FUNDAÇÃO EDSON QUEIROZ
UNIVERSIDADE DE FORTALEZA
CENTRO DE CIÊNCIAS TECNOLÓGICAS
MESTRADO EM INFORMÁTICA APLICADA

Daniel Gonçalves de Oliveira

Estudo comparativo entre metaheurísticas populacionais com tamanho da população variável

Fortaleza
2008

FUNDAÇÃO EDSON QUEIROZ
UNIVERSIDADE DE FORTALEZA
CENTRO DE CIÊNCIAS TECNOLÓGICAS
MESTRADO EM INFORMÁTICA APLICADA

Daniel Gonçalves de Oliveira

Estudo comparativo entre metaheurísticas populacionais com tamanho da população variável

Dissertação apresentada ao programa de Mestrado em Informática Aplicada da Universidade de Fortaleza como requisito parcial para a obtenção do título de Mestre em Informática.

Orientador: Prof. Dr. André Luís Vasconcelos Coelho

Fortaleza
2008

O48d Oliveira, Daniel Gonçalves.
Estudo comparativo entre metaheurísticas populacionais com tamanho da população variável / Daniel Gonçalves Oliveira. - 2008.
95 f.

Cópia de computador.
Dissertação (mestrado) – Universidade de Fortaleza, 2008.
"Orientação : Prof. Dr. André Luís Vasconcelos Coelho."

1. Heurística (Informática). 2. Computação evolutiva. 3. Redes neurais.
4. Otimização matemática. I. Título.

CDU 681.3:004.023

Daniel Gonçalves de Oliveira

Estudo comparativo entre metaheurísticas populacionais com tamanho da população variável

Data de Aprovação __/__/____

Banca Examinadora:

_____

Prof. André Luís Vasconcelos Coelho, D. Sc.
(Orientador: Universidade de Fortaleza – UNIFOR)

_____

Prof. Aurora Trinidad Ramirez Pozo, D. Sc.
(Membro externo: Universidade Federal do Paraná – UFPR)

_____

Prof. João Batista Furlan Duarte, D. Sc.
(Membro interno: Universidade de Fortaleza – UNIFOR)

Aos meus pais,
Pedro e Graça, meus pilares

Agradecimentos

Agradeço a todos que, de forma direta ou indireta, tornaram possível a realização deste trabalho.

Ao professor André Luís Vasconcelos Coelho, pela orientação, ensinamentos e principalmente pelas críticas sempre construtivas. Por ter me apoiado e me guiado durante todo o curso engrandecendo minha formação acadêmica.

A minha namorada Lívia, pela paciência, incentivo e força. Por sempre estar ao meu lado e nunca me deixar olhar para trás.

A minha amiga Lara, pela ajuda direta na finalização deste trabalho. Sem sua ajuda meu caminho nessa trilha seria bem mais difícil. E aos amigos David, Gustavo, Juliana e Adriano, que me incentivaram até o fim e estavam sempre dispostos a compartilhar conhecimento. Que dividem comigo não só as alegrias, mas também as angústias e medos que antecederam minhas conquistas.

Ao professor Vasco Furtado e todos os amigos da Célula de Engenharia do Conhecimento, por me acolher de forma calorosa e me prover um ótimo ambiente de estudo.

A todos os meus amigos, por me encherem de alegria e pelo simples fato de estarem sempre ao meu lado.

E aos meus pais, Pedro e Graça, e meu irmão Pedro, por serem minha base forte, meu sustento, minha força, meus amores.

Resumo da dissertação apresentada ao Corpo Docente do programa de Mestrado em Informática Aplicada da Universidade de Fortaleza, como parte dos requisitos necessários para a obtenção do grau de Mestre em Informática.

Estudo comparativo entre metaheurísticas populacionais com tamanho da população variável

Autor: Daniel Gonçalves de Oliveira
Orientador: Prof. Dr. André Luís Vasconcelos Coelho

Este trabalho apresenta quatro novos algoritmos heurísticos de cunho populacional, cujo tamanho da população varia ao longo de sua execução, sendo estes destinados à resolução de problemas de busca e otimização numérica. Estes algoritmos são extensões dos modelos padrão de duas metaheurísticas propostas recentemente na literatura e que vêm sendo aplicadas com sucesso em diferentes domínios; são elas a Otimização por Enxame de Partículas (PSO) e a otimização por Evolução Diferencial (DE). Ademais, os novos algoritmos são adaptações de dois outros modelos propostos no contexto de algoritmos Genéticos (AGs): o Algoritmo Genético com Tamanho Adaptativo da População (APGA) e o Algoritmo Genético com Tamanho da População Variável Baseado na Melhoria do *Fitness* (PRoFIGA). Com o intuito de validar empiricamente os algoritmos propostos, estes são avaliados, em termos de critérios de eficiência e eficácia, em três estudos de caso: otimização de funções numéricas de *benchmark*; descoberta de protótipos em agrupamentos de dados; e treinamento de redes neurais *feedforward*. Os resultados obtidos na otimização de funções numéricas de *benchmark* indicam a possibilidade de ganhos substanciais, em termos da localização de soluções (quase-)ótimas, em relação aos modelos com tamanho fixo da população. Já na tarefa de descoberta de protótipos em agrupamentos de dados, os resultados obtidos não evidenciaram ganhos significativos em termos de eficácia. Finalmente, os resultados obtidos na tarefa de treinamento de redes neurais artificiais indicam também ganhos de eficácia, embora menos expressivos, por parte dos algoritmos com população variável em relação aos algoritmos com tamanho fixo da população.

Palavras-Chave: Otimização, Metaheurísticas Populacionais, Controle de Parâmetros, Computação Evolutiva, Inteligência Coletiva, Otimização por Enxame de Partículas, Otimização por Evolução Diferencial, Agrupamento de dados, Redes Neurais Artificiais.

Abstract of the dissertation presented to the board of faculties of the Master Program in Applied Informatics at the University of Fortaleza, as partial fulfillment of the requirements for the Master's degree in Computer Science.

Comparative study on population-based metaheuristics with time-varying population size

Author: Daniel Gonçalves de Oliveira
Advisor: Prof. Dr. André Luís Vasconcelos Coelho

This work introduces four novel population-based heuristic algorithms, whose population size varies along the execution, which are aimed at solving problems of search and numerical optimization. These algorithms are extensions of the standard models of two metaheuristics recently proposed in the literature, which have been successfully applied in different fields. They are: Particle Swarm Optimization (PSO) and Differential Evolution (DE). In addition, these new algorithms are adaptations of two other models proposed in the context of Genetic algorithms (GAs), namely, the Adaptive Population size GA (APGA) and Population Resize on Fitness Improvement GA (PRoFIGA). In order to empirically validate the proposed algorithms, their implementations are evaluated in terms of efficiency and effectiveness in three different case studies: optimization of benchmark numerical functions; prototype selection for data clustering; and training of feedforward neural networks. The results obtained in the benchmark functions optimization indicate gains, in terms of the effectiveness issue, for time-varying population size models. Conversely, the results achieved by the time-varying population size models when dealing with the data clustering task have not shown gains in performance. Finally, in the training of artificial neural networks, the novel algorithms could outperform the standard models in terms of effectiveness criterion, although the gains incurred were less expressive than those obtained in the first case study.

Keywords: Optimization, Population-based Metaheuristics, Parameter Control, Evolutionary Computing, Particle Swarm Optimization, Differential Evolution, Data Clustering, Artificial Neural Networks.

Sumário

Lista de figuras

Lista de tabelas

Introdução

Problemas de busca e otimização são freqüentes no nosso dia-a-dia. Como exemplo prático e didático, poderíamos citar o caso de um telespectador que, diante de seu aparelho de televisão, tenta melhorar a sua imagem atuando na antena e a colocando em várias posições diferentes até que encontre um sinal que lhe agrade. Assim como essa tarefa simples de calibração que enfrentamos em nosso cotidiano, problemas de otimização são pervasivos em várias áreas de estudo, como logística, engenharia, economia, ciências contábeis e computação. O presente trabalho tem como cerne de investigação a especificação de quatro novos algoritmos heurísticos com populações de tamanho variável dirigidos à resolução de problemas de busca e otimização numérica.

Vários são os métodos já existentes para a resolução de problemas de otimização contínua ou discreta (Hiller e Lieberman 2002)(Burke e Kendall 2005). *Simplex* e *Branch* & *Bound*, por exemplo, são duas das principais abordagens investigadas no contexto da Programação Matemática, as quais garantem encontrar soluções ótimas para tais problemas. Entretanto, para instâncias que possuam muitas variáveis e restrições, sua modelagem se torna difícil e o tempo de resposta desses métodos se torna impraticável. Em contrapartida, os métodos heurísticos não garantem encontrar a solução ótima para um problema qualquer, porém são capazes de localizar, muitas vezes, soluções satisfatórias em um tempo razoável. Dentre as abordagens heurísticas, destacam-se aquelas de cunho metaheurístico (Glover e Kochenberger 2003).

Uma metaheurística pode ser definida como uma estratégia em alto nível para exploração “inteligente” do espaço de busca, ou seja, do conjunto das possíveis soluções para um determinado problema de busca/otimização, lançando mão geralmente de diferentes

métodos heurísticos de baixo nível (Blum e Roli 2003). Comumente, estas estratégias são classificadas como metaheurísticas de trajetória ou metaheurísticas populacionais. Dentre as metaheurísticas de trajetória mais conhecidas, encontram-se a Têmpera Simulada (Kirkpatrick, Gelatt e Vecchi 1983) e a Busca Tabu (Glover 1986). Estes métodos trabalham apenas com uma solução por iteração. A solução inicial é geralmente gerada aleatoriamente e, ao passar das iterações, descreve uma trajetória pelo espaço de busca.

O enfoque principal deste trabalho está nas metaheurísticas populacionais. Diferentemente das metaheurísticas de trajetória, nestas, várias soluções são manipuladas concorrentemente por iteração. Uma das metaheurísticas populacionais mais conhecidas é o Algoritmo Genético (AG), elaborada por Holland (1975) e posteriormente aprimorada por Goldberg (1989), que se fundamenta nos conceitos de genética populacional e da teoria da evolução das espécies postulada por Darwin (1859). O AG apenas é um dos algoritmos que se encontram sob a classificação de Algoritmos Evolutivos (AEs) (Bäck, Fogel e Michalewicz 1997).

Duas outras metaheurísticas populacionais introduzidas mais recentemente na literatura são: otimização por Enxame de Partículas (PSO[1]) e Evolução Diferencial (DE[2]). O primeiro algoritmo foi proposto por Eberhart e Kennedy(1995) e se baseia no comportamento coletivo exibido por espécies sociais (por exemplo, revoada de pássaros, formação de cardume de peixes, etc.). Já o DE (Price, Storn e Lampinen 2005) é um algoritmo que guarda certa semelhança com os AGs, mas que emprega operadores distintos, baseados em operações algébricas (multiplicação e subtração) sobre cadeias numéricas (soluções).

