# Descoberta de Regras de Conhecimento Utilizando Computação Evolutiva Multi-Objetivo

Rafael Giusti, Gustavo E.A.P.A. Batista (orientador)

Instituto de Ciências Matemáticas e de Computação
Universidade de São Paulo
{rgiusti,gbatista}@icmc.usp.br

**Resumo** A maioria dos sistemas de Aprendizado de Máquina – AM – visa classificadores com cobertura e precisão máximas. Embora essa tenha se mostrado uma boa abordagem para automatizar processos de tomada de decisão, nem sempre produz resultados tão bons quando o usuário busca modelos que apresentem outras propriedades. Assim, neste artigo é apresentada uma pesquisa de métodos de computação evolutiva multiobjetivo para a construção de regras de conhecimento individuais com base em critérios definidos pelo usuário, por meio de medidas de qualidade como precisão, Laplace, confiança etc. Neste trabalho é realizado também um estudo comparativo entre métodos de computação evolutiva voltados para problemas multiobjetivo e métodos de composição de *rankings*.

**Nível:** mestrado. **Conclusão:** 22/06/2010. **Banca:** orientador, Profa. Dra. Solange Oliveira Rezende (ICMC-USP), Prof. Dr. Estevam Rafael Hruschka Júnior (UFSCar). **Dissertação:** [1]. **Publicações:** [2,3].

## 1 Introdução

Algoritmos de classificação em geral são desenvolvidos com o objetivo principal de extrair classificadores genéricos e precisos. Essa abordagem é bastante eficiente quando o interesse reside na automatização do processo de tomada de decisão, mas pode ser insuficiente quando o usuário deseja adquirir para si o conhecimento extraído dos dados, pois desconsidera outras propriedades interessantes, além da cobertura e da precisão, como a dependência entre o corpo e a cabeça das regras, a descoberta de exceções etc.

Assim, torna-se interessante a descoberta de conhecimento por outros critérios, os quais possivelmente permitam que o usuário obtenha modelos de conhecimento que sejam mais úteis ou interessantes para si. Em Pila, A.D. [4] é proposto um método para construir regras de conhecimento individuais com base em diferentes critérios simultâneos definidos pelo usuário. A esses critérios o proponente dá o nome *propriedades específicas*, as quais são aferidas por medidas de qualidade de regra como precisão, novidade etc. A construção de regras com propriedades específicas foi abordada com a aplicação de um algoritmo de computação evolutiva baseado em composição de *rankings*. Durante a execução do projeto, da qual o primeiro autor deste artigo participou como aluno de iniciação científica, foi desenvolvido um ambiente e uma biblioteca de classes para descoberta de conhecimento com propriedades específicas, denominada ECLE – *Evolutionary Computing Learning Environment*.

Desde as primeiras aplicações a ECLE tem se mostrado eficaz na construção de regras com propriedades específicas [5,6,2] e também como um *framework* para construção de regras utilizando computação evolutiva. Apesar disso, fazia-se necessário explorar uma abordagem intrinsicamente multiobjetivo. Assim, os objetivos principais da Dissertação de Mestrado relacionada a este artigo são: (1) investigação de técnicas de otimização multiobjetivo, baseadas no conceito de eficiência de Pareto, para a construção de regras de conhecimento com propriedades específicas; (2) modificação da ECLE para a utilização de métodos de composição de *rankings* em um contexto de otimização multiobjetivo; e (3) a comparação da eficácia de métodos de composição de *rankings* contra a eficácia de métodos de computação evolutiva multiobjetivo baseados nos conceitos de eficiência de Pareto.

O restante deste artigo está organizado da seguinte forma. Na Seção 2 é feita uma discussão sobre aprendizado de máquina. Na Seção 3 são apresentados fundamentos de otimização. Na Seção 4 é feita uma breve apresentação a algoritmos evolutivos. Na Seção 5 os algoritmos de computação evolutiva implementados na ECLE são explicados. Na Seção 6 é descrito o plano de avaliação experimental e os resultados obtidos. Na Seção 7 é apresentada uma conclusão deste trabalho.

## 2 Aprendizado de Máquina

A premissa básica do *aprendizado de máquina* – AM – é que os dados contêm implicitamente conhecimento relacionado a algum processo que explica esses dados. Por exemplo, o histórico de um paciente pode conter conhecimento implícito que explique as condições médicas desse paciente. O papel de um sistema de aprendizado é construir um modelo capaz de explicar o processo relacionado aos dados, utilizando para isso métodos estatísticos que se baseiam em inferências sobre uma amostra dos dados [7].

Neste trabalho a amostra dos dados é representada no formato da *tabela atributo-valor*. Cada exemplo $E_i = (\mathbf{x}_i, y_i)$ é um vetor de valores $\mathbf{x}_i = (x_{i1}, \ldots, x_{iM})$ referente a um conjunto de atributos relacionados e um valor de classe $y_i$ relacionado ao atributo-classe. A premissa da indução sustenta que existe uma função desconhecida $f$, chamada *função-conceito*, tal que, para cada exemplo $E_i = (\mathbf{x}_i, y_i)$, tem-se $f(\mathbf{x}_i) = y_i$. O objetivo do aprendizado supervisionado é aproximar essa função conceito $f$.

