos a Mediana é a média dos elementos centrais. Se tivermos um conjunto com os elementos [0,1,1,3,4,5,5,8,15] a Mediana seria o número 4. Essa medida é importante pois não é afetada por valores máximos e mínimos dos extremos da amostra e ajuda a ter uma noção do quanto a média representa a amostra.

A distância total média está ligada a bateria de execuções que foram feitas nos algoritmos. A bateria de execuções executa um algoritmo com os mesmos parâmetros 100 vezes. Uma execução do algoritmo pode gerar até 100 soluções que formam a fronteira de

Pareto. Se somarmos a distância euclidiana entre cada ponto da fronteira de Pareto obtemos a distância total da fronteira de Pareto de uma execução. A distância total média é a soma total das distâncias das fronteiras de Pareto de N execuções divididas por N. o traço vermelho na Figura 19 mostra quais são as distâncias somadas para se obter a distância total.

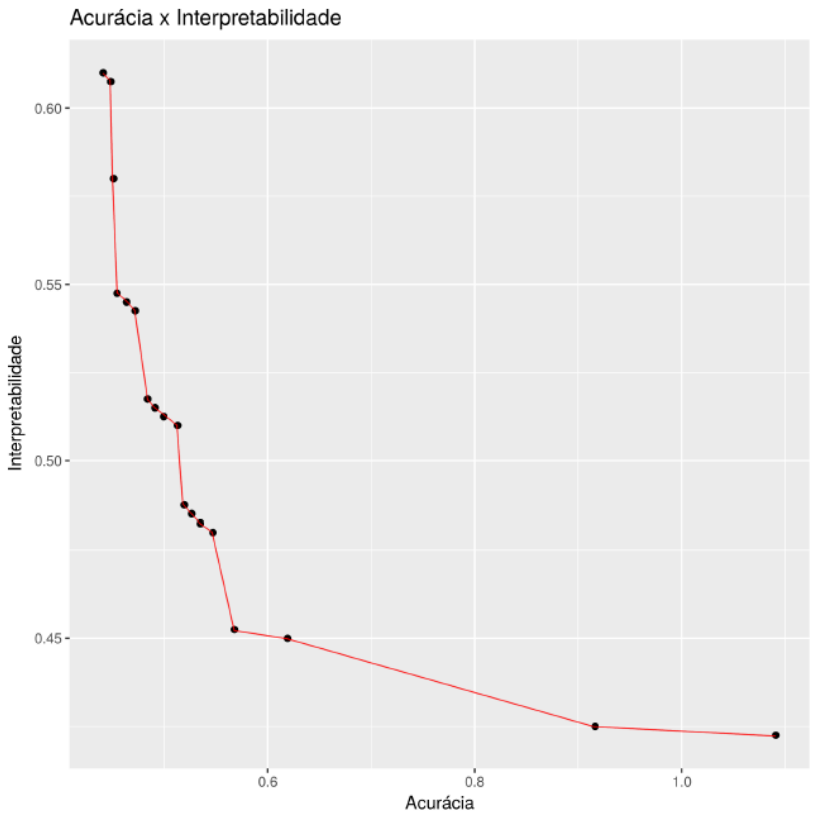

Figura 19 - Distância total euclidiana

A D.T.M é importante pois mostra o quanto os valores dos objetivos variaram no domínio. Nessa métrica, quanto maior o valor melhor o resultado.

A distância média entre soluções tem o funcionamento similar a distância total média. Cada execução gera até 100 soluções, o número de soluções geradas é dividido pela distância da fronteira. Essa medida é importante pois representa o quão ocupada está a fronteira. Se os valores forem baixos, significa que as soluções estão mais juntas. Assim como a métrica anterior, quanto maior essa medida melhor.

O número de soluções é a média do número de soluções obtidas nas 100 execuções. Assim como nas duas anteriores, nessa métrica quanto maior o valor, melhor o resultado.

Capítulo 4

ANÁLISE E EXPERIMENTOS

4.1 Bases de Dados

Uma base de dados foi retirada do artigo Cococcioni et al. (2017) e as outras foram retiradas do site: www.keel.se (KEEL - *Knownlegde Extraction based on Evolutionary Learning*) que possui um repositório público com diversas bases de dados para diversos tipos de problemas da área de aprendizado evolutivo. Todas foram utilizadas separando em metade das instâncias para treinamento e metade para teste. A base do artigo Cococcioni et al. (2017) é a *Box-Jenkins Gas Furnace* (BJGF), ela possui valores de ar que são injetados em um sistema de aquecimento a gás e têm as saídas de CO2 como resultado. Cada instancia da base é uma combinação de 6 valores entradas de ar e 5 valores de saída de CO2, sendo o último valor o que se busca acertar. A Tabela 1 mostra a relação das variáveis da base BJGF e os valores que elas podem assumir. A última linha dessa Tabela e das próximas que descrevem as variáveis das bases são as variáveis de saída.

Tabela 1 - Entradas e valores possíveis da base BJGF

Nome	Valor Mínimo	Valor Máximo
InputGasRate1	-2716	2834
InputGasRate2	-2716	2834
InputGasRate3	-2716	2834
InputGasRate4	-2716	2834
InputGasRate5	-2716	2834
InputGasRate6	-2716	2834
CO21	45.6	60.5
CO22	45.6	60.5
CO23	45.6	60.5
CO24	45.6	60.5

Foi utilizada também uma base chamada *mortgage* que contém valores semanais de 20 anos de hipotecas dos Estados Unidos. São 16 variáveis e a intenção é acertar a taxa de 30 anos de hipoteca. A Tabela 2 mostra as entradas e valores que elas podem assumir.

A base concrete tenta predizer a força de compreensão do concreto de acordo com as 7 diferentes medidas dos ingredientes. A Tabela 3 mostra os valores de entradas e como o valor deles pode variar.