Os três algoritmos heurísticos citados trabalham de maneira semelhante, aplicando operadores sobre o conjunto de soluções (população), almejando-se a convergência para um ponto de ótimo global. Logo, o desempenho (em termos de eficiência e eficácia) desses algoritmos está diretamente relacionado com a forma com que a população é manipulada. O tamanho da população passa a ser, portanto, um dos parâmetros mais críticos para esses algoritmos, visto que, se um valor pequeno for escolhido, o algoritmo pode convergir prematuramente para um ponto de ótimo local; por outro lado, se o valor escolhido for grande, a utilização de recursos computacionais tende a aumentar, prejudicando a convergência. Encontrar um valor apropriado para este parâmetro é uma tarefa bastante complicada, tendo

[1] Acrônimo do inglês *Particle Swarm Optimization*.
[2] Acrônimo do inglês *Differential Evolution*.

em vista que já foi demonstrado, para os AEs, tanto teórica quanto empiricamente, que o tamanho ideal para a população depende da complexidade do problema em questão (Koumousis e Katsaras 2006)(Smith e Smuda 1995). Alguns estudos defendem, por sinal, que o tamanho apropriado para a população de soluções deve variar ao longo do processo de busca. Com base nessas observações e motivados pelo fato de que na natureza o tamanho da população de uma espécie tende a variar continuamente e se estabilizar próximo a um valor apropriado (Fernandes e Rosa 2006), nos últimos anos, alguns pesquisadores vêm propondo novas extensões sobre os algoritmos evolutivos. Tais extensões apresentam a habilidade de, durante sua própria execução, controlar adaptativamente o valor do tamanho da população (Lobo e Lima 2005)(Eiben, Marchiori e Valkó 2004).

Os bons resultados encontrados com a utilização da noção de população de tamanho variável nos AGs e na Programação Genética (PG) (Spinosa e Pozo 2002) e a escassez de trabalhos sobre o assunto no contexto de outras metaheurísticas populacionais nos motivaram a adaptar os algoritmos PSO e DE para também operarem com populações variantes no tempo. Até o presente, o artigo de Teo (2005) parece ser o único a trazer alguma contribuição nesse sentido, já que ficou comprovado experimentalmente que a utilização de uma abordagem auto-adaptativa proposta pelo autor para a variação do tamanho da população é factível de ser explorada em DEs. Porém, em vez de criarmos um novo método de variação do tamanho da população, decidimos por adaptar dois métodos já conceituados na área dos AGs para o contexto de PSO e DE. Os dois métodos são o Algoritmo Genético com Tamanho Adaptativo da População (APGA[3]) (Bäck, Eiben e Vaart 2000) e o Algoritmo Genético com Tamanho da População Variável Baseado na Melhoria do *Fitness* (PRoFIGA[4]) (Eiben, Marchiori e Valkó 2004). Os novos algoritmos derivados foram batizados neste trabalho como PRoFIPSO e APPSO (no contexto do PSO) e PRoFIDE e APDE (no contexto do DE).

Em um primeiro momento, realizamos vários testes com funções *benchmark* de otimização numérica (Digalakis e Margaritis 2002)(Michalewicz 1996). A realização de testes utilizando esse tipo de função nos permitiu avaliar (em termos dos critérios de eficiência e eficácia) quão promissores são os novos algoritmos. Nestes testes executamos os algoritmos com o tamanho da população inicial variando entre 20, 50 e 100. Já a dimensão dos problemas variou entre 10, 20, 50 e 100. Dessa forma, pudemos mensurar o impacto que o tamanho da população inicial e a dimensão do problema exercem no desempenho dos

[3] Acrônimo do inglês *Adaptive Population size Genetic Algorithm*.
[4] Acrônimo do inglês *Population Resizing on Fitness Improvement Genetic Algorithm*.

algoritmos. Alem dos testes com funções *benchmark* de otimização numérica, aplicamos também os algoritmos em dois outros problemas computacionais, os quais podem ser modelados como problemas de otimização numérica. São eles: descoberta de protótipos para agrupamento (*clustering*) de dados (Paterlini e Krink 2006) e treinamento de redes neurais *feedforward* (Braga, Carvalho e Ludermir 2007)(Haykin 1999).

O restante do trabalho se articula da seguinte forma. No Capítulo 1, são introduzidas as metaheurísticas populacionais que serviram de alvo de investigação. No Capítulo 2, são descritos algoritmos com tamanho da população variável propostos no contexto de AGs e são introduzidas as versões adaptadas destes para o contexto de PSO e DE. No Capítulo 3, são apresentados os estudos de caso e é conduzida uma análise quantitativa e qualitativa dos resultados obtidos em nossos experimentos. O Capítulo 4 é destinado à conclusão e discussão de trabalhos futuros.

1. Metaheurísticas populacionais

Metaheurísticas compreendem métodos heurísticos genéricos, geralmente estocásticos, dirigidos à resolução de uma ampla gama de problemas de busca e otimização (Blum e Roli 2003)(Glover e Kochenberger 2003). Neste capítulo, apresentamos os conceitos básicos relacionados às metaheurísticas populacionais que serviram de alvo de investigação neste trabalho. Primeiramente, é descrita a lógica por trás dos algoritmos evolutivos, dando ênfase aos algoritmos genéticos. Em seguida, são explicados os algoritmos padrão do PSO e do DE. Finalmente, é feita uma breve discussão, salientando o papel importante do componente "população" nestes algoritmos.

1.1. Computação evolutiva

A Computação Evolutiva (CE) é uma área da computação embasada nos processos e princípios da evolução biológica (Bäck, Fogel e Michalewicz 1997). Não é surpresa que cientistas busquem inspiração no processo evolutivo, visto que o seu poder está muito claro ao se observar as diversas espécies de animais que competem e se adaptam ao ambiente para sobreviver. De forma bem direta, a evolução pode ser encarada como um processo de otimização, em que o objetivo é o de aumentar as chances de sobrevivência e reprodução dos organismos que habitam um ambiente competitivo e dinâmico. Essa é a metáfora-chave que está por trás do desenvolvimento dos algoritmos evolutivos.

A teoria evolucionária de Jean-Baptiste Lamarck (1744-1829) discorre sobre o conceito da hereditariedade, afirmando que o indivíduo se adapta durante seu tempo de vida, e

suas características são então repassadas para sua prole, que, sucessivamente, vai continuar se adaptando e transmitindo suas características aos seus descendentes. Segundo Lamarck, a adaptação dos indivíduos ocorre pela “Lei do uso e desuso”, segundo a qual o indivíduo vai perdendo características que ele não usa ao longo do tempo e adquirindo outras que lhe são úteis.

A teoria da evolução de Darwin (1859) defende que, em um ambiente com recursos limitados e populações estáveis, cada indivíduo compete com outros para sobreviver. Aqueles indivíduos com melhores características estarão mais aptos a sobreviver e se reproduzir, e essas características serão propagadas a sua prole. Essas características desejáveis são herdadas pelas gerações subseqüentes e, com o passar do tempo, acabam se tornando dominantes entre os indivíduos da população.

A segunda parte da teoria de Darwin afirma que, durante a geração da prole, eventos aleatórios podem causar mudanças em suas características. Se essas novas características forem favoráveis para o indivíduo, suas chances de sobrevivência ficam ainda maiores. Darwin, porém, não conhecia quais mecanismos tornavam possível essa herança de características entre gerações. Gregor Mendel descobriu que este processo de transmissão de características estava associado a uma unidade básica de informação, o gene (Linden 2006).

1.1.1. Algoritmos evolutivos e algoritmos genéticos

Através de operadores inspirados em processos e fenômenos naturais (como seleção, mutação e reprodução), um algoritmo evolutivo manipula iterativamente populações de estruturas denominadas indivíduos ou cromossomos, simulando a evolução de uma espécie. Durante esse processo, os indivíduos são avaliados, aqueles menos aptos serão punidos e os mais aptos (adaptados ao ambiente) são de algum modo privilegiados. Existem vários modelos computacionais implementados de algoritmos evolutivos (Bäck 1996)(Bäck, Fogel e Michalewicz 2000a)(Bäck, Fogel e Michalewicz 2000b); porém, para este trabalho, maior ênfase foi dada aos Algoritmos Genéticos (AGs). O pseudo-código de um AG pode ser visto na Figura 1.

Relacionando o pseudo-código na Figura 1 com as leis de Darwin, percebe-se que a primeira lei sobre Seleção Natural acontece no operador de seleção para reprodução, em que os melhores indivíduos possuem maiores chances de serem escolhidos para se reproduzir. A reprodução ocorre mediante o uso do operador de *crossover* com a recombinação do material genético dos indivíduos escolhidos no operador de seleção para reprodução. Já a segunda parte das leis de Darwin, que relata a existência de mudanças aleatórias, é modelada pelo operador de mutação.

```
01 t = 0, sendo t o contador de gerações;
02 criar e iniciar a população com N indivíduos P(0);
03 repetir
04     avaliar cada indivíduo, xi, na população P(t);
05     aplicar a seleção para reprodução em P(t)
06     aplicar o crossover para gerar novos indivíduos;
07     aplicar a mutação nos novos indivíduos;
08     selecionar indivíduos para a próxima geração P(t+1);
09     incrementar a geração, t = t + 1;
10 até critério de parada for verdadeiro
```

Figura 1 – Pseudo-código do modelo padrão de algoritmo genético.

Uma diferença que notamos entre a Computação Evolutiva e a evolução biológica é a existência do critério de parada. Existem várias formas de se determinar o fim do processo de otimização. Por exemplo, especificar uma quantidade fixa de gerações, ou o valor de aptidão (*fitness*) da solução ótima. Outra seria a de não melhorar o valor de aptidão do melhor indivíduo durante um certo número de gerações consecutivas.

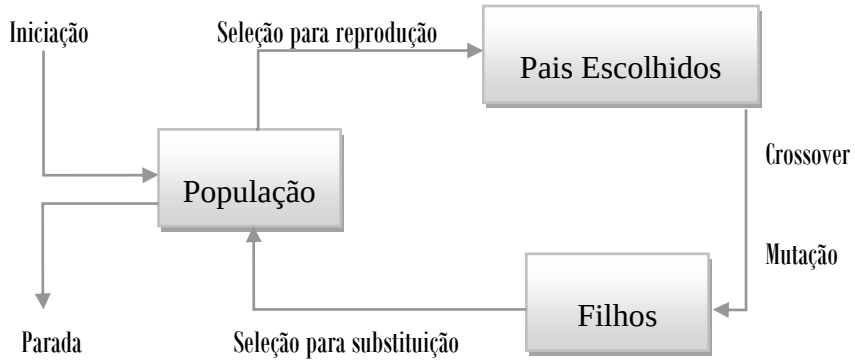

Figura 2 - Esquema básico de um algoritmo evolutivo (Eiben e Smith 2003).

Na Figura 2, temos um esquema conceitual do algoritmo mostrado na Figura 1. Uma população inicial é gerada e avaliada, após o que começa o laço principal do algoritmo. Alguns indivíduos da população são escolhidos, para, na fase de recombinação (via

operador de *crossover*), serem gerados novos indivíduos. Nessa seleção, os indivíduos que possuem maior grau de aptidão são mais susceptíveis a serem escolhidos. Isso não quer dizer que os demais não possam ser escolhidos também, já que se trata geralmente de um operador estocástico. Com os novos indivíduos gerados, eles passam por um processo de mutação e são então avaliados. Esse processo de mutação é estocástico e também existem várias estratégias para realizá-lo. De posse dos filhos, temos que selecionar quais indivíduos farão parte da população na próxima geração, dentre pais e filhos. Feita a seleção para substituição, o algoritmo reinicia até que a condição de parada seja atingida e o algoritmo termine. Embora esses operadores sejam estocásticos, sua aplicação é direcionada por meio de parâmetros de controle previamente definidos. Portanto, além do tamanho da população, a taxa de seleção, a taxa de reprodução e a taxa de mutação são parâmetros de controle a serem (previamente) definidos. Definir os valores para estes parâmetros não é uma tarefa de fácil resolução, pois seus valores ótimos variam dependendo das características e natureza do problema (Eiben e Smith 2003).