Uma *regra de conhecimento* é um modelo de representação de hipóteses que promove a associação de uma classe $C_v \in \{C_1, \ldots, C_{Ncl}\}$ a um exemplo qualquer $E_i$, condicionada a um conjunto de restrições relacionadas aos atributos do conjunto de dados. Regras de conhecimento são geralmente representadas na forma $R$: *if* complexo *then class* = $C_v$, na qual $C_v$ é uma classe e complexo é uma série de conjunções de termos referentes aos atributos do conjunto de dados. O complexo é também chamado *corpo – body* ou $B$ – e a classe associada pela regra ao exemplo é também chamada *cabeça – head* ou $H$. Um exemplo seria a regra "*if* Umidade = Normal *then class* = Jogar", que poderia ser facilmente lida como "se a umidade do ar estiver normal, então o sujeito irá jogar tênis".

Definido um domínio qualquer, pode-se supor um grande número de regras de conhecimento que representam conceitos válidos desse domínio. Faz-se necessário, portanto, uma metodologia de avaliação de regras de conhecimento. Dados uma regra

$R: B \rightarrow H$ e um exemplo $E_i = (\mathbf{x}_i, y_i)$, pode-se testar $R$ contra $E_i$ verificando se os termos de $B$ são compatíveis com os valores de $\mathbf{x}_i$. Em caso positivo, diz-se que o corpo da regra *cobre* o exemplo e que a regra *prediz* que a classe do exemplo é $H$. Se a classe do exemplo de fato for $H$, então diz-se que a cabeça da regra cobre o exemplo.

Pode-se encarar o teste de uma regra contra um exemplo como a análise de dois eventos, $B$ e $H$, os quais indicam cobertura do corpo e da cabeça, respectivamente. Como há apenas dois valores possíveis para $H$ (cobertura ou não-cobertura) e para $B$ (idem) e como não se pode saber de antemão quais serão os resultados dos eventos $B$ e $H$ para qualquer teste, pode-se encarar $H$ e $B$ como duas variáveis aleatórias binárias. A *matriz de contingência* é uma tabela $2 \times 2$ que sintetiza as ocorrências de $H$ e $B$ para uma regra qualquer quando esta é avaliada contra um conjunto de dados. No diagrama a seguir é mostrada a estrutura geral da matriz de contingência quando descrita em termos de frequência relativa. Nela, $f_{hb}$ representa a frequência relativa da ocorrência simultânea dos eventos $H$ e $B$ e assim por diante.

| | $H$ | $\overline{H}$ | |
|---|---|---|---|
| $B$ | $f_{hb}$ | $f_{\overline{h}b}$ | $f_b$ |
| $\overline{B}$ | $f_{h\overline{b}}$ | $f_{\overline{h}\overline{b}}$ | $f_{\overline{b}}$ |
| | $f_h$ | $f_{\overline{h}}$ | 1 |

A matriz de contingência é a base do *framework* de Lavrač [8], permitindo definir muitas medidas de qualidade que estimam a utilidade de uma regra com relação a diferentes propósitos. Por exemplo, a medida de precisão, definida como a probabilidade condicional de cobertura da cabeça, dado que houve cobertura do corpo, pode ser descrita no *framework* de Lavrač como $Pre(R) = P(H|B) = f_{hb}/f_b$.

## 3 Conceitos de Otimização

Otimização é um processo que, dado um conjunto de objetivos e restrições, visa encontrar uma ou mais soluções que maximizem (ou minimizem) os objetivos e, ao mesmo tempo, respeitem as restrições impostas. Tipicamente, o problema de otimização é modelado como um conjunto de variáveis de decisão e uma função-objetivo. As *variáveis de decisão* são parâmetros relacionados ao problema em questão. A *função-objetivo* é uma função que associa as variáveis de decisão a um valor numérico. Uma *solução* é um vetor de valores associados às variáveis de decisão que descreve uma possível estratégia de resolução do problema de otimização, sendo uma *solução ótima* aquela para a qual a função objetivo assume o máximo ou mínimo valor possível. Soluções que não são ótimas são chamadas *sub-ótimas*. Uma solução é *factível* se os valores das variáveis de decisão obedecem às restrições do modelo.

Problemas nos quais existe uma única função-objetivo ou várias funções correlacionadas são chamados problemas de otimização *simples-objetivo*. Problemas de otimização que envolvem mais de uma função e pelo menos duas funções conflitantes – *i.e.*, otimizar uma tende a degenerar a outra e vice-versa – são chamados problemas de *otimização multiobjetivo*. Por exemplo, construir regras maximizando simultaneamente cobertura e precisão é um problema de otimização multiobjetivo, pois regras mais genéricas tendem a ser menos precisas e regras mais precisas tendem a ser mais específicas.