Tabela 2 - Entradas e valores possíveis da base *mortgage*

Nome	Valor Mínimo	Valor Máximo
1MonyhCDRate	3.02	20.76
1Y-CMaturityRate	77.055	142.645
3M-Rate-AuctionAverage	6.49	18.63
3M-Rate-SecondaryMarket	2.67	16.75
3Y-CMaturityRate	2.69	16.76
5Y-CMaturityRate	4.09	16.47
BankCredit	4.17	76.13
Currency	1130.9	4809.2
DemandDeposits	105.6	533.0
FederalFunds	225.8	412.1
MoneyStock	2.86	20.06
CheckableDeposits	381.1	1154.1
LoansLeases	269.9	803.4
SavingsDeposits	868.1	3550.3
TradeCurrencies	175.6	1758.1
30Y-CMortgageRate	3.02	17.15

Tabela 3 - Entradas e valores possíveis da base *concrete*

Nome	Valor Mínimo	Valor Máximo
Cement	102.0	540.0
BlastFurnaceSlag	0.0	359.4
FlyAsh	0.0	200.10006
Water	121.8	247.0
Superplasticizer	0.0	32.200001
FineAggregate	594.0	992.6
Age integer	1	365
ConcreteCompressiveStrength	2.33	82.6

A base *wankara* contém os dados meteorológicos de uma cidade durante aproximadamente 4 anos e tenta predizer qual será a temperatura média. A Tabela 4 contém as variáveis e os valores que elas podem assumir.

A Tabela 5 resume o número de Variáveis de cada base de dados e o número de instâncias.

Tabela 4 - Entradas e valores possíveis da base *wankara*

Nome	Valor Mínimo	Valor Máximo
Max_temperature	23.0	100.0
Min_temperature	-7.1	65.5
Dewpoint	-3.1	57.6
Precipitation	0.0	4.0
Sea_level_pressure	29.16	30.6
Standard_pressure	26.3	27.18
Visibility	0.2	11.5
Wind_speed	0.0	18.0
Max_wind_speed	2.19	57.4
Mean_temperature	7.9	81.8

Tabela 5 - Bases de dados utilizadas

Bases de dados	Instâncias	Variáveis
BJFG	290	11
ele-2	1056	5
concrete	1030	8
wankara	1609	10
mortgage	1048	16

4.2 Experimentos

Essa seção traz as baterias de testes feitas com as bases da seção 4.1. Os parâmetros dos experimentos se basearam no artigo Cococcioni et al. (2017).

Como há aleatoriedade nas mutações, cruzamentos e divisão da base de dados em treinamento e teste, cada bateria de teste consiste em executar o algoritmo cem vezes e obter a média dos resultados. O resultado de cada execução é um conjunto de N soluções (o número de soluções varia e pode ser no máximo 100), nesse conjunto cada solução tem uma acurácia, interpretabilidade e distância em relação as outras soluções. A acurácia e interpretabilidade de cada solução de uma execução é somada e dividida pelo número de soluções encontradas, assim é obtida uma média de acurácia e interpretabilidade de uma execução. As distâncias de cada solução são somadas e dividas pelo número de soluções para se obter a distância total da fronteira e a média de distância entre as soluções. Após a aquisição destes dados de uma

execução, faz-se os mesmos passos para as outras noventa e nove execuções e então é obtida a média. A Figura 20 apresenta uma representação esquemática de uma bateira de testes.

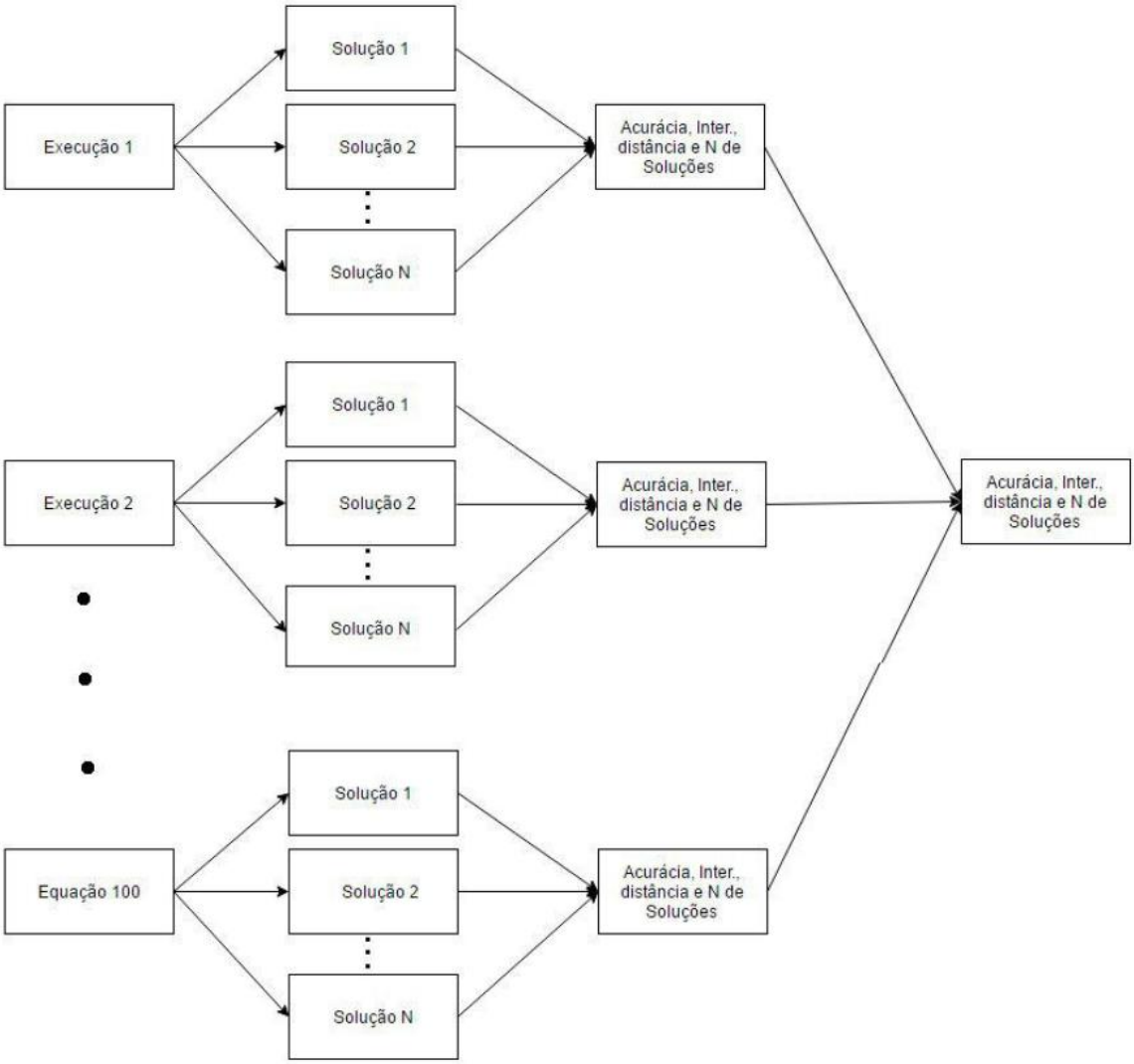

Figura 20 - Esquema da bateria de testes

Os valores de probabilidade de cruzamento e mutação são os mesmos usados no artigo Cococcioni et al. (2007). O cruzamento tem 0.5 de chances de ocorrer, se ocorrer o cruzamento a probabilidade da mutação ser ativada é 0.01. Se não ocorrer o cruzamento, a mutação é ativada e por isso vai ocorrer. Quando a mutação é ativada a chance de ocorrer a primeira mutação é 0.5 e a segunda mutação só ocorre se a primeira não ocorrer.