Um dos pontos críticos dos AEs está na possibilidade de ocorrer convergência prematura da população (Bäck, Fogel e Michalewicz 1997). Este é um problema que faz com que o algoritmo convirja rapidamente para um ponto de mínimo/máximo local, não conseguindo mais se desvencilhar da base de atração desse ponto. Uma das possíveis causas para este fenômeno seria a existência de alta pressão seletiva na população, principalmente nas primeiras gerações do processo evolutivo, levando à falta de diversidade da população.

1.2. Enxame de partículas

Otimização por Enxame de Partículas (PSO) é uma metaheurística que faz parte da área de pesquisa conhecida como Inteligência Coletiva (Kennedy e Eberhart 2001). Seu funcionamento é baseado na simulação do comportamento coletivo exibido por algumas espécies de animais sociais (Kennedy e Eberhart 1995), tais como aquele observado em um bando de pássaros ou em um cardume de peixes. As regras que regem a coreografia de um bando de pássaros voando sincronamente, ou um cardume de peixes, trocando de direção repentinamente, se espalhando e reagrupando, indicam que existe um esforço para manter uma distância ótima entre cada indivíduo e seus vizinhos. Estudos mostram que pelo menos

alguns indivíduos de um cardume conseguem se beneficiar de descobertas e experiências passadas vividas no cardume durante a procura por comida. Esse mecanismo de troca de informações sugere uma grande vantagem para a evolução do coletivo. Essa hipótese é a idéia-base por trás do PSO (Kennedy e Eberhart 1995).

Cada solução no PSO é chamada de partícula e esta se movimenta pelo espaço de busca com uma aceleração que é alterada dinamicamente, de acordo com a sua experiência e as experiências das outras partículas. Não existe operador/processo de seleção no PSO; assim, as mesmas partículas permanecem do início ao fim da execução do algoritmo. Por ser um método simples, de fácil implementação e baixo custo computacional (Eberhart, Simpson e Dobbins 1996), tem sido foco de vários estudos recentes.

Existem duas versões de implementação do PSO (Kennedy e Eberhart 2001): a versão local, que utiliza o conceito de vizinhança como sendo um conjunto de indivíduos próximos (logicamente ou fisicamente); e a versão global, em que a vizinhança engloba toda a população. Neste trabalho, optamos por utilizar a versão global do PSO. Nesta versão, cada partícula representa um vetor-solução de dimensão d, $X_i = (x_{i1}, x_{i2}, \dots, x_{id})$, possuindo, além do seu valor de *fitness*, um vetor de velocidade, $V_i = (v_{i1}, v_{i2}, \dots, v_{id})$, e um vetor com a melhor posição já visitada pela partícula, $P_i = (p_{i1}, p_{i2}, \dots, p_{id})$. Logo, a movimentação da partícula acontece guiada por sua velocidade, o melhor ponto que já visitou e a posição em que o melhor indivíduo da sua vizinhança já esteve, esta denotada por $P_g = (p_{g1}, p_{g2}, .., p_{gd})$. Na Figura 3 é dado o pseudo-código do algoritmo PSO.

```
01 criar aleatoriamente as partículas no espaço de busca.
02 avaliar as partículas
03 g=0;
04 enquanto critério de parada não for satisfeito faça
05     para cada partícula atualizar o vetor de velocidade.
06     para cada partícula atualizar a posição.
07     para cada partícula i faça
08            se pb(i)>pb(i-1) então atualizar pb(i);
09     se pg(g)>pg(g-1) então atualizar pg(g);
10 fim
```

Figura 3 - Pseudo-código para o PSO.

Assim como em AEs, o valor de aptidão (*fitness*) f_i indica quão adequada é a solução representada pela partícula *i* para solucionar o problema em vista; ou seja, para um problema de maximização, quão maior for este valor, melhor será a solução representada pelo vetor x_i.

Na geração t, o vetor de velocidade da próxima geração é recalculado para cada partícula de acordo com a seguinte equação:

$$v_i(t+1) = wv_i(t) + c_1\emptyset_1\big(p_i(t) - x_i(t)\big) + c_2\emptyset_2\left(p_g(t) - x_i(t)\right), \quad (1)$$

em que w é o peso de inércia, c_1, c_2 são conhecidos como parâmetro cognitivo e parâmetro social, respectivamente, e $\emptyset_1$ e $\emptyset_2$ são variáveis aleatórias geradas no intervalo entre [0,1].

O valor de w é crucial para a convergência do PSO. Sua função é controlar o impacto do valor da velocidade em gerações anteriores sobre o seu valor na geração atual. Valores altos para w facilitarão a exploração de uma maior área do espaço de busca, mas podem comprometer o tempo para convergência e a utilização de recursos computacionais; por outro lado, valores pequenos para w facilitarão o refinamento da solução em regiões de ótimos locais, mas podem implicar em uma convergência prematura (Shi e Eberhart 1998a) (Shi e Eberhart 1998b). Estudos realizados por Yoshida et al. (1999) indicam bons resultados quando o valor de w decresce linearmente durante a execução no intervalo começando com um valor próximo de um e decrescendo até próximo de zero. Com um método mais elaborado, o artigo (Shi e Eberhat 2001) sugeriu um mecanismo "*fuzzy"* para encontrar um valor ideal para o parâmetro w; os resultados mostraram um ganho significante no desempenho do PSO. Já os estudos realizados em (Liping, Huanjun e Shangxu 2003) indicam que a estratégia de decrescer linearmente o valor de w pode levar o algoritmo a ficar "preso" em ótimos locais. A proposta do trabalho supracitado foi então alterar o valor de w aleatoriamente no intervalo entre zero e um. Os resultados dessa proposta indicam ganhos ainda melhores do que os da estratégia anteriormente citada.

Os valores para c_1 e c_2 não são tão cruciais para a convergência do algoritmo, mas contribuem para uma rápida convergência, para a "fuga" de ótimos locais e para o refinamento do resultado final. Um estudo extenso para os parâmetros de aceleração foi realizado por Kennedy (1998). Este afirma que, com os valores $c_1 = c_2 = 0{,}5$, pode-se obter melhores resultados que com os valores $c_1 = c_2 = 2$, comumente tomados como padrão na literatura. Estudos recentes (Carlisle e Dozier 2001) indicam que escolher valores maiores para o parâmetro cognitivo, c_1, do que para o parâmetro social, c_2, pode trazer resultados interessantes, desde que $c_1 + c_2 \leq 4$. Os parâmetros $\emptyset_1$ e $\emptyset_2$ são usados para manter a diversidade na população.

Tendo a velocidade calculada, a partícula *i* é acelerada para sua próxima posição seguindo esta simples equação:

$$x_i(t+1) = x_i(t) + v_i(t+1) \tag{2}$$

Na equação do cálculo da velocidade, não existe nenhum mecanismo que limite seu valor; por isso, muitas vezes, é utilizado um parâmetro V_{max} para esse fim. Dessa forma, a partícula não é lançada para fora do espaço de busca. O valor de V_{max} deve ser muito bem escolhido, pois valores muito grandes podem fazer com que as partículas fiquem fora do espaço de busca, trazendo resultados ruins, ao passo que valores pequenos poderão resultar em exploração insuficiente do espaço de busca.

Uma variação do PSO elaborada em (Clerc e Kennedy 2002) inseriu na Equação (1) um parâmetro conhecido como coeficiente de constrição. Esse parâmetro tem a função de fazer com que as partículas recebam valores menores para os seus vetores de velocidade a cada geração. A nova equação pode ser observada em (3). Um estudo comparativo entre o PSO utilizando peso de inércia e utilizando fator de constrição, realizado em (Eberhart e Shi 2000), acena que o PSO utilizando fator de constrição obteve melhores resultados que o PSO com peso de inércia quando para este último o parâmetro V_{max} assume valor dentro do mesmo domínio dos valores das variáveis da partícula.

$$v_i(t+1) = X\left(wv_i(t) + c_1\emptyset_1\big(p_i(t) - x_i(t)\big) + c_2\emptyset_2\left(p_g(t) - x_i(t)\right)\right) \tag{3}$$

$$X = \frac{2}{\left|2-\varphi-\sqrt{\varphi^2-4\varphi}\right|} \tag{4}$$

É importante mencionar que, na Equação (3), quando o coeficiente de constrição é utilizado, não se deve usar o peso de inércia, e vice-versa. Os outros parâmetros são os mesmos utilizados na Equação (1). Na Equação (4), temos o cálculo do fator de constrição, sendo que $\varphi = c_1 + c_2, \varphi > 4$.

Estudos anteriores (Shi e Eberhart 2000) indicam que o PSO não é sensível à calibração do tamanho da população, tomando como base o formato das curvas de convergência deste algoritmo para diferentes valores deste parâmetro. Carlisle e Dozier (2001) utilizaram o PSO com coeficiente de constrição proposto em (Clerc e Kennedy 2002) para testar cinco funções *benchmark* já conhecidas na literatura (Digalakis e Margaritis 2002),

variando o tamanho da população de 5 a 200. Os autores concluíram que 30 aparenta ser um tamanho adequado para qualquer tipo de problema. Porém, essa conclusão foi obtida levando-se em consideração somente valores estáticos para o tamanho da população, além de funções com dimensões pequenas e médias (≤ 30).

Outro estudo mais recente (Huang e Mohan 2006) mostra que para problemas de elevadas dimensões (≥ 100) o PSO tradicional começa a sofrer perda de desempenho. Para resolver esse problema, os autores propõem o algoritmo μPSO, que utiliza apenas três a cinco partículas por execução e, conseqüentemente, necessita de menos avaliações para encontrar um resultado ótimo. Os resultados experimentais apresentados indicam ganho em eficiência e eficácia por parte do novo algoritmo em comparação ao PSO canônico, sendo este último avaliado com tamanhos de população maiores.

1.3. Evolução diferencial

Em outubro de 1994, Kenneth Price criou o algoritmo populacional chamado Têmpera Genética, que era uma variação do algoritmo de trajetória Têmpera Simulada. Após a publicação do artigo sobre esse algoritmo na revista Dr. Dobb's, Rainer Storn procurou por Price para discutir a resolução do problema polinomial de Chebyshev pelo Têmpera Genética. Na literatura, a resolução desse problema é considerada uma tarefa muito difícil para a maioria dos métodos de otimização. Price avaliou e encontrou uma forma para resolver o problema com cinco dimensões, mas a convergência era bastante lenta e os parâmetros de configuração do algoritmo difíceis de serem determinados. Após essa primeira conquista, Price modificou o algoritmo para trabalhar com ponto flutuante no lugar de bits e para utilizar operadores algébricos no lugar de operadores lógicos. Foi aí que o pesquisador descobriu a mutação diferencial na qual a metaheurística Evolução Diferencial (DE) é toda baseada (Price, Storn e Lampinen 2005). Diferentemente do Têmpera Genética, o algoritmo DE foi capaz de resolver o problema polinomial de Chebyshev com 33 dimensões sem apresentar perda de desempenho. Atualmente, esse algoritmo se mostra capaz de resolver vários outros problemas de otimização complicados, apresentando vantagens sobre outros algoritmos convencionais.

O funcionamento do DE inicia-se com a criação da população inicial a partir de valores escolhidos aleatoriamente dentro do domínio do problema, população esta formada por vetores numéricos. A principal idéia por trás do DE é a criação de novos vetores mutantes a cada geração *G*, como observamos na Equação (5). Para cara vetor da população (chamado vetor-referência), três outros vetores diferentes entre si ($x_{r1,G}$, $x_{r2,G}$ e $x_{r3,G}$) são escolhidos aleatoriamente na população. Os vetores $x_{r2,G}$ e $x_{r3,G}$ são subtraídos entre si. O resultado desta subtração é multiplicado por um peso e somado ao vetor $x_{r1,G}$, resultando em um novo vetor. Este novo vetor é então mesclado ao vetor-referência, e, se o vetor resultante for melhor que o vetor-referência, então ele assume seu lugar; caso contrário, o novo vetor é descartado. Essa discussão é retratada na Figura 4. Na Figura 5, temos o pseudo-código do DE.