Dado um conjunto de soluções de um problema multiobjetivo, é possível detectar soluções sub-ótimas através do critério de dominância. Uma solução $s_i$ é dita *dominada* por uma solução $s_j$ se $s_i$ e não é melhor que $s_j$ em nenhum objetivo e $s_j$ é melhor que $s_i$ em pelo menos um objetivo.

Ao se detectar soluções dominadas, pode-se separar as soluções entre dominadas e não-dominadas. O conjunto de soluções não-dominadas é chamado *fronteira de Pareto*. Na Figura 1 são indicadas as posições que as soluções de dois conjuntos distintos ocupam no *espaço de objetivos*. Tanto em 1 (a) quanto em 1 (b), o "volume" cinza indica o espaço no qual podem ser encontradas soluções factíveis. As linhas contínuas indicam possíveis pontos ocupados por soluções que não podem ser dominadas por nenhuma solução factível. O conjunto dessas soluções é chamado fronteira Pareto-ótima.

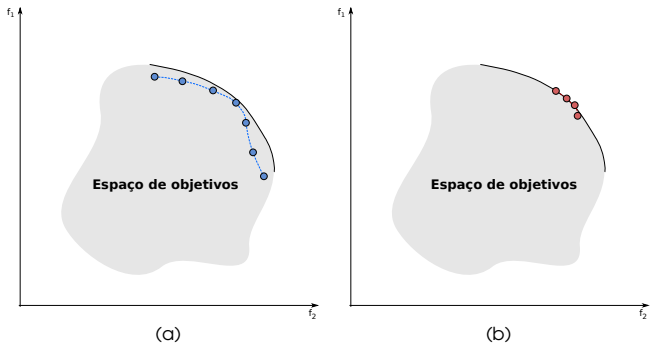


**Figura 1.** Fronteiras de Pareto.

A fronteira de Pareto pode ser extensa, contendo muitas soluções igualmente boas. Para selecionar uma delas, é necessário considerar novos critérios, o que tipicamente exige o auxílio de um humano. Por isso, um sistema de otimização multiobjetivo que produza um conjunto de soluções sub-ótimas, porém variadas e "razoavelmente" próximas da fronteira Pareto-ótima – Figura 1 (a) — pode ser preferível a um que produza um conjunto de soluções ótimas, porém muito semelhantes – Figura 1 (b) por dar maior liberdade de escolha a um eventual especialista.

## 4 Algoritmos Evolutivos

Algoritmos evolutivos são um caso particular de computação evolutiva que têm como característica principal a representação das soluções como indivíduos que competem em um espaço virtual para propagar suas características. Algoritmos evolutivos implementam um processo de seleção artificial que remete aos conceitos da teoria da evolução pela seleção natural [9]. A cada iteração $t = 0, 1, 2, ...$ o algoritmo considera uma população de indivíduos $P(t)$ que *representa* um conjunto de soluções factíveis. Para simular a competição entre os indivíduos, uma *função de avaliação* testa a eficácia da solução representada por cada indivíduo e atribui ao indivíduo um valor de *aptidão*. A

informação de aptidão é utilizada por um mecanismo de seleção estocástico, o *operador de seleção*, que seleciona de forma aleatória um conjunto de indivíduos privilegiando os melhores da população, compondo uma *população intermediária* $P'(t)$. A população intermediária é então aleatoriamente modificada, normalmente com a aplicação de um operador de *crossover* seguido da aplicação de um operador de mutação, e uma nova população $P(t+1)$ é construída. Esse ciclo de avaliação, seleção e modificação continua até que um indivíduo — ou um conjunto de indivíduos — que representa a solução ótima seja encontrado ou algum outro critério de parada seja satisfeito [10].

Neste trabalho, os indivíduos são regras seguem a representação *Michigan*, na qual cada indivíduo do algoritmo evolutivo é uma regra de conhecimento independente das demais. Essa representação foi escolhida por não haver interesse em utilizar todas as regras como um único classificador. A função de avaliação, por sua vez, é selecionada pelo usuário como uma medida de qualidade de regra ou uma combinação de medidas de qualidade de regra, descritas em termos das variáveis do *framework* de Lavrač, de modo que o sistema buscará construir regras de conhecimento que atendam às propriedades que o usuário deseja. Uma descrição detalhada desses e dos outros elementos dos algoritmos evolutivos implementados na ECLE pode ser encontrada em [1].

## 5 A Biblioteca de Classes ECLE

A ECLE – *Evolutionary Computing Learning Environment* – é uma biblioteca de classes e simultaneamente um ambiente de construção de regras de conhecimento. A ECLE trata a construção de regras de conhecimento com diversas propriedades específicas como um problema de otimização multiobjetivo, no qual cada objetivo de otimização é uma medida de qualidade de regra definida pelo usuário. Uma solução para esse problema é uma regra de conhecimento válida. Portanto, solucionar esse problema de otimização multiobjetivo significa encontrar uma ou mais regras de conhecimento que apresentem em razoável grau as propriedades específicas desejadas. A ECLE pode empregar três diferentes algoritmos de computação evolutiva para a resolução desse problema de otimização multiobjetivo, explicados a seguir.