Ainda seguindo os parâmetros utilizados em Cococcioni et al. (2007), foram utilizados 7 conjuntos *Fuzzy* e 20 regras por solução. O artigo fez testes variando o número de regras e

restringindo o número de variáveis possíveis por regra. Como no algoritmo implementado não há restrição quanto ao número de variáveis por regras, foi utilizado 20 regras que é o dado utilizado pelo artigo sem restrições de variáveis.

Esta seção será subdividida de acordo com as bases de regras utilizadas para os testes. Os resultados mostrados inicialmente foram retirados da base de dados do artigo Cococcioni et al. (2007).

4.2.1 BJGF

Por ser a base utilizada no artigo, essa foi a base que mais recebeu baterias de testes. No total foram 5 baterias utilizando o algoritmo (2+2)M-PAES e 5 utilizando o mesmo algoritmo com a implementação de seleção da proposta desse trabalho. A Tabela 6 e a Tabela 7 são das baterias no (2+2)M-PAES sem a escolha aleatória de soluções da fronteira.

Tabela 6 - Testes utilizando o (2+2)M-PAES SEM seleção

Métricas	Bateria 1			Bateria 2			Bateria 3		
	Treino	Teste	Inter.	Treino	Teste	Inter.	Treino	Teste	Inter.
Min.	0.4761	0.7116	0.4019	0.5482	0.7529	0.3826	0.4394	0.7787	0.3802
1º Qu.	0.5269	0.9380	0.4678	0.5143	1.1260	0.4555	0.4996	1.0287	0.4507
Mediana	0.5650	1.3748	0.4874	0.5449	1.9805	0.4769	0.5296	1.4546	0.4739
Média	0.5744	1.6845	0.4879	0.5661	2.2316	0.4804	0.5592	1.6791	0.4740
3º Qu.	0.5975	2.2212	0.5095	0.5850	3.0506	0.5146	0.5723	2.1802	0.5022
Max.	1.1482	4.6188	0.5556	0.9996	6.1819	0.5514	1.2196	3.9722	0.5635
D.T.M	2.05498624			2.98607736			3.63529809		
D.M.S	0.02669686			0.03497314			0.04456694		
N. Sol.	90.02			89.70			92.54		

Tabela 7 - Continuação da Tabela 6

Métricas	Bateria 4			Bateria 5		
	Treino	Teste	Inter.	Treino	Teste	Inter.
Min.	0.4458	0.6811	0.3729	0.4455	0.7157	0.3873
1º Qu.	0.5137	1.2326	0.4563	0.5173	0.8956	0.4480
Mediana	0.5599	1.9571	0.4883	0.5452	1.4941	0.4691
Média	0.5670	2.1917	0.4861	0.5604	1.7908	0.4739
3º Qu.	0.5964	2.8626	0.5115	0.5934	2.3601	0.5014
Max.	1.0091	6.9465	0.5646	0.8178	5.6783	0.5851
D.T.M	2.52212068			2.20481937		
D.M.S	0.02716027			0.02402982		
N. Sol.	93.72			93.22		

Os campos da Tabela 6 e da Tabela 7 em destaque cinza são os melhores valores encontrados entre as 5 baterias de testes. A métrica mais importante de avaliação de interpretabilidade e de acurácia é a média. Se for observada a média das acurácias dos testes, nota-se a existência de dois valores que são aproximadamente 1.6, dois valores acima de 2.1 e somente um valor entre 1.6 e 2.1. Isto mostra como os resultados podem variar devido aos fatores aleatórios, mesmo que cada bateria executou o algoritmo cem vezes.

Se analisarmos a interpretabilidade dos resultados, pode-se perceber que não houveram variações expressivas, tanto que se subtrair do menor valor de interpretabilidade com o maior valor obtêm-se como resultado 0.014. Então, pode-se dizer que a aleatoriedade não influencia tanto na interpretabilidade quanto na acurácia nesta base.

Todas apresentaram uma boa quantidade de soluções totais, pois o valor máximo de soluções no arquivo é cem.

Para comparação dos resultados, foi rodado também 5 baterias com a alteração proposta neste trabalho e os resultados estão na Tabela 8 e na Tabela 9.

Tabela 8 - Testes utilizando o (2+2) M-PAES COM seleção na base BJGF

Métricas	Bateria 1			Bateria 2			Bateria 3		
	Treino	Teste	Inter.	Treino	Teste	Inter.	Treino	Teste	Inter.
Min.	0.4503	0.7556	0.3824	0.4868	0.7118	0.4138	0.4589	0.8015	0.3574
1º Qu.	0.4959	0.9109	0.4452	0.5365	0.9052	0.4610	0.5339	1.3873	0.4653
Mediana	0.5212	1.3120	0.4674	0.5519	1.3400	0.4833	0.5642	1.9224	0.4903
Média	0.5317	1.5652	0.4681	0.5754	1.4955	0.4823	0.5782	2.1550	0.4862
3º Qu.	0.5525	1.8487	0.4879	0.5966	1.9373	0.5063	0.6041	2.5445	0.5128
Max.	0.7571	6.2949	0.5671	1.1322	3.9646	0.5746	0.9600	5.6981	0.5533
D.T.M	2.36351320			2.41055877			2.25894266		
D.M.S	0.02489346			0.02680183			0.02589917		
N. Sol.	95.09			91.59			90.38		

Tabela 9 - Continuação da Tabela 8

Métricas	Bateria 4			Bateria 5		
	Treino	Teste	Inter.	Treino	Teste	Inter.
Min.	0.4432	0.7614	0.3336	0.4807	0.7095	0.4100
1º Qu.	0.5409	0.9530	0.4606	0.5636	0.8828	0.4566
Mediana	0.5741	1.2861	0.4837	0.5915	1.2256	0.4816
Média	0.5901	1.4784	0.4821	0.6083	1.6625	0.4817
3º Qu.	0.6208	1.5941	0.5045	0.6445	2.1140	0.5053
Max.	1.0391	3.8290	0.5734	0.8682	6.4244	0.5457
D.T.M	2.81783386			2.9341409		

D.M.S	0.03064586	0.0332013
N. Sol.	93.72	92.41

Assim como nas baterias anteriores a interpretabilidade também não teve uma alteração significativa na média. Para sintetizar todos os resultados em uma única tabela e facilitar a visualização, foi tirada uma média das 5 baterias sem seleção e das 5 baterias com seleção e o resultado foi apresentado na Tabela 10.