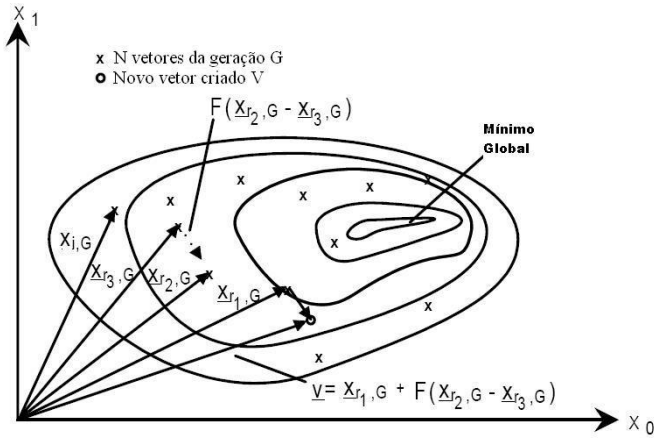

Figura 4 - Funcionamento do algoritmo DE – versão DE/rand/1/exp (Storn e Price 1995).

De um modo mais formal, seu funcionamento[5] é como segue. Para cada vetor $x_{i,G}, i = 0,1,..,N$, da população na geração *G*, um vetor v é criado de acordo com a equação:

$$v = x_{r1,G} + F(x_{r2,G} - x_{r3,G}), \tag{5}$$

onde $r1, r2, r3 \in [0, N]$, sendo estes inteiros e mutuamente diferentes, e $F > 0$. Os índices $r1, r2$ e $r3$ são escolhidos aleatoriamente e devem diferir do índice do vetor-referência. F é uma constante real, que controla a amplitude da variação diferencial $(x_{r2,G} - x_{r3,G})$.

[5] Estamos assumindo a versão padrão do DE conhecida como DE/rand/1/exp (Storn e Price 1995).

```
01 criar aleatoriamente os indivíduos no espaço de busca
02 avaliar indivíduos
03 enquanto não encontrou critério de parada
04     Para cada vetor da população P faça
05           r1 = rand(NP);
06           r2 = rand(NP);
07           r3 = rand(NP);
08           ui = xr3 + F*(xr1 - xr2);
09           se (f(ui) <= f(xi))então
10                 pAux = ui;
11           senão
12                 pAux = xi;
13     fim para;
14 fim enquanto
```

Figura 5 - Pseudo-código do algoritmo DE.

Com o intuito de manter a diversidade entre os vetores, o vetor v não é simplesmente copiado para o novo vetor. O novo vetor será formado por valores selecionados do vetor-referência *i* e por valores do vetor v. Podemos comparar este operador ao *crossover* do AG, já que elementos de dois vetores são escolhidos para formar um novo vetor (vetor "filho"). A forma com que este *crossover* é realizado está exposta na Figura 6.

```
L=0;
faça{
    L=L+1;
}enquanto((rand < CR) e (L < D))
```

Figura 6 - Algoritmo do *crossover* do DE.

De acordo com o algoritmo acima, um ponto L de corte é selecionado, e desse ponto em diante os valores do vetor escolhido vão sendo trocados pelos valores do vetor v até que o valor de *rand* seja menor que o valor de CR. Esse parâmetro é escolhido pelo usuário e pertence ao intervalo $[0,1]$. Depois que o novo vetor é criado, ele é avaliado e, se seu valor de *fitness* for melhor que o do vetor-referência, então ele entrará em seu lugar; do contrário, como já mencionado, o novo vetor é descartado, continuando o vetor-referência na população. A criação do vetor v e a aplicação do *crossover* podem ser realizadas de outras maneiras. Para maiores detalhes, pode-se consultar (Price, Storn e Lampinen 2005)(Storn e Price 1995).

Recentemente, foi proposta uma variação do algoritmo DE, a qual possui tamanho da população variável (Teo 2005). O algoritmo proposto se chama Evolução Diferencial com População Auto-adaptativa (DESAP) e possui duas versões: uma absoluta e outra relativa. O fato de ser um modelo auto-adaptativo significa que um parâmetro que controla o tamanho da

população faz parte de cada indivíduo. Ao final da avaliação de cada indivíduo, é aplicado um somatório (específico para cada versão do DESAP) que vai calcular o tamanho da população na próxima geração. O algoritmo utiliza elitismo para garantir que o melhor indivíduo não seja removido de uma geração para outra. Os resultados computacionais apresentados por Teo (2005) indicam que o algoritmo DESAP pode obter ganhos em comparação ao DE original, embora não substanciais. Optamos por não utilizar o mecanismo de variação do tamanho da população do DESAP em nossos algoritmos pois sua característica auto-adaptativa o torna difícil de estender ao modelo PSO.

1.4. O componente conceitual "população"

Para este trabalho, o componente conceitual mais importante por trás dos algoritmos supracitados é a população de soluções. No contextos dos AEs, podemos afirmar que a população como um todo é a unidade de evolução, ao passo que cada indivíduo é uma unidade de avaliação (Eiben e Smith 2003). Após definida a representação dos indivíduos de uma população, sua criação pode ser simples (por exemplo, gerando-se uma quantidade de indivíduos aleatórios) ou mais elaborada (por exemplo, mediante a definição de estruturas espaciais e de vizinhança entre os seus indivíduos) (Eiben e Smith 2003).

Apesar dessas alternativas, uma pergunta importante que deve ser sempre respondida é: qual deve ser o tamanho ideal da população para que a metaheurística em uso provenha soluções satisfatórias para o problema em vista? Essa questão retrata a importância desse parâmetro de controle, já que sua escolha afeta diretamente o desempenho e a eficiência de qualquer metaheurística populacional. No caso de uma população pequena, a cobertura do espaço de busca vai ser pequena, ocasionando problemas de convergência prematura para soluções locais. Ou seja, populações pequenas implicam em baixa eficácia e alta convergência. Em contrapartida, se o tamanho da população for elevado, temos baixa eficiência e baixa convergência. Isso porque será necessário um grande esforço computacional para se obter um resultado satisfatório.

O dilema acima retratado é colocado no contexto dos AGs como "diversidade genética versus convergência prematura", uma vez que estes conceitos são conflitantes. Atender a essa relação de compromisso é uma tarefa de suma importância em geral para qualquer algoritmo

heurístico populacional. A diversidade genética é uma medida que indica a quantidade de soluções diferentes na população (Eiben e Smith 2003), sendo que o cálculo do nível de diversidade de uma população pode ser obtido de diferentes modos. Avaliar a quantidade de fenótipos ou genótipos diferentes é um dos modos mais comumente empregados.

Por outro lado, o tamanho da população e seu grau de diversidade afetam também o dilema entre *exploration* e *exploitation* (exploração e refinamento). A idéia é que as metaheurísticas populacionais comecem realizando uma exploração do espaço de busca e, com o passar das gerações, passem a refinar as melhores soluções já obtidas. Encontrar um tamanho ideal para a população que também atenda a esse dilema de forma otimizada em todas as circunstâncias vem se apresentando como uma tarefa quase infactível.

Contudo, existe a alternativa visando à escolha adaptativa desse parâmetro de controle mediante o uso da noção de população com tamanho variável. Na natureza, as populações das diversas espécies não possuem um tamanho fixo ao logo do tempo; a dimensão da população varia de acordo com uma taxa de crescimento e tem tendência a se estabilizar próximo a um valor de equilíbrio. Mudanças no habitat (provocados, por exemplo, por fenômenos da natureza como terremotos e tsunâmis) provocam oscilações e reequilíbrio populacional.

Lançando-se mão dessa metáfora, surgiram mais recentemente na literatura extensões dos algoritmos evolutivos explorando a possibilidade da adaptação do número de indivíduos ao longo do processo evolutivo. No próximo capítulo, apresentaremos alguns desses modelos de AGs com tamanho de população variável, bem como a adaptação destes tanto para o contexto do algoritmo PSO como para o do algoritmo DE.

1.5. Conclusão

Neste capítulo, foram apresentados conceitos relativos a três metaheurísticas populacionais. Enquanto AGs utilizam operadores baseados na natureza para simular a evolução das espécies, o PSO é baseado na simulação do comportamento coletivo exibido por algumas espécies de animais sociais e o DE é baseado na evolução diferencial de soluções representadas por vetores numéricos. No próximo capítulo, versões com tamanho variável da

população desses algoritmos são apresentadas, incluindo aquelas especialmente adaptadas para o PSO e o DE.

2. Algoritmos com população de tamanho variável

Como observado no capítulo anterior, nas metaheurísticas populacionais, o tamanho da população é um parâmetro de controle de extrema importância, que pode influenciar diretamente na qualidade do resultado final encontrado. Segundo Goldberg (1989), se o tamanho da população em AGs for pequeno, a diversidade genética tenderá a ser pequena, ocasionando uma baixa cobertura do espaço de busca. Como resultado, tem-se o fenômeno da convergência prematura. Por outro lado, se o tamanho da população for elevado, o esforço para localizar a solução ótima bem como o desperdício computacional serão maiores, elevando o tempo de execução até a convergência.

No contexto dos AEs, vários estudos experimentais e teóricos já foram realizados almejando-se encontrar um tamanho ideal para a população (Smith e Smuda 1995). Atualmente, já há quase um consenso de que não existe um valor ideal para o tamanho da população que traga sempre bons resultados. Em (DeJong 1975), foi estudado pela primeira vez, dentre outros aspectos dos AGs, qual seria o tamanho ótimo das populações para um conjunto representativo de problemas de otimização. Grefenstette (1986), por outro lado, fez uso de um AG de nível meta para controlar os parâmetros de outro AG; ou seja, o problema de otimização do meta-AG era encontrar, dentre outros fatores, um valor adequado para o tamanho da população do AG principal.

Diante dessa importância do tamanho da população, surgiram no contexto dos AGs novos modelos com tamanho de população variável. O algoritmo genético com variação aleatória do tamanho da população (RVPS[6]) (Costa, Tavares e Rosa 1999) e o algoritmo

[6] Acrônimo do inglês *Random Variation of the Population Size*.

genético sem parâmetros (PLGA[7]) (Harik e Lobo 1999) são dois exemplos desses modelos, além de outros descritos em (Lobo e Lima 2005). O potencial ganho no uso do conceito de população com tamanho variável (por exemplo, possibilitar ao próprio algoritmo encontrar o valor mais adequado deste parâmetro no curso de sua execução) nos motivou a adaptar alguns desses modelos para o contexto de outras metaheurísticas populacionais, particularmente para o PSO e DE.

A seguir, apresentamos os modelos estendidos de AGs com tamanho variável da população que foram utilizados como referencial para nosso estudo. Os modelos APGA e PRoFIGA foram particularmente escolhidos por terem sido os que apresentaram melhor desempenho no estudo comparativo conduzido por Eiben, Marchiori e Valkó (2004). Além disso, esses algoritmos representam duas classes distintas de abordagens de abordagens destinadas a controle de parâmetros em AEs (Eiben e Smith 2003): determinística (PRoFIGA) e adaptativa (APGA). Posteriormente, mostraremos como foi feita a adaptação desses dois algoritmos para o âmbito do PSO e do DE.