### 5.1 Composição de *Rankings*

Composição de *rankings* é uma forma de converter um problema de otimização multiobjetivo em um problema de otimização simples-objetivo. Um *ranking* é uma ordem parcial sobre conjunto de elementos, como uma ordem imposta sobre um conjunto de indivíduos pelo critério de altura. A posição de cada indivíduo no *ranking* é chamada *rank*. Quando se possui um conjunto de elementos classificados em diferentes *rankings*, cada um formado com base em um critério distinto, pode-se construir um *ranking* composto atribuindo-se a cada elemento um *rank* que reflete seu desempenho com relação aos demais elementos quando se considera todos os critérios simultaneamente.

Dado um problema de otimização multiobjetivo com $m$ objetivos distintos para o qual se tem um conjunto de soluções $S_1, \ldots, S_n$, pode-se definir inicialmente $m$ *rankings* normalizados (*i.e.*, tal que a somatória dos *ranks* é a somatória de uma PA de razão 1 para o mesmo número de elementos) $R_1, \ldots, R_m$, sendo cada *ranking* $R_i$

ordenado pela função-objetivo $f_i$. Em seguida, para cada solução $S_i, i \in [1, n]$, cujos *ranks* são dados por $r_{1i}, \ldots, r_{mi}$, é definido um valor de *rank* composto $r_i$. Um exemplo desse procedimento é dado no diagrama abaixo. Inicialmente, todas as soluções são *rankeadas* segundo dois objetivos, quantificados pelas funções a serem maximizadas $f_1(\mathbf{x})$ e $f_2(\mathbf{x})$. Com base nisso são construídos os *rankings* $R_1$ e $R_2$. Então, para cada solução é calculado um *rank* composto equivalente ao valor médio dos *ranks* obtidos por essa solução nos *rankings* iniciais. O resultado final é o *ranking* $R_c$ que é normalizado de modo a produzir o *ranking* final $R_f$.

| | $S_1$ | $S_2$ | $S_3$ | $S_4$ | $S_5$ | $S_6$ | $S_7$ | $S_8$ |
|---|---|---|---|---|---|---|---|---|
| $f_1$ | 10.3 | 9.6 | 9.8 | 9.6 | 8.0 | 8.7 | 8.7 | 7.1 |
| $R_1$ | 1 | 3.5 | 2 | 3.5 | 7 | 5.5 | 5.5 | 8 |
| $f_2$ | 9.0 | 9.8 | 6.6 | 7.3 | 5.6 | 5.6 | 5.1 | 4.7 |
| $R_2$ | 2 | 1 | 4 | 3 | 5.5 | 5.5 | 7 | 8 |
| $R_c$ | 1.5 | 2.25 | 3 | 3.25 | 6.25 | 5.5 | 6.25 | 8 |
| $R_f$ | 1 | 2 | 3 | 4 | 6.5 | 5 | 6.5 | 8 |

É importante ressaltar, entretanto, que a composição de *rankings* não substitui com perfeição a comparação das soluções por dominância. Por exemplo, no diagrama acima observa-se em $R_1$ e $R_2$ que não existe relação de dominância entre $S_1$ e $S_2$. Apesar disso, o *ranking* final sugere que a solução $S_1$ seria superior à solução $S_2$.

Na ECLE, a composição de *rankings* é utilizada para implementar uma função de avaliação na qual indivíduos com menor *rank* composto recebem maior valor de aptidão. A ECLE implementa cinco métodos de composição de *rankings*: média aritmética, média harmônica, mediana, o *ranking* de Condorcet e o *ranking* recíproco, acrescentando um mecanismo de manutenção de diversidade [1].

### 5.2 VEGA

O VEGA – *Vector Evaluated Genetic Algorithm* – é uma simples extensão do algoritmo evolutivo típico para problemas de otimização simples-objetivo [11]. Se o número de funções-objetivo a serem otimizadas é $M$, a cada iteração o VEGA particiona a população inicial $P(t)$ em $M$ subpopulações disjuntas $P_1(t), \ldots, P_M(t)$ de forma aleatória. Cada indivíduo é avaliado por uma única função-objetivo específica para a partição em que ele se encontra. Isto é, se as funções-objetivo são $f_1(\mathbf{x}), \ldots, f_M(\mathbf{x})$, então os indivíduos da partição $P_i(t)$ são avalidos pela função-objetivo $f_i(\mathbf{x})$.