Tabela 10 - Média das 5 baterias sem e com seleção na base BJGF

Métricas	Sem seleção			Com seleção		
	Treino	Teste	Inter.	Treino	Teste	Inter.
Min.	0,471	0,728	0,38498	0,46398	0,74796	0,37944
1º Qu.	0,5137	1,04418	0,45566	0,53416	1,00784	0,45774
Mediana	0,54892	1,65222	0,47912	0,56058	1,41722	0,48126
Média	0,56542	1,91554	0,48046	0,57674	1,67132	0,48014
3º Qu.	0,58892	2,53494	0,50784	0,6037	2,00772	0,50336
Max.	1,03886	5,47954	0,56404	1,066	5,2422	0,56282
D.T.M	2,680660348			2,556997878		
D.M.S	0,031485406			0,028288324		
N. Sol.	91,84			92,638		

Pode-se observar da Tabela 10 que quando não há seleção de soluções, o treinamento acerta em média mais, porém tem como consequência o erro médio maior nos testes. Observa-se que a média do treinamento sem seleção e com seleção, notamos que a diferença é de aproximadamente 0,02 enquanto no teste a diferença sobe para 0,24 em favor da seleção de soluções. A diferenças de médias entre as interpretabilidades foi de apenas 0,00032, ou seja, praticamente iguais.

Ao contrário do que se previa, a seleção de regras não melhorou a amplitude e nem a dispersão das soluções, mas a diferença não foi tão grande quanto ao ganho no acerto médio do teste. A diferença média entre o número de soluções encontradas foi menor que um, sendo então um valor muito baixo.

4.2.1.1 Ele-2

Os resultados das baterias rodadas nesta base obtiveram valores maiores de acurácia devido a característica da base possuir uma saída que varia de 64.470001 à 8546.030273.

A Tabela 11 representa os resultados obtidos em 2 baterias de testes com o algoritmo (2+2)M-PAES sem a seleção de soluções.

Pode ser observado que a bateria que obteve o melhor treinamento não obteve o melhor valor nos testes, que foi a mesma coisa que aconteceu na Tabela 10.

A interpretabilidade teve a mesma característica dos testes do BJGF de não variar muito. E a média da acurácia dos testes entre as baterias variou em 23,1. É interessante observar essa variação para se ter uma noção no quanto pode variar essa medida entre as baterias. O ideal seria ter sido feito mais baterias de testes. A Tabela 12 possui os valores obtidos em 2 baterias rodadas com a seleção de soluções.

Novamente a interpretabilidade não teve grande variação. E a diferença entre a acurácia média do teste entre as duas baterias foi de 32,8.

Assim como foi feito com a base BJGF na subseção 4.2.1, foi tirada a média entre as 2 baterias de cada experimento e o resultado foi sintetizado na Tabela 13.

Tabela 11 - Testes utilizando o (2+2)M-PAES na base ele-2 SEM seleção

Métricas	Bateria 1			Bateria 2		
	Treino	Teste	Inter.	Treino	Teste	Inter.
Min.	276.0	292.9	0.4397	258.2	325.2	0.4728
1º Qu.	303.2	345.2	0.5126	288.6	361.8	0.5289
Mediana	313.7	362.7	0.5446	301.6	379.0	0.5497
Média	313.4	365.1	0.5465	301.4	388.2	0.5501
3º Qu.	323.1	381.2	0.5836	312.9	409.0	0.5743
Max.	357.5	442.9	0.6609	356.4	509.1	0.6345
D.T.M	440.746889			345.003618		
D.M.S	4.774033			3.794854		
N. Sol.	93.650000			92.610000		

Tabela 12 - Testes utilizando o (2+2)M-PAES na base ele-2 COM seleção

Métricas	Bateria 1			Bateria 2		
	Treino	Teste	Inter.	Treino	Teste	Inter.
Min.	254.5	336.3	0.4447	257.9	306.9	0.4602
1º Qu.	275.1	361.0	0.5327	278.1	338.0	0.5313
Mediana	289.4	387.2	0.5576	288.8	354.1	0.5587
Média	292.2	389.9	0.5620	293.5	357.1	0.5636
3º Qu.	306.2	414.8	0.5976	303.7	371.2	0.5953
Max.	359.6	468.3	0.6688	363.6	446.0	0.6775
D.T.M	402.476206			308.284594		

D.M.S	4.770407	3.824342
N. Sol.	88.390000	91.610000

Tabela 13 - Média das duas baterias da base ele-2 sem e com seleção

Métricas	Sem seleção			Com seleção		
	Treino	Teste	Inter.	Treino	Teste	Inter.
Min.	267,1	309,05	0,45625	256,2	321,6	0,45245
1º Qu.	295,9	353,5	0,52075	276,6	349,5	0,532
Mediana	304,65	370,85	0,54715	289,1	370,65	0,55815
Média	307,4	376,65	0,5483	292,85	373,5	0,5628
3º Qu.	318	395,1	0,57895	304,95	393	0,59645
Max.	356,95	476	0,6477	361,6	457,15	0,67315
D.T.M	392,8752525			355,3804		
D.M.S	4,2844435			4,2973745		
N. Sol.	93,13			90		

As tabelas mostram que o algoritmo sem seleção em média foi um pouco melhor na interpretabilidade e em compensação foi pior na acurácia. Isso é faz sentido já que acurácia e interpretabilidade são objetivos conflitantes. Isso não apareceu tão explicitamente na média das baterias da seção 4.2.1 Tabela 10 provavelmente devido à baixa diferença de interpretabilidade.