2.1. GAVaPS

O Algoritmo Genético com Tamanho da População Variável (GAVaPS) (Arabas, Michalewicz e Mulawka 1994)(Michalewicz 1996) parece ter sido uma das primeiras abordagens evolutivas com tamanho da população variável publicadas na literatura — uma outra foi aquela proposta em (Smith e Smuda 1995). Nessa abordagem, foi introduzida a idéia de "idade do indivíduo", que representa a quantidade de gerações durante a qual um indivíduo da população permanecerá "vivo".

De acordo com o GAVaPS, na geração inicial, cada indivíduo é avaliado e recebe um tempo de vida baseado em sua aptidão (*fitness*). Quanto melhor o valor de aptidão do indivíduo, maior será seu tempo de vida. Assim, a pressão seletiva é alcançada indiretamente já que indivíduos com melhor valor de aptidão permanecerão mais tempo na população e terão maiores chances de serem escolhidos para reprodução. A "idade dos indivíduos" é

[7] Acrônimo do inglês *Parameter-less Genetic Algorithm*.

incrementada a cada geração, e quando esta atinge o tempo de vida, o indivíduo correspondente é retirado da população.

O tempo de vida pode ser calculado utilizando três abordagens diferentes (Michalewicz 1996): proporcional, linear e bi-linear. A abordagem proporcional segue idéia similar à da seleção pela roleta (Rezende 2005): o valor do tempo de vida de um indivíduo é proporcional à qualidade de sua aptidão em relação aos demais indivíduos da geração em que foi criado. Na abordagem linear, o tempo de vida é calculado utilizando-se o melhor e o pior valor de aptidão encontrados até o momento, além do valor de aptidão do próprio indivíduo. O problema dessa abordagem é que, se muitos indivíduos estiverem com valores de aptidão elevados, um tempo de vida alto será atribuído a eles, fazendo com que estes fiquem vivos por muito tempo. Isso pode levar à perda de eficiência computacional do processo evolutivo. Por fim, a abordagem bi-linear se posiciona entre as duas mencionadas anteriormente, visto que ela utiliza tanto a média dos valores de *fitness* como o maior e menor valores de *fitness* encontrados até o momento para calcular o tempo de vida dos novos indivíduos. Seguem abaixo as equações relativas a cada uma das abordagens descritas acima (Michalewicz 1996):

- Proporcional: min $(MinLt + n\frac{fitness[i]}{AvgFit}, MaxLt)$
- Linear: $MinLt + 2n\frac{fitness[i]-AbsFitMin}{AbsFitMax-AbsFitMin}$
- Bi-Linear: $\begin{cases} MinLt + n\frac{fitness[i]-MinFit}{AvgFit-MinFit} \; se \; AvgFit \geq fitness[i] \\ \frac{1}{2}(MinLt + MaxLt) + n\frac{fitness[i]-AvgFit}{MaxFit-AvgFit} \; se \; AvgFit < fitness[i] \end{cases}$

Nessas equações, *MaxLt* e *MinLt* denotam o tempo de vida máximo e mínimo, respectivamente (parâmetros estes definidos pelo usuário), e $n = \frac{1}{2}(MaxLt - MinLt)$. Já *AvgFit*, *MaxFit* e *MinFit* representam a média, o maior e o menor valor de *fitness* da população na geração atual, respectivamente. Por fim, *AbsFitMax* e *AbsFitMin* denotam o maior e o menor valor de *fitness* encontrados desde o início até a geração atual.

Na fase de reprodução, uma população auxiliar é criada com tamanho $p \times PopSize$, sendo *p* a taxa de reprodução (definida previamente) e *PopSize* o tamanho da população atual. Essa população auxiliar é também chamada de *offspring*.

Para o GAVaPS, não existe o operador de seleção para reprodução como no AG. Os indivíduos são selecionados aleatoriamente, não importando seu valor de *fitness*, e

posteriormente são aplicados sobre esses indivíduos os operadores de *crossover* e mutação. A prole resultante dessas operações vai sendo inserida na população auxiliar até que esta esteja completa. Na Figura 7, encontra-se o pseudo-código do algoritmo GAVaPS.

```
01 g=0;  //geração
02 gerar população inicial P(g);
03 para cada indivíduo i da população P(g) faça
04     avaliar aptidão do indivíduo i;
05 fim para
06 enquanto critério de parada não for satisfeito faça
07     g=g+1;
08     incrementar a idade de cada indivíduo em 1;
09     recombinar P(g);
10     avaliar P(g);
11     remover os indivíduos com idade maior que o tempo de vida
12 fim enquanto
```

Figura 7 – Pseudo-código do algoritmo GAVaPS.

O tempo de vida atribuído a um indivíduo continua o mesmo durante toda sua existência. Depois que os indivíduos da população auxiliar são avaliados, essa população é inserida na população principal e, antes de terminar a geração, os indivíduos com idade maior ou igual ao seu tempo de vida são removidos da população. Assim, o tamanho da população no início da próxima geração será: $PopSize(g+1) = PopSize(g) + AuxPopSize(g) - D(g)$, sendo que $D(g)$ é a quantidade de indivíduos que "morreram" durante a geração g.

Algumas variações sobre esse algoritmo foram criadas. Por exemplo, no GAVaPS sem incesto, niGAVaPS (Fernandes, Tavares e Rosa 2000), cada indivíduo mantém uma tabela que identifica seus ancestrais. Desta forma, é possível prevenir o acasalamento entre um indivíduo e ele próprio, bem como entre ele e seus ancestrais. Além disso, no niGAVaPS, é possível se configurar o nível de parentesco permitido entre os indivíduos selecionados para o *crossover*.

No presente trabalho, utilizamos outra variação do GAVaPS, descrita a seguir.

2.2. APGA

O Algoritmo Genético com Tamanho da População Adaptativo (APGA) (Bäck, Eiben e Vaart 2000) é uma variação do algoritmo GAVaPS proposto para resolver alguns problemas por ele apresentado. Segundo testes realizados por Eiben (2004), o algoritmo GAVaPS se mostrou bastante sensível à escolha da taxa de reprodução. Freqüentemente, os autores observaram que o tamanho da população cresce de forma desproporcional, principalmente para problemas de difícil resolução. Após experimentos preliminares, constatamos que esse fato decerto ocorre com o GAVaPS, o que nos levou a trabalhar com o APGA.

A principal diferença entre APGA e GAVaPS é que, para o primeiro, a idade do melhor indivíduo não é incrementada com o passar das gerações, ou seja, o melhor indivíduo nunca é retirado da população. Esse artifício é conhecido como "elitismo", que proporciona a possibilidade de crescimento contínuo do melhor valor de *fitness* com o decorrer das gerações. O elitismo tende a melhorar o refinamento da melhor solução sem prejudicar a exploração do espaço de busca (Linden 2006).

Além disso, a abordagem utilizada para o cálculo do tempo de vida é sempre a bi-linear e o APGA é de natureza *steady-state*, ou seja, a população não é toda recriada a cada geração; pelo contrário, apenas dois novos indivíduos resultam do operador de *crossover* e são inseridos na população a cada geração. Esses novos indivíduos também sofrem mutação.

2.3. PRoFIGA

O Algoritmo Genético com Tamanho da População Variável Baseado na Melhoria do *Fitness* (PRoFIGA) (Eiben, Marchiori e Valkó 2004) é um algoritmo em que, diferentemente do GAVaPS, as regras para crescimento ou diminuição do tamanho da população são determinísticas e baseadas somente no melhor valor de *fitness* da população.

O funcionamento do PRoFIGA é baseado no dilema entre *exploration* e *exploitation* (exploração e refinamento) mencionado no capítulo anterior. A idéia é que, quanto maior o tamanho da população, maior a área do espaço de busca coberta e maior a exploração deste

espaço. Por outro lado, quando o tamanho da população for pequeno, o algoritmo se concentra nas soluções que já possui e realiza o refinamento destas. PRoFIGA tem como principal estratégia a criação de grandes populações no início de sua execução (*exploration*) e com o passar das gerações vai diminuindo o tamanho da população e se concentrando nos melhores pontos encontrados até então, refinando-os (*exploitation*).

Na Figura 8, encontra-se o pseudo-código do algoritmo PRoFIGA. Nesse algoritmo, o tamanho da população varia de acordo com três critérios. O primeiro indica que, se houve melhoria na qualidade do melhor indivíduo na ultima geração, a dimensão da população aumentará de forma proporcional à melhoria. Em (6), temos a equação que calcula o valor desse incremento. O segundo critério estipula que, se durante determinado número de gerações não se verificar uma melhoria do valor de aptidão da melhor solução, a população também aumenta. Já o último critério estipula que, se os dois casos anteriores falharem, então a população diminui em uma porcentagem de 1-5%.

```
01 g=0;  //geração
02 gerar população inicial P(g);
03 para cada indivíduo i da população P(g) faça
04     avaliar aptidão do indivíduo i;
05 fim para
06 enquanto critério de parada não for satisfeito faça
07     g=g+1;
08     selecionar indivíduos para reprodução a partir de P(g-1);
09     aplicar operador de reprodução;
10     aplicar operador de mutação sobre novos indivíduos;
11     avaliar novos indivíduos;
12     inserir novos indivíduos na população P(g);
13     se melhor indivíduo melhorou então
14            aumentar o tamanho da população;
15     se então não melhorou por V avaliações então
16            aumentar o tamanho da população;
17     então
18            diminuir o tamanho da população;
19     fim se
20     avaliar novos indivíduos;
21 fim enquanto
```

Figura 8 – Pseudo-código do algoritmo PRoFIGA.

O cálculo do fator de crescimento, X, é o seguinte:

$$X = increaseFactor(maxEvalNum - currEvalNum)\frac{maxFitnessNew - maxFitnessOld}{initMaxFitness}, \quad (6)$$

onde *increaseFactor* é um parâmetro (informado pelo usuário) no intervalo (0,1], e *maxEvalNum* e *currEvalNum* denotam a quantidade máxima de avaliações e a quantidade de

avaliações da geração atual, respectivamente. No caso de não se saber o real valor de *maxEvalNum*, pode-se atribuir um valor bem alto a este parâmetro. Já os parâmetros *maxFitnessNew*, *maxFitnessOld* e *initMaxFitness* denotam, respectivamente, o maior valor de *fitness* encontrado na geração atual, o maior valor de *fitness* encontrado até a geração anterior e aquele encontrado na população inicial. Além desses parâmetros utilizados na Equação (6), o PRoFIGA possui outros quatro parâmetros importantes que são utilizados durante a execução do algoritmo, a saber (Eiben, Marchiori e Valkó 2004): *decreaseFactor*, *V*, *minPopulationSize* e *maxPopulationSize*. Esses parâmetros denotam, respectivamente, a taxa de decrescimento do tamanho da população, a quantidade de avaliações sem que o valor de *fitness* do melhor indivíduo melhore antes que a população cresça, o tamanho mínimo da população e o tamanho máximo desta.

A população pode crescer ou diminuir de diferentes maneiras. Para aumentar o seu tamanho, podemos clonar os melhores indivíduos ou selecionar alguns aleatoriamente, ou mesmo por meio de torneio. Outra possibilidade é gerar novos indivíduos de forma aleatória. Para diminuir o tamanho, podemos remover os piores indivíduos ou escolher alguns aleatoriamente, a fim de se preservar a diversidade.

O restante do algoritmo funciona como um AG normal. Alguns indivíduos são selecionados, se recombinam por *crossover*, e daí a prole gerada sofre mutações, sendo avaliada e reinserida na população. O processo continua até que o critério de parada seja atingido.