Após a etapa de avaliação, o operador de seleção é aplicado em cada partição separadamente, selecionando indivíduos para compor a população intermediária $P'(t)$. É importante que o operador de seleção seja aplicado isoladamente em cada partição, pois como as funções-objetivo podem ser bastante distintas em termos de escala, distribuição e até mesmo objetivo – maximização ou minimização – os valores de aptidão dos indivíduos em partições distintas não são comparáveis. Definida a população intermediária $P'(t)$, o VEGA aplica os operadores de *crossover* e mutação como no algoritmo evolutivo típico, definindo a população seguinte $P(t+1)$ e reiniciando todo o processo na iteração seguinte. O ciclo continua até que a convergência seja alcançada ou outro critério de parada seja satisfeito. A ECLE implementa o algoritmo VEGA acrescentando um mecanismo de manutenção de diversidade.

### 5.3 NSGA-II

Dado um conjunto de soluções qualquer, se as soluções contidas na fronteira de Pareto forem temporariamente removidas, pode-se verificar uma "segunda" fronteira de Pareto, composta pelas soluções que são dominadas apenas por aquelas presentes na "primeira" fronteira de Pareto. Repetindo-se esse procedimento até que não restem soluções, obtém-se uma ordenação por não-dominância – *non-dominated sorting*.

O conceito de ordenação por não-dominância é o princípio do *Elitist Non-dominated Sorting Genetic Algorithm* – NSGA-II [12]. De forma sucinta, o NSGA-II é um algoritmo de computação evolutiva multiobjetivo elitista que procura selecionar tantas fronteiras de Pareto "mais externas", resultantes da ordenação por não-dominância, quanto possível. Dada uma população $P(t)$ de tamanho $N$, a primeira etapa do processo de aplicação do NSGA-II consiste em, com o uso de um operador genético de seleção, construir uma população intermediária $P'(t)$ de $P(t)$ e submeter os indivíduos de $P$ – doravante o parâmetro $(t)$ será omitido – a operadores de *crossover*, originando assim uma sub-população $Q$ composta por $N$ *offsprings*. Em seguida, as populações $P$ e $Q$ são unidas, gerando uma nova subpopulação $R$, de tamanho $2 \times N$, que é então ordenada por critério de não-dominância. As "primeiras" fronteiras de Pareto resultantes dessa ordenação por não-dominância são transferidas para a população intermediária $Z$, sendo aplicado um operador de manutenção de diversidade para selecionar apenas os indivíduos mais diversos da "última" fronteira que "couber" em $Z$. Então, a população $Z$ é submetida a um operador de seleção de torneio modificado que seleciona preferencialmente soluções pertencentes a fronteiras de Pareto "mais externas" da ordenação por não-dominância em $R$. A partir daí, o algoritmo evolutivo executa normalmente os passos de *crossover* e mutação.

## 6 Avaliação Experimental

Até o início das atividades deste projeto, a única abordagem utilizada pela ECLE para construir regras com múltiplas propriedades específicas consistia em converter um problema de otimização multiobjetivo – *i.e.*, construir regras que maximizem diversas medidas de qualidade potencialmente conflitantes – em um problema de otimização simples-objetivo por meio de composição de *rankings*. Tal conversão permite que o problema de otimização multiobjetivo seja solucionado com a aplicação de um algoritmo de computação evolutiva simples-objetivo e, como demonstrado por [4], é capaz de construir regras de conhecimento que atendam aos critérios estabelecidos pelo usuário. Apesar disso, trabalhos anteriores com a ECLE e métodos de otimização por composição de *rankings* levaram ao questionamento de que tal abordagem, ainda que oferecendo bons resultados, poderia não ser capaz de fornecer os melhores resultados possíveis. Sendo assim, no trabalho de mestrado relacionado a este artigo, teve-se como um dos objetivos implementar métodos de computação evolutiva intrinsecamente multiobjetivo. Foram implementados os métodos VEGA e NSGA-II e foram aplicadas modificações aos métodos de composição de *rankings* para que a avaliação fosse realizada segundo os critérios de eficiência de Pareto.

O primeiro passo da avaliação experimental dos métodos implementados consistiu em definir critérios objetivos de avaliação. Para que se entenda melhor o problema

envolvido, considere os dois conjuntos apresentados Figura 2. O volume acinzentado representa os pontos que as soluções factíveis do problema ocupam no espaço de objetivos, enquanto a linha contínua contornando a "face" superior direita do volume representa o espaço ocupado pelas soluções da fronteira Pareto-ótima. Os círculos representam soluções de um conjunto de soluções qualquer (por exemplo, soluções produzidas pelo método NSGA-II), ao passo que os quadrados representam soluções de um outro conjunto de soluções arbitrário (por exemplo, soluções produzidas pelo método VEGA). As linhas tracejadas que percorrem as soluções mais próximas da fronteira Pareto-ótima são a interpolação das fronteiras de Pareto definidas pelos dois conjuntos.