Apesar de uma das medidas de fronteira (D.M.S) ter obtido melhor resultado no algoritmo com seleção, esse resultado foi influenciado pelo N. Sol (três a menos) e por isso não reflete no resultado geral das medidas de fronteira. Devido a isso é possível concluir que o algoritmo sem seleção obteve melhor resultado nas medidas de fronteira. A D.T.M foi aproximadamente 37 unidades melhor e obteve 3 soluções a mais. Apesar do algoritmo com seleção ter se saído pior nas medidas de fronteira e levemente pior na interpretabilidade, ele obteve melhores resultados de acurácia tanto no treinamento quanto no teste.

4.2.1.2 Concrete

Para essa base foram feitas 2 baterias, uma para o algoritmo com seleção e outra sem a seleção de soluções e os resultados obtidos encontram-se na Tabela 14.

Tabela 14 - Bateria de resultado da base concrete com e sem seleção

Métricas	Sem seleção			Com seleção		
	Treino	Teste	Inter.	Treino	Teste	Inter.
Min.	7.621	8.403	0.3683	7.301	8.682	0.3606
1º Qu.	8.113	9.609	0.4907	7.892	9.524	0.4718

Mediana	8.608	10.133	0.5226	8.270	9.896	0.5072
Média	8.640	10.098	0.5177	8.340	9.909	0.4995
3º Qu.	9.110	10.536	0.5459	8.701	10.218	0.5242
Max.	10.307	11.659	0.6182	9.912	11.308	0.6016
D.T.M	4.00752480			4.23356047		
D.M.S	0.06203698			0.07275359		
N. Sol.	77.05			74.06		

Os resultados mostram que o algoritmo com seleção se saiu melhor em relação ao sem solução em quase todas as métricas. Apesar da bateria rodar cada algoritmo cem vezes é importante notar que nos testes passados houveram baterias do algoritmo sem seleção que foram melhores que os algoritmo com seleção. Se observarmos a bateria 3 da Tabela 6 e a bateria 3 da Tabela 8 e fossemos tirar alguma conclusão poderíamos dizer que o sem seleção é bem melhor inclusive na acurácia, entretanto no resultado final mostrado na Tabela 10 com a média das 5 baterias, foi observado que a seleção de soluções melhorou a acurácia. Por isso não é possível somente com uma bateria dizer que para essa base é melhor usar o algoritmo com seleção de soluções.

Podemos observar também que todos os valores foram levemente melhores.

4.2.1.3 Wankara

Apenas duas baterias foram feitas para essa base, uma rodando o algoritmo sem a seleção da proposta e outra com a seleção da proposta. Os resultados se encontram na Tabela 15.

Tabela 15 - Testes utilizando o (2+2)M-PAES na base *wankara*

Métricas	Sem seleção			Com seleção		
	Treino	Teste	Inter.	Treino	Teste	Inter.
Min.	2.489	2.772	0.3421	2.612	2.763	0.3098
1º Qu.	2.751	3.010	0.4216	2.807	3.082	0.4116
Mediana	2.836	3.194	0.4478	2.918	3.197	0.4436
Média	2.876	3.199	0.4438	2.937	3.225	0.4363
3º Qu.	2.996	3.331	0.4711	3.025	3.339	0.4638
Max.	3.344	3.820	0.5182	3.810	4.145	0.5236
D.T.M	4.48960543			5.03749805		
D.M.S	0.04862525			0.05513447		
N. Sol.	93.96			92.93		

Os resultados mostram que o algoritmo sem seleção foi melhor na maioria das métricas de interpretabilidade e acurácia, mas se for observada a diferença entre os valores obtidos em cada uma das métricas, pode-se perceber que a diferença é muito pequena. No teste sem seleção, a mediana resultante foi de 3.194 enquanto com seleção foi de 3.197. Isso significa que os elementos do meio da amostragem são praticamente iguais. A diferença de 0.06 na média pode ter sido causada pelo valor máximo de acurácia encontrada no algoritmo com seleção, pois a diferença entre a acurácia máxima encontrada no algoritmo sem seleção e a encontrada no algoritmo com seleção foi de aproximadamente 0.3. Este é um valor alto se for comparado aos demais valores de diferença das outras métricas de acurácia no teste.

A interpretabilidade média foi melhor no algoritmo com seleção, mas também não foi expressiva. A dispersão da fronteira de Pareto no algoritmo com seleção foi ligeiramente melhor.

4.2.1.4 Mortgage

Para essa base foram feitas 4 baterias, 2 com seleção e 2 sem soluções. A Tabela 16 foi testada sem a seleção da proposta. Não houve muita variação entre as duas baterias. Mas percebemos que a bateira 1 que obteve pior média de acurácia no treino, obteve melhor acurácia na média do teste.

Tabela 16 - Testes utilizando o (2+2)M-PAES na base *mortgage* SEM seleção

Métricas	Bateria1			Bateria2		
	Treino	Teste	Inter.	Treino	Teste	Inter.
Min.	0.2934	0.3381	0.4155	0.2924	0.3485	0.4005
1º Qu.	0.3398	0.4170	0.4710	0.3302	0.4046	0.4675
Mediana	0.3594	0.4438	0.4947	0.3496	0.4409	0.4905
Média	0.3673	0.4503	0.4920	0.3577	0.4549	0.4918
3º Qu.	0.3841	0.4763	0.5117	0.3792	0.4871	0.5177
Max.	0.5638	0.6391	0.5670	0.5873	0.8009	0.5690
D.T.M	0.652863746			0.587095376		
D.M.S	0.008066263			0.007539224		
N. Sol.	91.19			89.93		

A Tabela 17 é o resultado do teste com a seleção.

Tabela 17 - Testes utilizando o (2+2)M-PAES na base *mortgage* COM seleção

Métricas	Bateria 1			Bateria 2		
	Treino	Teste	Inter.	Treino	Teste	Inter.
Min.	0.3129	0.3482	0.3938	0.3092	0.3455	0.3942
1º Qu.	0.3431	0.3986	0.4700	0.3468	0.3955	0.4515
Mediana	0.3604	0.4407	0.4927	0.3654	0.4154	0.4752
Média	0.3670	0.4480	0.4911	0.3708	0.4326	0.4751
3º Qu.	0.3815	0.4765	0.5181	0.3868	0.4558	0.4976
Max.	0.6117	0.6506	0.5649	0.5347	0.6601	0.5466
D.T.M	0.94210051			0.657058288		
D.M.S	0.01046519			0.008490584		
N. Sol.	92.12			88.97		

Mesmo com um N. Sol. maior a bateria 1 conseguiu ter uma D.M.S maior que a bateria 2. Isso aconteceu, pois, o valor de D.T.M da bateria 1 foi bem maior que o da bateria 2.