2.4. PSO com tamanho da população variável

Como apresentado anteriormente, o PSO é um algoritmo populacional proposto na linha de Inteligência Coletiva (Kennedy e Eberhart 2001), sendo que, até o presente, poucos trabalhos já foram publicados investigando o impacto do tamanho da população de partículas. Em verdade, aparentemente, o artigo (Coelho e Oliveira 2008) parece ter sido o primeiro a explorar o conceito de população variável no âmbito do PSO. Neste trabalho, adaptamos a idéia do PRoFIGA e do APGA ao PSO, o que deu origem a dois novos algoritmos, o PRoFIPSO e o APPSO, respectivamente.

A concepção do PRoFIPSO foi direta (ver Figura 9), aplicando-se a mesma lógica de variação do tamanho da população do PRoFIGA ao PSO, fazendo, porém, algumas adaptações. O algoritmo é iniciado normalmente, sendo que, ao fim de cada geração, são aplicadas as técnicas de variação populacional do PRoFIGA. Se houver melhora no valor do *fitness* do melhor indivíduo, ou caso a população fique por um certo número de avaliações sem melhorar, o tamanho do *swarm* cresce. Por outro lado, caso nenhuma dessas condições sejam satisfeitas, a população decresce de acordo com uma porcentagem prefixada (1% - 5%). Por outro lado, o crescimento se dá da seguinte forma: metade das novas partículas são clonadas de boas partículas escolhidas via torneio e a outra metade é composta por novas partículas criadas aleatoriamente, a fim de manter a diversidade e aumentar a exploração do espaço de busca. Os componentes do vetor de velocidade e do vetor da melhor posição já visitada (P_i) das partículas clonadas são feitos nulos. As partículas removidas são sempre as piores da população atual.

```
01 criar aleatoriamente as partículas no espaço de busca;
02 avaliar as partículas;
03 g=0;
04 enquanto critério de parada não for satisfeito faça
05     g=g+1;
06     para cada partícula atualizar o vetor de velocidade;
07     para cada partícula atualizar a posição;
08     para cada partícula i faça
09            se pb(i)> pb(i-1) então atualizar pb(i);
10     se pg(g)> pg(g-1) então atualizar pg(g);
11     se melhor indivíduo melhorou então
12            aumentar o tamanho população;
13     se não melhorou por V avaliações então
14            aumentar o tamanho população;
15     então
16            diminuir o tamanho população;
17 fim
```

Figura 9 - Pseudo-código do algoritmo PRoFIPSO.

Para o APPSO (Figura 10), cada partícula passa a ter uma idade e um tempo de vida (baseado em seu valor de aptidão), seguindo a lógica do APGA. Como em qualquer PSO, as partículas se movimentam pelo espaço de busca mediante o cálculo de suas velocidades. Contudo, uma modificação significativa é que, no APPSO, a cada geração, duas partículas são selecionadas para reprodução via *crossover*, gerando novas partículas, as quais, após sofrerem mutação, são incorporadas à população. Particularmente, adotamos o operador de *crossover* aritmético, já que este se mostra adequado para trabalhar com codificações em ponto flutuante

(Michalewicz 1996). No mais, as idades das partículas são incrementadas a cada geração e as partículas com idade igual ao tempo de vida são excluídas da população.

No APPSO, todas as partículas são reavaliadas a cada iteração, pois, como elas estão em movimento, suas posições mudam e elas precisam receber um novo valor de *fitness*. Porém, apenas as partículas que conseguiram melhorar o seu valor de *fitness* vão receber um novo tempo de vida e terão a idade reiniciada. As demais continuarão com o mesmo tempo de vida calculado anteriormente e terão a idade incrementada a cada geração.

```
01 criar aleatoriamente as partículas no espaço de busca;
02 avaliar as partículas;
03 calcular o tempo de vida de cada partícula;
04 g=0;
05 enquanto critério de parada não for satisfeito faça
06     g=g+1;
07     incrementar a idade das partículas;
08     para cada partícula atualizar o vetor de velocidade;
09     para cada partícula atualizar a posição;
10     para cada partícula i faça
11            se pb(i)> pb(i-1) então
12                   atualizar pb(i);
13                   recalcular tempo de vida;
14                   idade da partícula = 0;
15     se pg(g)> pg(g-1) então atualizar pg(g);
16     selecionar partículas para reprodução;
17     aplicar operador de crossover;
18     aplicar operador de mutação sobre as novas partículas;
19     avaliar as novas partículas;
20     inserir as novas partículas na população;
21     retirar as partículas com idade igual ao tempo de vida;
22 fim
```

Figura 10 - Pseudo-código do algoritmo APPSO.

2.5. DE com tamanho da população variável

O processo para adaptação do PRoFIGA e APGA ao DE ocorreu de modo semelhante ao visto na seção anterior para o caso do PSO. Aqui também tivemos dois novos algoritmos criados: o PRoFIDE e o APDE.

No PRoFIDE (Figura 11), os passos do algoritmo padrão do DE ocorrem normalmente e, antes de terminar a iteração, são aplicados sobre a população os mecanismos de variação

(crescimento ou decrescimento) do PRoFIGA[8]. Assim como no PRoFIPSO, quando a população deve crescer, metade dos novos indivíduos são clones de bons indivíduos escolhidos por torneio e a outra metade é de novos indivíduos gerados aleatoriamente. Os indivíduos removidos também são sempre os piores da população.

```
01 criar aleatoriamente os indivíduos no espaço de busca
02 avaliar indivíduos
03 enquanto critério de parada não satisfeito
04     Para cada vetor da população P faça
05            r1 = rand(NP);
06            r2 = rand(NP);
07            r3 = rand(NP);
08            ui = xr3 + F*(xr1 - xr2);
09            se (f(ui) <= f(xi))então
10                   pAux = ui;
11            senão
12                   pAux = xi;
13     fim para;
14     se melhor indivíduo melhorou então
15            aumentar o tamanho da população;
16     se não melhorou por V avaliações então
17            aumentar o tamanho da população;
18     então
19            diminuir o tamanho da população;
20 fim enquanto
```

Figura 11 - Pseudo-código do algoritmo PROFiDE.

Assim como no APPSO, no APDE (Figura 12), dois indivíduos são selecionados via torneio a cada geração e passam pelo operador do *crossover*, gerando dois novos indivíduos, os quais sofrem mutação e são inseridos na população. Os indivíduos passam a ter as variáveis “idade” e “tempo de vida”. O tempo de vida é proporcional ao valor de aptidão e é calculado de acordo com o método bi-linear, como no APGA. Assim como no PRoFIPSO, as partículas são reavaliadas a cada geração, pois suas posições mudam. Porém, o tempo de vida apenas é recalculado quando o valor do *fitness* melhora. Nesse caso, a idade do individuo é feita nula.

[8] São os mesmos procedimentos apresentados na Seção 2.3.

```
01 criar aleatoriamente os indivíduos no espaço de busca
02 avaliar indivíduos
03 calcular o tempo de vida
04 enquanto não encontrou critério de parada
05     Para cada vetor da população P faça
06           r1 = rand(NP);
07           r2 = rand(NP);
08           r3 = rand(NP);
09           ui = xr3 + F*(xr1 - xr2);
10           se (f(ui) <= f(xi))então
11                 pAux = ui;
12                 recalcular tempo de vida;
13                 idade da partícula = 0;
14           senão
15                 pAux = xi;
16     fim para;
17     selecionar partículas para reprodução;
18     aplicar operador de crossover;
19     aplicar operador de mutação sobre as novas partículas;
20     avaliar as novas partículas;
21     inserir as novas partículas na população;
22     retirar as partículas com idade igual ao tempo de vida;
23 fim enquanto
```

Figura 12 - Pseudo-código do algoritmo APDE.

2.6. Conclusão

Neste capítulo, foram apresentados modelos estendidos de AGs com tamanho da população variável, sendo que o APGA e o PRoFIGA foram posteriormente adaptados ao contexto do PSO e DE, dando origem às versões APPSO, PRoFIPSO, APDE e PRoFIDE. No próximo capítulo, são apresentados e discutidos os resultados obtidos com esses novos algoritmos, comparando-os com suas versões com tamanho fixo da população.

3. Experimentos computacionais

Neste capítulo, apresentamos e discutimos os resultados obtidos com os algoritmos propostos no presente trabalho. Vários testes foram realizados para que pudéssemos mensurar (em termos de eficiência e eficácia) o desempenho. Em um primeiro momento, realizamos testes com problemas *benchmark* de otimização numérica, a fim de analisarmos o comportamento dos algoritmos sob vários aspectos. Fizemos também a aplicação dos algoritmos em dois problemas não-triviais de otimização relativos à área de Aprendizado de Máquina (Mitchell 1997). São eles: descoberta de protótipos para agrupamento (c*lustering*) de dados (Paterlini e Krink 2006) e treinamento de redes neurais *feedforward* (Haykin 1999). Utilizamos o *framework* computacional PEA desenvolvido para a dissertação de mestrado de Valkó (2003) como substrato para a implementação dos nossos algoritmos. Esse *framework* também serviu de base para os experimentos conduzidos em (Eiben, Marchiori e Valkó 2004).

3.1. Experimentos com funções benchmark

Funções *benchmark* são bastante utilizadas para fins de avaliação de algoritmos de busca e otimização. Esse tipo de teste nos permite tanto avaliar o impacto da escolha de parâmetros específicos dos algoritmos como avaliar o seu comportamento computacional como um todo. Nesse contexto, foi realizado por Digalakis e Margaritis (2002) um estudo experimental avaliando o desempenho de algoritmos genéticos sobre 14 funções diferentes. Nesse artigo, os autores também realizaram um estudo bibliográfico apontando trabalhos anteriores nos quais essas funções foram utilizadas, seus objetivos e configurações. Por outro

lado, Michalewicz (1996) também apresenta um repertório considerável de funções *benchmark*, algumas delas diferentes daquelas expostas por Digalakis e Margaritis (2002).

Para o presente trabalho, foram utilizadas várias funções *benchmark* para fins de análise do desempenho dos novos algoritmos, muitas das quais presentes nos trabalhos acima citados. Alguns exemplos são as funções Schwefel, Sphere, Corana, Kennedy e Goldstein-Price, as quais foram facilmente resolvidas pelos algoritmos PSO e DE. Optamos, então, por nos restringir àquelas funções que fossem mais difíceis para estes algoritmos e, ao mesmo tempo, possuíssem características e graus de dificuldades distintos entre elas. Essas funções, em um total de quatro, são apresentadas na Tabela 1. Julgamos que essas funções são suficientes para avaliarmos o comportamento quando se confronta algoritmos de diferentes complexidades ou quando determinados parâmetros de configuração são alterados.

Tabela 1 - Funções *benchmark*

Nome	*Função*	*Limites*
Ackley	$f = -20exp\left(-0.2\left(\sqrt{\frac{1}{n}\sum_{i=0}^{n} {x_i}^2}\right)\right) - exp\left(\frac{1}{n}\sum_{i=0}^{n}\cos(2\pi x_i)\right) + 20 + exp(1)$	$-32.768 \leq x_i \leq 32.768$
Griewangk	$f = 1 + \sum_{i=0}^{n}\left(\frac{{x_i}^2}{4000}\right) - \prod_{i=0}^{n}\left(\cos\left(\frac{x_i}{\sqrt{i}}\right)\right)$	$-600 \leq x_i \leq 600$
Rastrigin	$f = \sum_{i=0}^{n}({x_i}^2 - 10\cos(2\pi x_i))$	$-5.12 \leq x_i \leq 5.12$
Rosenbrock	$f = \sum_{i=0}^{n} 100\,({x_i}^2 - x_{i+1})^2 + (1 - x_i)^2$	$-2.048 \leq x_i \leq 2.048$

Todas as quatro funções da Tabela 1 são de minimização e, exceto a última, são multimodais, ou seja, possuem vários pontos de mínimos locais e um valor de mínimo global.