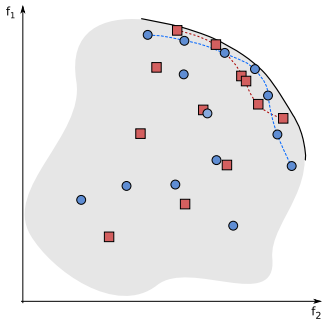


**Figura 2.** Dois conjuntos de soluções com fronteiras de Pareto destacadas.

Deve-se fazer então o seguinte questionamento: qual dos conjuntos fornece as opções que poderiam ter mais serventia para o usuário, considerando-se que o interesse do usuário no processo de otimização multiobjetivo é ser capaz de escolher uma ou mais soluções dentre um conjunto de soluções eficientes e diversas? Para lidar com esse problema decidiu-se avaliar os dois critérios seguintes.

1. O principal fator de comparação é a dominância. Dados dois conjuntos de soluções $C_1$ e $C_2$ tal que as soluções de $C_1$ dominem um grande número de soluções em $C_2$ e a recíproca não seja verdadeira, então $C_1$ deve ser considerado superior. Caso contrário, deve haver uma tendência a se declarar um empate entre os conjuntos;
2. A distribuição das soluções deve ser levada com consideração, especialmente na fronteira de Pareto. Conjuntos mais diversos são preferíveis a conjuntos com soluções muito próximas.

Foram verificadas empiricamente diversas medidas e o quanto elas eram representativas em diferentes cenários de avaliação experimental. Ao final, decidiu-se utilizar as métricas $C'$ *rankeada* e distância euclidiana total média.

**Métrica de Dominância $C'$ *rankeada*:** Dados $N$ conjuntos de soluções no espaço multiobjetivo, $C_1, C_2, \ldots, C_N$, inicialmente são descartadas as soluções dominadas de cada conjunto $C_i$ e, de cada grupo de soluções não-dominadas com valores idênticos em todas as funções objetivos são removidas todas as soluções, exceto uma.

Assim, todas as soluções de $C_i$ são constituídas por soluções não-dominadas que ocupam pontos distintos do espaço de objetivos. A medida $C'$ é então calculada para cada conjunto de soluções $C_i$. Para calcular $C'$ de $C_i$ é interessante definir primeiramente o conjunto complementar de $C_i$, composto por todas as soluções consideradas, exceto aquelas contidas em $C_i$, isto é:

$$\complement(C_i) = C_1 \cup C_2 \cup \ldots \cup C_{i-1} \cup C_{i+1} \cup C_{i+2} \ldots C_N$$

Então define-se $C'$:

$$C' = 1 - \frac{|\{s_x \in C_i \text{ tal que } \exists\, s_y \in \complement(C_i) \text{ que domina } s_x\}|}{|C_i|}$$

O conjunto com o valor mais alto de medida $C'$ é considerado superior aos demais em termos de eficiência de Pareto. Removendo-se as soluções do melhor – ou melhores – conjunto e repetindo-se a avaliação iterativamente, pode-se obtém-se um *ranking* que permite comparar todos os conjuntos de soluções com relação a eficiência de Pareto. O *rank* de cada conjunto de soluções pode ser então utilizado como uma métrica de dominância.

**Distância Total Média:** Dado um conjunto $C = \{s_1, s_2, \ldots, s_L\}$, considera-se cada solução $s_i \in C$ como um ponto no espaço $M$-dimensional determinado pelas funções objetivo $f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_M(\mathbf{x})$. Uma medida de distância euclidiana, $d(x,\ y)$, aplicada no espaço $M$-dimensional, é utilizada para avaliar a distância entre duas soluções quaisquer $s_x$ e $s_y$. Para encontrar essa medida de diversidade, em uma primeira etapa as soluções dominadas são descartadas e, de cada grupo de soluções com valores idênticos em todas as funções objetivos, são removidas todas as soluções, exceto uma. Assim, todas as soluções de $C = \{s_1, s_2, \ldots, s_L\}$ são soluções não-dominadas que ocupam pontos distintos do espaço de objetivos. O passo seguinte consiste em calcular os valores de distância $d(i,\ j)$ para todos os pares de soluções do conjunto $C$ restante. A razão entre a somatória de todos os valores de distância para o número de soluções contidas em $C$ é então utilizada como um estimador da diversidade do conjunto $C$.

### 6.1 Avaliação Experimental

Para avaliar os métodos de otimização por *rankings* e por eficiência de Pareto foram realizados diversos experimentos em 13 conjuntos de dados. Todos os conjuntos de dados foram selecionados do repositório *online* da Universidade da Califórnia [13].

Foram comparados sete métodos de avaliação multiobjetivo: os cinco métodos de composição de *rankings*, o método VEGA e o NSGA-II. O principal objetivo da otimização foi verificar qual método permitia encontrar fronteira de Pareto mais eficiente. Como segundo objetivo, verificou-se também a diversidade média das regras construídas com cada método de otimização. Como os métodos de composição de *rankings* não possuem mecanismo de manutenção de diversidade intrínseco, acrescentou-se a medida de distância euclidiana total como uma função-objetivo extra, a qual foi ignorada na avaliação dos resultados. O método VEGA foi avaliado duas vezes: primeiramente sem mecanismo de manutenção de diversidade e em seguida com a distância euclidiana

total como função-objetivo. Os resultados do primeiro grupo foram identificados como vega e os do segundo grupo como vegadiv.