Repetindo o que foi feito em bases que foram realizadas mais duas baterias totais, foi sintetizado em na Tabela 18 a média obtida nas baterias sem a implementação da seleção e com a implementação da seleção.

Tabela 18 - Média das duas baterias da base *mortgage* sem e com seleção

Métricas	Sem seleção			Com seleção		
	Treino	Teste	Inter.	Treino	Teste	Inter.
Min.	0,2929	0,3433	0,4080	0,3110	0,3468	0,3940
1º Qu.	0,3349	0,4108	0,4692	0,3449	0,3970	0,4607
Mediana	0,3545	0,4423	0,4926	0,3629	0,4280	0,4839
Média	0,3625	0,4526	0,4919	0,3689	0,4403	0,4831
3º Qu.	0,3816	0,4817	0,5147	0,3841	0,4661	0,5078
Max.	0,5755	0,72	0,5680	0,5732	0,6553	0,5557
D.T.M	0,619979561			0,799729399		
D.M.S	0,007802744			0,009477887		
N. Sol.	90,56			90,54		

O treinamento sorteando entre todas as soluções do arquivo (sem seleção) foi melhor que o treinamento sorteando entre a metade das soluções do arquivo (com seleção) e isso ocasionou novamente em uma melhoria na acurácia média do teste. Houve também uma pequena melhora na interpretabilidade média. A D.T.M e a D.M.S foram melhores apesar do N. Sol ser muito parecido.

Capítulo 5

CONCLUSÃO E TRABALHOS FUTUROS

5.1 Conclusão

Este trabalho apresentou um modo diferente de selecionar as soluções que sofrerão as operações genéticas. Em Cococcioni et al. (2007) as soluções são escolhidas aleatoriamente entre todas do arquivo e neste trabalho elas foram escolhidas previamente tendo como objetivo a melhora da dispersão da fronteira de Pareto. Os resultados médios obtidos com os testes em 5 bases diferentes mostraram que a D.T.M melhorou em 3 bases, a D.M.S melhorou em 4 bases e o número de soluções melhorou em uma base. Se contarmos que cada base tem a possibilidade de melhorar 3 métricas de fronteira de Pareto então o máximo que seria possível melhorar são 15 métricas. Dessas 15, 8 foram melhoradas. Então pode-se concluir que o método proposto, em média, não piora a dispersão da fronteira. Para comparar a acurácia e interpretabilidade foram utilizadas somente as médias obtidas nos testes de cada base. Das 5 bases analisadas, a acurácia mostrou-se pior em somente uma base, sendo que não foi melhor por uma diferença de 0.02. Então conclui-se que em média a solução proposta obtém melhores acurácias nos testes. A interpretabilidade média foi melhor em 3 das 5 bases, porém em todas as vezes na qual ela foi melhor e pior, a diferença foi inexpressiva. Então não houve diferença entre o método da proposta e o método apresentado em Cococcioni et al. (2007). Por fim, conclui-se que não houve ganho na dispersão das fronteiras e nem na interpretabilidade, porém a acurácia se mostrou em média melhor com seleção das soluções proposta nesse trabalho.

5.2 Trabalhos Futuros

Como sugestão de trabalhos futuros está a alteração dos parâmetros de seleção de soluções, o teste em mais bases de dados e a implementação da seleção proposta deste trabalho nas variações do algoritmo base (1+1) PAES. O algoritmo seleciona somente metade da fronteira, seria interessante também realizar testes selecionando mais e menos soluções para tentar descobrir se existe uma proporção de seleção que melhore os valores dos testes. Em algumas das bases testadas foram realizadas somente 2 baterias (uma com seleção e outra sem), seria bom fazer mais baterias nessas bases e expandir para outras bases que a KEEL disponibiliza.

REFERÊNCIAS

ABADEH, Mohammad Saniee; MOHAMADI, Hamid; HABIBI, Jafar. Design and analysis of genetic fuzzy systems for intrusion detection in computer networks. **Expert Systems with Applications**, [s. l.], v. 38, n. 6, p. 7067–7075, 2011. Disponível em: <http://linkinghub.elsevier.com/retrieve/pii/S0957417410013692>. Acesso em: 28 jun. 2017.

CARIDÁ, V. F.; MORANDIN, O.; TUMA, C. C. M. Approaches of fuzzy systems applied to an AGV dispatching system in a FMS. **The International Journal of Advanced Manufacturing Technology**, [s. l.], v. 79, n. 1–4, p. 615–625, 2015. Disponível em: <http://link.springer.com/10.1007/s00170-015-6833-8>. Acesso em: 22 maio. 2017.

CINTRA, Marcos E.; MONARD, Maria C.; CAMARGO, Heloisa de Arruda. Using fuzzy formal concepts in the genetic generation of fuzzy systems. In: 2012 IEEE INTERNATIONAL CONFERENCE ON FUZZY SYSTEMS 2012, Brisbane - Australia. **Anais**... Brisbane - Australia: IEEE, 2012. Disponível em: <http://ieeexplore.ieee.org/document/6251310/>. Acesso em: 22 mar. 2017.

COCOCCIONI, Marco et al. A Pareto-based multi-objective evolutionary approach to the identification of Mamdani fuzzy systems. **Soft Computing**, [s. l.], v. 11, n. 11, p. 1013–1031, 2007. Disponível em: <http://link.springer.com/10.1007/s00500-007-0150-6>. Acesso em: 7 fev. 2019.

CORDÓN, O. et al. Ten years of genetic fuzzy systems: current framework and new trends. **Fuzzy Sets and Systems**, [s. l.], v. 141, n. 1, p. 5–31, 2004. Disponível em: <http://linkinghub.elsevier.com/retrieve/pii/S0165011403001118>. Acesso em: 28 jun. 2017.

CORNE, David W. et al. PESA-II: Region-based Selection in Evolutionary Multiobjective Optimization. **PROCEEDINGS OF THE GENETIC AND EVOLUTIONARY COMPUTATION CONFERENCE (GECCO'2001**, [s. l.], p. 283--290, 2001. Disponível em: <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.10.2194>. Acesso em: 7 fev. 2019.