A função Ackley é obtida modulando uma função exponencial com uma onda cosseno de amplitude moderada (Bäck 1996). Sua forma se caracteriza por ondas de baixa amplitude na região mais externa (devido à dominância exponencial) e um vale no centro, onde a dominância do cosseno se faz mais influente (ver Figura 13). Nossos testes indicam que esta função é de difícil resolução por parte dos algoritmos PSO e AG. O valor do seu mínimo global é zero.

Griewangk é uma função de complexidade $O(n\ln(n))$, onde n é a quantidade de dimensões (parâmetros). Seu gráfico tem formato de parábola, sendo que seu mínimo global fica na parte de baixo (ver Figura 15). Na Figura 14, temos uma visão mais aproximada dessa função, em que podemos observar que toda a parábola é formada por vários "morros". O valor do seu mínimo global também é zero.

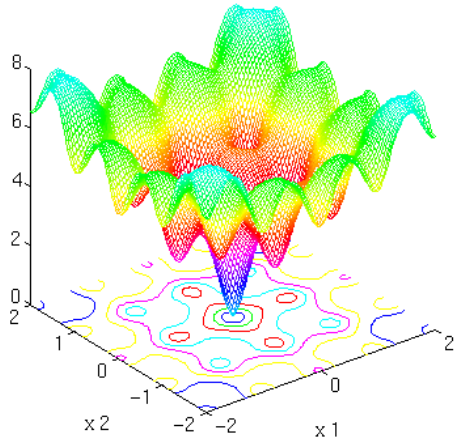

Figura 13 - Função Ackley.

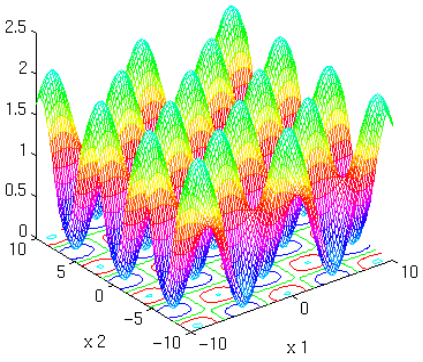

Figura 14 - Função Griewangk.

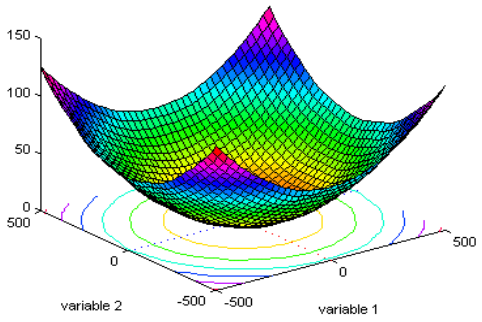

Figura 15 - Função Griewangk em maior escala.

Rastrigin (Figura 16) é uma função de difícil otimização para AGs devido a seu grande espaço de busca e à grande quantidade de mínimos locais (Digalakis e Margaritis 2002). Sua complexidade é $O(n\ln(n))$, onde n é a quantidade de parâmetros, e o valor do seu mínimo global é zero.

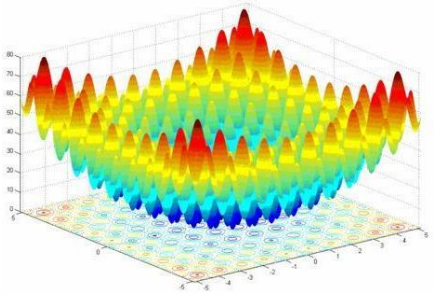

Figura 16 - Função Rastrigin.

Finalmente, a função Rosenbrock (Figura 17) é considerada um problema unimodal difícil, pois possui um pico estreito e fino que circula toda a parábola. Algoritmos que não são capazes de descobrir boas direções terão problemas em encontrar seu mínimo global. O valor de seu mínimo global também é zero.

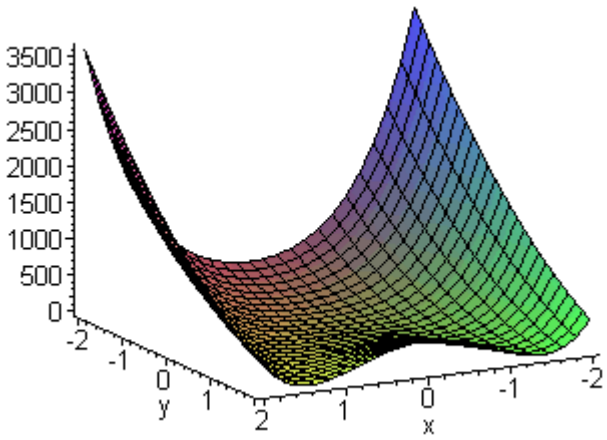

Figura 17 - Função Rosenbrock.

3.1.1. Configurações

Para podermos realizar uma análise comparativa significativa, utilizamos no total nove algoritmos. São eles: GA, APGA, PRoFIGA, PSO, APPSO, PRoFIPSO, DE, APDE e

PRoFIDE. Para os algoritmos DE, APDE e PRoFIDE, utilizamos três esquemas distintos relacionados ao modelo padrão, os quais são discutidos mais apropriadamente em (Price, Storn e Lampinen 2005), a saber: DE/rand/1/exp, que serviu de base para a discussão no Capítulo 1; DE/rand/2/exp; e DE/rand/1/bin. Basicamente, o que se altera nos dois últimos esquemas em relação ao primeiro é o tipo de operador de *crossover* efetivamente usado (exponencial *versus* binário), o tipo de indivíduo escolhido para ser o vetor-referência (aleatório *versus* melhor indivíduo) e o número de termos de diferenciação (um *versus* dois).

Nossos testes foram realizados com a dimensão dos indivíduos variando no conjunto $\{10,20,50,100\}$. Todos os algoritmos utilizaram *crossover* aritmético, visto que ele é mais apropriado do que os operadores de *crossover* baseados em pontos de corte para problemas com representação numérica (Eiben e Smith 2003). O critério de parada adotado para todos os algoritmos foi cinco milhões de avaliações no total ou 300 mil avaliações sem que a aptidão do melhor indivíduo melhorasse. A versão do PSO utilizada foi a com coeficiente de constrição e os valores de c_1 e c_2 usados foram ambos 2,05 (Clerc e Kennedy 2002); assim, pela Equação (4) temos $X = 0{,}7298$. Os parâmetros específicos para os modelos DE foram sempre $F = 0{,}5$ e $CR = 0{,}5$.

Para os algoritmos PRoFIGA, PRoFIPSO e PRoFIDE, os seguinte parâmetros foram configurados como a seguir, conforme indicado em (Eiben, Marchiori e Valkó 2004): $increaseFactor = 0{,}4$, $decreaseFactor = 0{,}02$, $V = 500$, $minPopulationSize = 15$ e $maxPopulationSize = 1.000$. Ademais, especificamente para APGA, APPSO e APDE, temos: $minLt = 1$ e $maxLt = 11$. Todas essas configurações dos parâmetros de controle foram obtidas através de experiências prévias ou aceitando as configurações propostas por outros autores. Muito de nosso conhecimento sobre a sensibilidade do PRoFIPSO e APPSO aos valores dos parâmetros escolhidos vem do estudo apresentado em (Coelho e Oliveira 2008). No presente trabalho, aprimoramos um pouco mais a escolha desses parâmetros e também aplicamos alguns deles ao PRoFIDE e APDE.

Finalmente, vale mencionar que, para cada tríade (problema, dimensão do problema, tamanho da população inicial), realizamos 10 execuções. O tamanho da população inicial variou no conjunto $\{20,50,100\}$.

3.1.2. Resultados

Os resultados obtidos foram organizados em tabelas que se encontram no Apêndice I. Suas colunas estão nomeadas como *A, B, C* e *P*, que significam respectivamente a média dos melhores valores de *fitness* encontrados, a média das quantidades de avaliações necessárias para se encontrar o melhor valor de *fitness*, a média do tamanho da população quando se encontrou o melhor valor de *fitness*, e o tamanho da população inicial. Para os algoritmos sem tamanho da população variável, a coluna *C* não se faz necessária.

Pelos valores expressos nas tabelas da função Ackley, podemos observar que o PSO padrão teve dificuldades neste problema em relação aos demais algoritmos. Essa dificuldade fica mais evidente quando a dimensão é maior ou igual a 50. Já os resultados do PRoFIPSO e APPSO, em termos de eficácia, foram sempre melhores que o do PSO, mas, mesmo assim, não conseguiram superar os resultados do PRoFIGA e APGA. Em geral, o PRoFIPSO se mostra um algoritmo que utiliza muitas avaliações para convergir à solução ótima. Isso é fácil de entender já que todas as partículas são avaliadas e reavaliadas a cada geração. Outro fato que contribui para a grande quantidade de avaliações é o de tanto o PRoFIPSO quanto o PRoFIDE terem um grande crescimento no tamanho de sua população no início de sua execução. Em verdade, notamos que esta observação se aplica a todas as funções *benchmark.*

Na Figura 18, podemos observar como se deu a relação “quantidade de indivíduos versus número de avaliações” para o Ackley com dimensão 100. No inicio da execução, uma grande quantidade de partículas são criadas pelo PRoFIPSO e PRoFIDE, e com o passar das gerações essa quantidade vai diminuindo até que se estabiliza. Para esse mesmo caso, o PRoFIDE obteve resultados sempre piores em relação ao APDE e ao próprio DE. O que nos chama a atenção no PRoFIDE são os valores do desvio-padrão relativos à melhor solução encontrada a cada execução, que são sempre bem elevados. Isso implica dizer que os resultados encontrados nas 10 execuções do algoritmo variaram bastante, demonstrando a dificuldade deste em escapar dos mínimos locais do problema e a sua sensibilidade à configuração da população inicial. Já o APDE conseguiu resultados melhores que o PRoFIDE, tanto em termos da qualidade das melhores soluções encontradas como em termos do número de avaliações. O APDE/rand/1/bin, por exemplo, conseguiu resultados próximos aos resultados do APGA (em termos de melhor solução encontrada) enquanto o PRoFIDE, que utilizou mais avaliações, não conseguiu o mesmo desempenho.

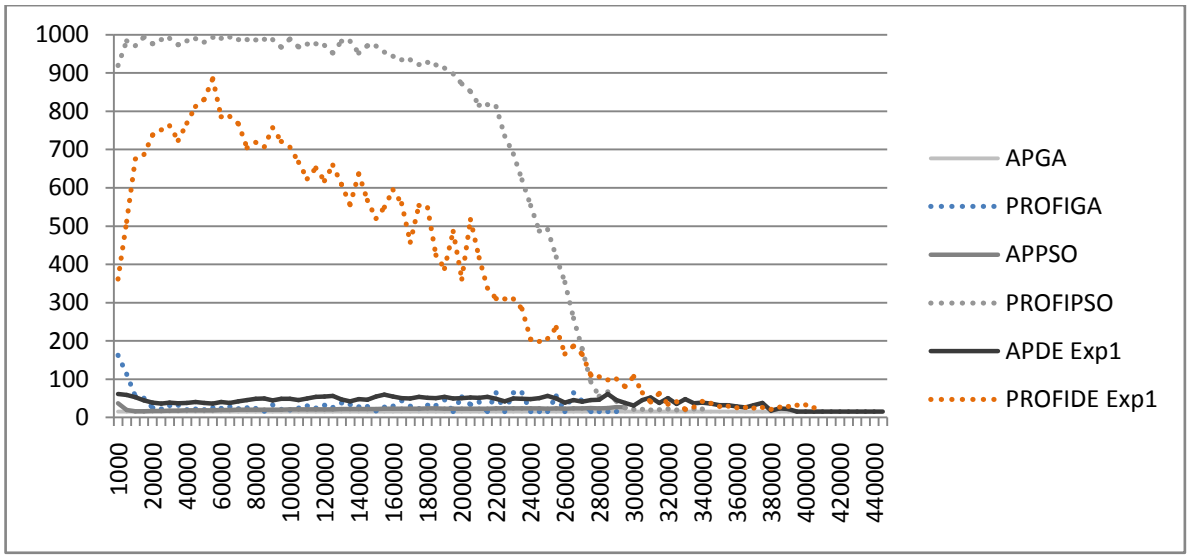

Figura 18 - Gráfico da variação do tamanho da população em relação ao número de avaliações para a função Ackley com dimensão 100 e tamanho da população inicial 50.