Foram definidos cinco cenários de experimentação. Em cada cenário, um conjunto distinto de medidas de qualidade de regras foi utilizada. Os cenários e as medidas foram: (1) precisão e suporte; (2) Novidade, suporte e Laplace; (3) Piatetsky Shapiro's, suporte e precisão; (4) Piatetsky Shapiro's, suporte e Laplace; e (5) Coeficiente-$\Phi$, suporte e precisão. As equações podem ser consultadas em [1].

Os testes foram realizados com particionamento $5 \times 2$-*Cross-Fold Validation*, na qual o conjunto de dados é particionado em dois conjuntos disjuntos, cada um utilizado alternadamente para treinamento e teste. Esse particionamento é realizado cinco vezes. Os experimentos foram repetidos $10$ vezes.

Na Tabela 1 são fornecidos os valores médios e desvio padrão de desempenho para cada método considerando-se todos os conjuntos de dados e cenários de execução. Observa-se que o NSGA-II tem desempenho médio superior tanto em termos de dominância quanto em diversidade da fronteira de Pareto, obtendo *rank* médio de $2.54$ no critério de dominância e diversidade média $0.97$. O método VEGA com mecanismo de diversidade também obtém *rank* médio razoável, $3.99$, ao passo o método da composição de *rankings* pela média aritmética obtém o segundo melhor valor médio de diversidade, $0.71$.

**Tabela 1.** Síntese dos resultados experimentais.

| Métodos | Médias Desv. |
|---|---|
| condorcet | size: 2.5400 / 1.1937 |
| | dom: 5.1596 / 1.8337 |
| | div: 0.2543 / 0.0239 |
| harmonic | size: 1.7108 / 0.7521 |
| | dom: 5.8748 / 1.6416 |
| | div: 0.0908 / 0.0131 |
| mean | size: 4.1770 / 1.7646 |
| | dom: 3.8968 / 1.8711 |
| | div: 0.7102 / 0.0460 |

| Métodos | Médias Desv. |
|---|---|
| median | size: 3.8320 / 1.7130 |
| | dom: 4.2784 / 1.9181 |
| | div: 0.5714 / 0.0413 |
| nsga2 | size: 6.6595 / 2.3363 |
| | dom: 2.5428 / 1.5814 |
| | div: 0.9723 / 0.0509 |
| reciprocal | size: 2.2253 / 1.1297 |
| | dom: 5.6517 / 1.7544 |
| | div: 0.2546 / 0.0275 |

| Métodos | Médias Desv. |
|---|---|
| vega | size: 3.2030 / 1.4839 |
| | dom: 4.5986 / 1.9331 |
| | div: 0.2924 / 0.0258 |
| vegadiv | size: 4.2935 / 1.8371 |
| | dom: 3.9973 / 1.9853 |
| | div: 0.5368 / 0.0376 |

Foi aplicado o teste estatístico de Friedman[1]. Com $95\%$ de confiança o teste de Friedman rejeitou a hipótese nula de que todos os métodos são idênticos e um teste de Nemenyi foi realizado para verificar quais métodos apresentam diferenças estatísticas. Na Figura 3 é apresentado um diagrama CD sumarizando o teste de Nemenyi para dominância. A reta graduada é utilizada para indicar os *ranks* atribuídos aos métodos no teste de hipótese, portanto, quanto menor, melhor. Métodos cujos *ranks* diferem de um valor inferior à diferença crítica (CD) não apresentam diferença estatisticamente significativa. Uma linha conectando esses métodos indica a ausência de diferença estatística.

Com base no teste, pode-se inferir que o método NSGA-II apresentou superioridade estatisticamente significativa sobre quase todos os outros métodos de otimização, com exceção dos métodos VEGA associado a mecanismo de diversidade e ao método

[1] O teste de Friedman é o equivalente não-paramétrico de avaliações-repetidas do ANOVA. Verifique [14] para maiores discussões sobre testes estatísticos em AM.

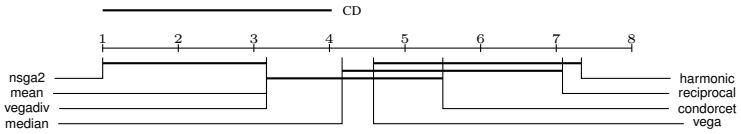


**Figura 3.** Diagrama CD para o teste de hipótese referente à dominância.

de composição de *rankings* baseado na média aritmética. Os métodos do *ranking* recíproco e de composição de *rankings* pela média harmônica apresentaram inferioridade estatisticamente significativa com relação a quatro outros métodos.

A mesma avaliação foi realizada para comparar a diversidade dos conjuntos de regras construídos pelos diferentes métodos. O diagrama CD é mostrado na Figura 4.