CORNE, David W.; KNOWLES, Joshua D.; OATES, Martin J. The Pareto Envelope-Based Selection Algorithm for Multiobjective Optimization. In: [s.l.] : Springer, Berlin, Heidelberg, 2000. p. 839–848.

DE OLIVEIRA, J. V. Semantic constraints for membership function optimization. **IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans**, [s. l.], v. 29, n. 1, p. 128–138, 1999. Disponível em: <http://ieeexplore.ieee.org/document/736369/>. Acesso em: 8 fev. 2019.

DEB, K. et al. A fast and elitist multiobjective genetic algorithm: NSGA-II. **IEEE Transactions on Evolutionary Computation**, [s. l.], v. 6, n. 2, p. 182–197, 2002. Disponível em: <http://ieeexplore.ieee.org/document/996017/>. Acesso em: 7 fev. 2019.

DEB, Kalyanmoy.; KALYANMOY, Deb. **Multi-objective optimization using evolutionary algorithms**. [s.l.] : John Wiley & Sons, 2001. Disponível em: <https://dl-acm-org.ez31.periodicos.capes.gov.br/citation.cfm?id=559152>. Acesso em: 7 fev. 2019.

DURILLO, Juan J.; NEBRO, Antonio J. jMetal: A Java framework for multi-objective optimization. **Advances in Engineering Software**, [s. l.], v. 42, n. 10, p. 760–771, 2011. Disponível em: <https://linkinghub.elsevier.com/retrieve/pii/S0965997811001219>. Acesso em: 7 fev. 2019.

ELHAG, Salma et al. On the combination of genetic fuzzy systems and pairwise learning for improving detection rates on Intrusion Detection Systems. **Expert Systems with Applications**, [s. l.], v. 42, n. 1, p. 193–202, 2015. Disponível em: <http://linkinghub.elsevier.com/retrieve/pii/S0957417414004783>. Acesso em: 28 jun. 2017.

FAZZOLARI, Michela et al. A Review of the Application of Multiobjective Evolutionary Fuzzy Systems: Current Status and Further Directions. **IEEE Transactions on Fuzzy Systems**, [s. l.], v. 21, n. 1, p. 45–65, 2013. a. Disponível em: <http://ieeexplore.ieee.org/document/6204330/>. Acesso em: 7 fev. 2019.

FAZZOLARI, Michela et al. A study on the application of instance selection techniques in genetic fuzzy rule-based classification systems: Accuracy-complexity trade-off. **Knowledge-Based Systems**, [s. l.], v. 54, p. 32–41, 2013. b. Disponível em: <http://linkinghub.elsevier.com/retrieve/pii/S095070511300213X>. Acesso em: 2 jul. 2014.

FONSECA, Carlos M.; FLEMING, Peter J. Genetic Algorithms for Multiobjective Optimization: FormulationDiscussion and Generalization. In: PROCEEDINGS OF THE FIFTH INTERNATIONAL CONFERENCE 1993, San Mateo - CA - USA. **Anais**... San Mateo - CA - USA: Morgan Kaufmann Publishers Inc., 1993. Disponível em: <http://dl.acm.org/citation.cfm?id=645513.657757>. Acesso em: 29 fev. 2016.

GACTO, M. J.; ALCALÁ, R.; HERRERA, F. Interpretability of linguistic fuzzy rule-based systems: An overview of interpretability measures. **Information Sciences**, [s. l.], v. 181, n. 20, p. 4340–4360, 2011. Disponível em: <http://linkinghub.elsevier.com/retrieve/pii/S0020025511001034>. Acesso em: 28 jun. 2017.

GUILLAUME, S. Designing fuzzy inference systems from data: An interpretability-oriented review. **IEEE Transactions on Fuzzy Systems**, [s. l.], v. 9, n. 3, p. 426–443, 2001. Disponível em: <http://ieeexplore.ieee.org/document/928739/>. Acesso em: 7 fev. 2019.

HAJELA, P.; LIN, C. Y. Genetic search strategies in multicriterion optimal design. **Structural Optimization**, [s. l.], v. 4, n. 2, p. 99–107, 1992. Disponível em: <http://link.springer.com/10.1007/BF01759923>. Acesso em: 7 fev. 2019.

HORN, J.; NAFPLIOTIS, N.; GOLDBERG, D. E. A niched Pareto genetic algorithm for multiobjective optimization. In: PROCEEDINGS OF THE FIRST IEEE CONFERENCE ON EVOLUTIONARY COMPUTATION. IEEE WORLD CONGRESS ON COMPUTATIONAL INTELLIGENCE 2002, **Anais**... : IEEE, 2002. Disponível em: <http://ieeexplore.ieee.org/document/350037/>. Acesso em: 7 fev. 2019.

ISHIBUCHI A'*, Hisao; MURATA, Tadahiko; TIIRK~EN, I. B. **Single-objective and two-objective genetic algorithms for selecting linguistic rules for pattern classification problemsFuzzy Sets and Systems**. [s.l: s.n.]. Disponível em: <https://sci2s.ugr.es/sites/default/files/files/TematicWebSites/GeneticFuzzySystems/1997-Ishibuchi-Single-objective_and_two_objective_genetic_algorithms_for_selecting_linguistic_rules_for_pattern_classification_problems.pdf>. Acesso em: 7 fev. 2019.

ISHIBUCHI, Hisao; NOJIMA, Yusuke. Evolutionary multiobjective optimization for the design of fuzzy rule-based ensemble classifiers. **International Journal of Hybrid Intelligent Systems**, [s. l.], v. 3, n. 3, p. 129–145, 2006. Disponível em: <http://www.medra.org/servlet/aliasResolver?alias=iospress&doi=10.3233/HIS-2006-3302>. Acesso em: 7 fev. 2019.

KIM, Eun-Hu; OH, Sung-Kwun; PEDRYCZ, Witold. Reinforced rule-based fuzzy models: Design and analysis. **Knowledge-Based Systems**, [s. l.], v. 119, p. 44–58, 2017. Disponível em: <http://linkinghub.elsevier.com/retrieve/pii/S0950705116304853>. Acesso

em: 28 jun. 2017.

KNOWLES, Joshua D.; CORNE, David W. Approximating the Nondominated Front Using. [s. l.], v. 8, n. 2, p. 149–172, 2000. Disponível em: <https://www.macs.hw.ac.uk/~dwcorne/newkc.dvi.pdf>. Acesso em: 7 fev. 2019.