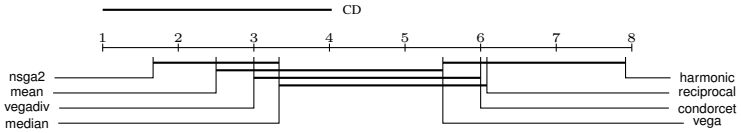


**Figura 4.** Diagrama CD para o teste de hipótese referente à diversidade.

Como pode ser observado, o método NSGA-II apresenta superioridade estatisticamente significativa com relação a quatro métodos e superioridade estatisticamente significativa a quase todos os métodos baseados em composição de *rankings*, à exceção da composição pela média aritmética e pela mediana. O método de composição pelo *ranking* médio não apresentou diferença estatisticamente significante NSGA-II com relação à diversidade. Novamente, o método de composição de *rankings* baseado na média harmônica apresentou inferioridade estatisticamente significativa com relação a quatro métodos.

## 7 Conclusão

Neste trabalho foram pesquisados dois algoritmos evolutivos multiobjetivos para a construção de regras de conhecimento com base em critérios arbitrários especificados pelo usuário. Os métodos VEGA e NSGA-II foram incorporados à biblioteca de classes ECLE. Os métodos de computação evolutiva multiobjetivo foram avaliados experimentalmente e comparados com métodos baseados em composição de *rankings* previamente existentes na ECLE. Verificou-se que o NSGA-II apresentou desempenho signficativamente superior ao da maioria dos métodos baseados em composição de *rankings* e que os métodos de diversidade empregados aumentaram a capacidade do sistema de satisfazer às medidas desejadas pelo usuário. A isso atribuímos o fato de a diversidade permitir ao sistema explorar mais eficazmente a fronteira de Pareto com uma população mais diversa.

É interessante observar a ausência de diferença estatisticamente significativa, em termos de diversidade, dos resultados produzidos do grupo vega para os do grupo vegadiv. Esperava-se que a inclusão de mecanismo de manutenção da diversidade permitisse encontrar conjuntos de soluções muito mais diversos. É possível que um outro mecanismo de manutenção de diversidade produza melhores resultados. Como recomendação de trabalho futuro, pode-se comparar os resultados apresentados com resultados obtidos por outros mecanismos de manutenção de diversidade.

**Agradecimentos:** os autores agradecem à FAPESP o apoio ao trabalho de mestrado que permitiu a execução deste projeto de pesquisa.

## Referências

1. Giusti, R.: Descoberta de Regras de Conhecimento Utilizando Computação Evolutiva Multi-Objetivo (2010) Dissertação de Mestrado, ICMC-USP.
2. Giusti, R., Batista, G.E.A.P.A., Prati, R.C.: Evaluating Ranking Composition Methods for Multi-Objective Optimization of Knowledge Rules. In: Proceedings of the 8th International Conference on Hybrid Inteliggent Systems, IEEE Computer Society (2008) 1–6
3. Giusti, R., Batista, G.: Discovering knowledge rules with multi-objective evolutionary computing. In: Machine Learning and Applications (ICMLA), 2010 Ninth International Conference on. (dec. 2010) 119 –124
4. Pila, A.D.: Computação Evolutiva para Construção de Regras de Conhecimento com Propriedades Específicas. Tese de doutorado, ICMC/USP, São Carlos, SP (2007)
5. Pila, A.D., Giusti, R., Prati, R.C., Monard, M.C.: A multi-objective evolutionary algorithm to build knowledge classification rules with specific properties. In: HIS, IEEE Computer Society (2006) Publicado em CD-ROM.
6. Batista, G.E.A.P.A., Prati, R.C., Monard, M.C., Giusti, R., Milaré, C.R.: Classificação associativa utilizando seleção e construção de regras: um estudo comparativo. In: Encontro Nacional de Inteligência Artificial (ENIA): Anais do Congresso da Sociedade Brasileira de Computação. (2007) 1321–1330
7. Alpaydin, E.: Introduction to Machine Learning. MIT Press (2004)
8. Lavrač, N., Flach, P., Zupan, B.: Rule evaluation measures: A unifying view. In Springer-Verlag, ed.: Lecture Notes in Artificial Intelligence. (1999) 174–185
9. Michalewicz, Z.: Genetic Algorithms + Data Structures = Evolution Programs. IE-Springer-Verlag (1997)
10. Mitchell, M.: An Introduction to Genetic Algorithms. The MIT Press (1998)
11. Schaffer, J.: Some experiments in machine learning using vector evaluated genetic algorithms. Tese de doutorado, Vanderbilt University, Nashville, TN, USA (1984)
12. Deb, K., Agrawal, S., Pratap, A., Meyarivan, T.: A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II. In: Parallel Problem Solving from Nature PPSN VI, Springer (2000) 849–858
13. Blake, C., Keogh, E., Merz, C.: Uci irvine repository of machine learning databases (1998) http://www.ics.uci.edu/∼mlearn/MLRepository.html.
14. Demsar, J.: Statistical comparisons of classifiers over multiple data sets. Journal of Machine Learning Research **7** (2006) 1–30