KOHL, Matthias. **Introduction to statistical data analysis with R**. [s.l: s.n.]. Disponível em: <https://bookboon.com/en/introduction-to-statistical-data-analysis-with-r-ebook>. Acesso em: 7 fev. 2019.

LAUMANNS, M.; ZITZLER, E.; THIELE, L. A unified model for multi-objective evolutionary algorithms with elitism. In: PROCEEDINGS OF THE 2000 CONGRESS ON EVOLUTIONARY COMPUTATION. CEC00 (CAT. NO.00TH8512) 2000, **Anais**... : IEEE, 2000. Disponível em: <http://ieeexplore.ieee.org/document/870274/>. Acesso em: 7 fev. 2019.

LAUMANNS, Marco; ZITZLER, Eckart; THIELE, Lothar. On The Effects of Archiving, Elitism, and Density Based Selection in Evolutionary Multi-objective Optimization. In: [s.l: s.n.]. p. 181–196.

MITCHELL, Melanie (Computer scientist); MELANIE. **An introduction to genetic algorithms**. [s.l.] : MIT Press, 1996. Disponível em: <https://dl-acm-org.ez31.periodicos.capes.gov.br/citation.cfm?id=230231>. Acesso em: 7 fev. 2019.

MURATA, T.; ISHIBUCHI, H. MOGA: multi-objective genetic algorithms. In: INTERNATIONAL CONFERENCE ON EVOLUTIONARY COMPUTATION 1995, **Anais**... : IEEE, 1995. Disponível em: <http://ieeexplore.ieee.org/document/489161/>. Acesso em: 22 fev. 2019.

NICOLETTI, Maria do Carmo; CAMARGO, Heloisa De Arruda. **Fundamentos Da Teoria De Conjuntos Fuzzy**. [s.l: s.n.]. Disponível em: <https://books.google.com.br/books/about/Fundamentos_Da_Teoria_De_Conjuntos_Fuzzy.html?id=vGVHbwAACAAJ&redir_esc=y>. Acesso em: 7 fev. 2019.

PEDRYCZ, Witold; GOMIDE, Fernando. **An introduction to fuzzy sets : analysis and design**. [s.l.] : MIT Press, 1998. Disponível em: <https://mitpress.mit.edu/books/introduction-fuzzy-sets>. Acesso em: 8 fev. 2019.

PIMENTA, Adinovam H. M.; CAMARGO, Heloisa de Arruda. NSGA-DO: Non-Dominated Sorting Genetic Algorithm Distance Oriented. In: 2015 IEEE INTERNATIONAL CONFERENCE ON FUZZY SYSTEMS (FUZZ-IEEE) 2015, **Anais**... : IEEE, 2015. Disponível em: <http://ieeexplore.ieee.org/document/7338080/>. Acesso em: 15 maio. 2017.

RAHLF, Thomas. **Data visualisation with R : 100 examples**. [s.l: s.n.].

RIID, Andri; PREDEN, J?rgo-S?ren. Design of Fuzzy Rule-based Classifiers through Granulation and Consolidation. **Journal of Artificial Intelligence and Soft Computing Research**, [s. l.], v. 7, n. 2, 2017. Disponível em: <http://www.degruyter.com/view/j/jaiscr.2017.7.issue-2/jaiscr-2017-0010/jaiscr-2017-0010.xml>. Acesso em: 28 jun. 2017.

RIZA, Lala Septem et al. An Expert System for Diagnosis of Sleep Disorder Using Fuzzy Rule-Based Classification Systems. **IOP Conference Series: Materials Science and Engineering**, [s. l.], v. 185, p. 012011, 2017. Disponível em: <http://stacks.iop.org/1757-899X/185/i=1/a=012011?key=crossref.d5a86d67a20f3ec2fb6b19c3847452dd>. Acesso em: 28 jun. 2017.

SCHAFFER, J. David. Multiple Objective Optimization with Vector Evaluated Genetic Algorithms. **undefined**, [s. l.], 1985. Disponível em: <https://www.semanticscholar.org/paper/Multiple-Objective-Optimization-with-Vector-

Genetic-Schaffer/e51bc6ce7b1e19ff3f5b5386d2ca2b7e65eecfc3>. Acesso em: 8 fev. 2019.

SETNES, M.; BABUSKA, R.; VERBRUGGEN, H. B. Rule-based modeling: precision and transparency. **IEEE Transactions on Systems, Man and Cybernetics, Part C (Applications and Reviews)**, [s. l.], v. 28, n. 1, p. 165–169, 1998. Disponível em: <http://ieeexplore.ieee.org/document/661100/>. Acesso em: 8 fev. 2019.

SILVERMAN, B. W. **DENSITY ESTIMATION FOR STATISTICS AND DATA ANALYSIS**. [s.l: s.n.]. Disponível em: <https://ned.ipac.caltech.edu/level5/March02/Silverman/paper.pdf>. Acesso em: 8 fev. 2019.

SRINIVAS, N.; DEB, Kalyanmoy. Muiltiobjective Optimization Using Nondominated Sorting in Genetic Algorithms. **Evolutionary Computation**, [s. l.], v. 2, n. 3, p. 221–248, 1994. Disponível em: <http://www.mitpressjournals.org/doi/10.1162/evco.1994.2.3.221>. Acesso em: 8 fev. 2019.

ZITZLER, Eckart; DEB, Kalyanmoy; THIELE, Lothar. Comparison of Multiobjective Evolutionary Algorithms: Empirical Results. **Evolutionary Computation**, [s. l.], v. 8, n. 2, p. 173–195, 2000. Disponível em: <http://www.mitpressjournals.org/doi/10.1162/106365600568202>. Acesso em: 21 fev. 2019.

ZITZLER, Eckart; LAUMANNS, Marco; THIELE, Lothar. **SPEA2: Improving the Strength Pareto Evolutionary Algorithm**. [s.l: s.n.]. Disponível em: <https://pdfs.semanticscholar.org/6672/8d01f9ebd0446ab346a855a44d2b138fd82d.pdf>. Acesso em: 8 fev. 2019.

ZITZLER, Eckart; THIELE, Lothar. An Evolutionary Algorithm for Multiobjective Optimization: The Strength Pareto Approach. In: COMPUTER ENGINEERING AND NETWORKS LABORATORY 1998, Zurich - Switzerland. **Anais**... Zurich - Switzerland Disponível em: <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.40.7696>. Acesso em: 8 fev. 2019.