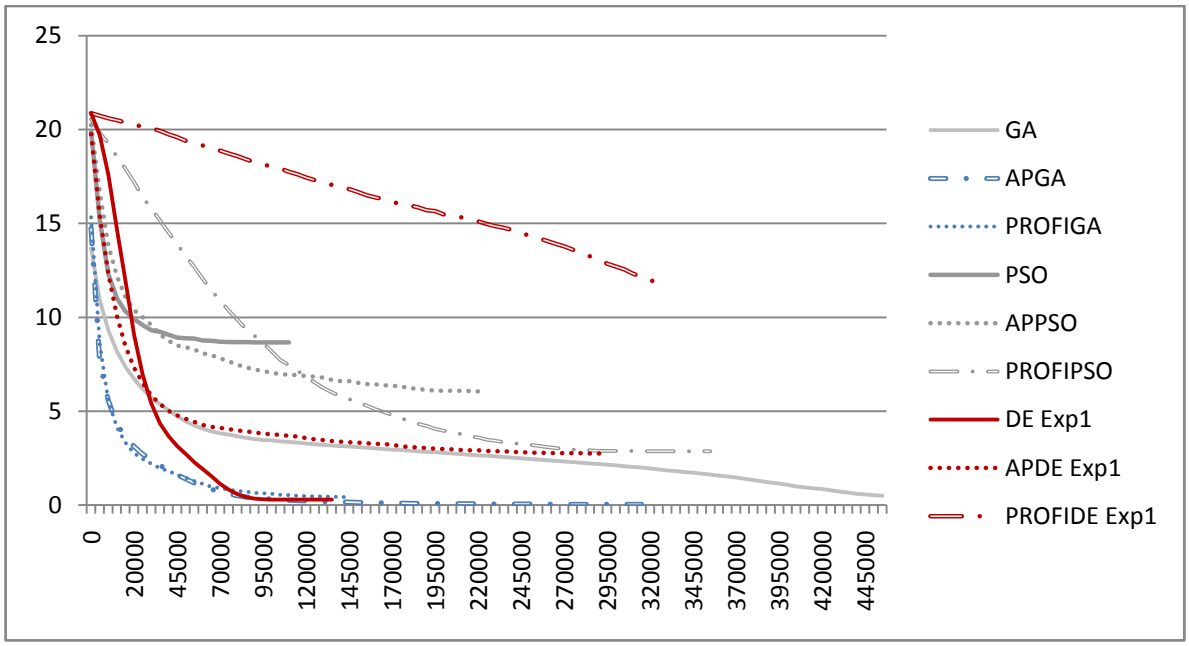

Figura 19 - Comportamento médio do melhor valor de aptidão em função do número de avaliações para a função Ackley com dimensão 100 e tamanho da população inicial 50.

Na Figura 19, podemos observar o comportamento médio dos algoritmos com relação ao melhor valor de *fitness* para o caso de dimensão 100. Tanto o PRoFIPSO como o PRoFIDE possuem uma queda tênue com o passar das gerações (ou seja, demoram mais para

convergir), enquanto os outros algoritmos apresentam uma queda brusca antes das 45.000 avaliações. Podemos afirmar que o APGA e o PRoFIGA tiveram um bom desempenho em termos de eficiência, pois, para a marca acima, já haviam, localizado uma solução mais satisfatória em relação às dos demais.

A função Griewangk foi uma função de fácil resolução para todos os algoritmos, com exceção do PRoFIDE. No geral, as versões do PRoFIDE, mais uma vez, obtiveram grandes valores no desvio-padrão relativos à melhor solução encontrada, enquanto o APDE obteve resultados semelhantes ao DE, porém, lançando mão de mais avaliações. Nesta função, os melhores resultados, em termos de eficácia, foram encontrados geralmente pelo PRoFIPSO, embora este tenha utilizado no total mais avaliações do que os demais algoritmos. Assim como para a função Ackley, nas primeiras gerações a quantidade de partículas da população do PRoFIPSO ficou em geral por volta de 1.000, que era o máximo permitido, e depois caía gradualmente com o passar das gerações. A Figura 20 traz os resultados médios em termos de melhor valor de aptidão para essa função, considerando dimensão 20 e tamanho da população inicial 20. Verifica-se nesse caso que grande parte dos algoritmos exibe bom índice de eficiência, já que consegue localizar soluções satisfatórias antes da marca das 5.000 avaliações.

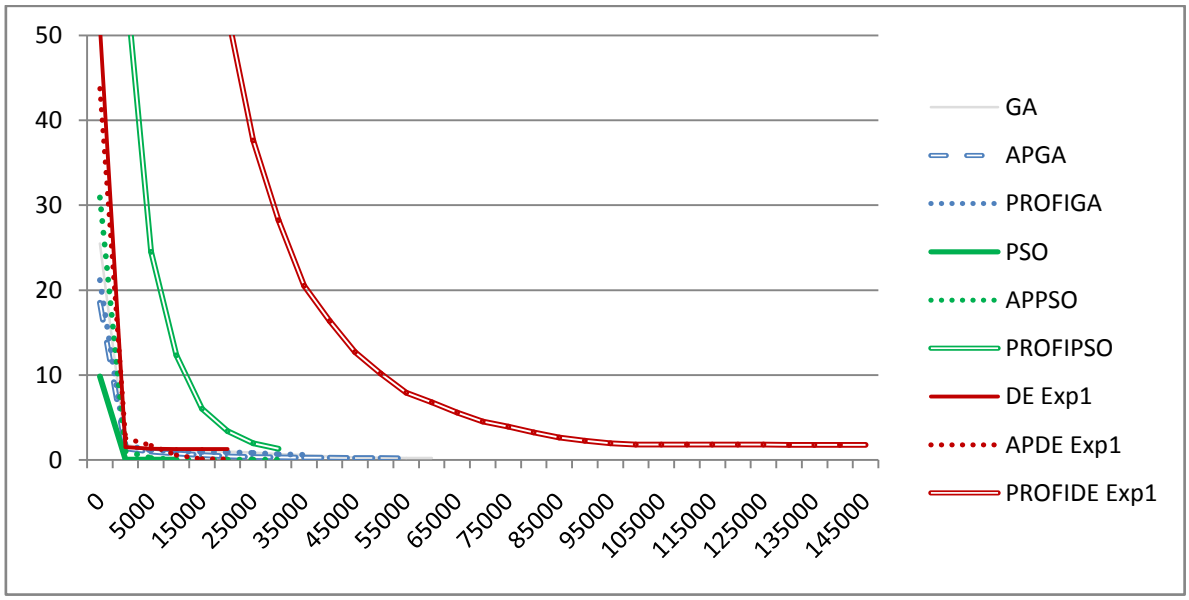

Figura 20 - Comportamento médio do melhor valor de aptidão em função do número de avaliações para a função Griewangk com dimensão de tamanho 20 e população inicial 20.

Para a função Rastrigin, os resultados encontrados pelos AGs (padrão e com tamanho da população variável) foram melhores do que os demais algoritmos. Assim como na função Ackley, a função Rastrigin é de difícil resolução para os modelos de PSO. Para a dimensão 100, o DE/rand/1/exp foi quem, no geral, conseguiu encontrar as melhores soluções utilizando menos avaliações, como podemos observar na Figura 21. Os resultados do APGA também foram bons, porém este utilizou mais avaliações do que o DE em geral.

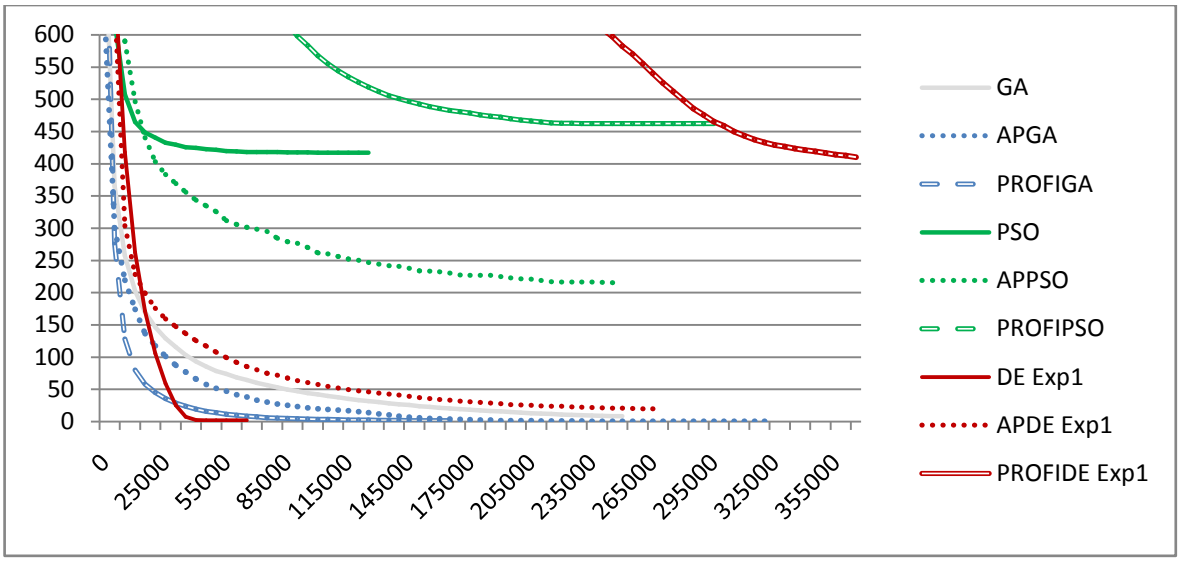

Figura 21 - Comportamento médio do melhor valor de aptidão em função do número de avaliações para a função Rastrigin com dimensão de tamanho 100 e população inicial 20.

Finalmente, a função Rosenbrock se apresentou como de difícil resolução para os AGs, mesmo para o PRoFIGA e APGA — estes apresentaram dificuldades para as dimensões 50 e 100, ficando atrás até do AG padrão para esses casos. Este mesmo quadro se repetiu para o DE, APDE e PRoFIDE. Já os algoritmos PSO, APPSO e PRoFIPSO foram os que em geral conseguiram os melhores resultados, dentre todos. Para as dimensões 10 e 20, tanto a melhor solução encontrada pelo APPSO como pelo PRoFIPSO ficaram bem próximas ao mínimo global, e o desvio-padrão das execuções foi praticamente inexistente. Porém, quando a dimensão passou para 50 e 100, o PRoFIPSO não conseguiu mais manter o nível de desempenho, encontrando soluções piores do que as do próprio PSO. Por outro lado, o APPSO continuou encontrando boas soluções, embora com não tanta constância (desvio-padrão das melhores soluções aumentou), principalmente para dimensão 100. Na Figura 22, podemos observar o comportamento descrito acima.

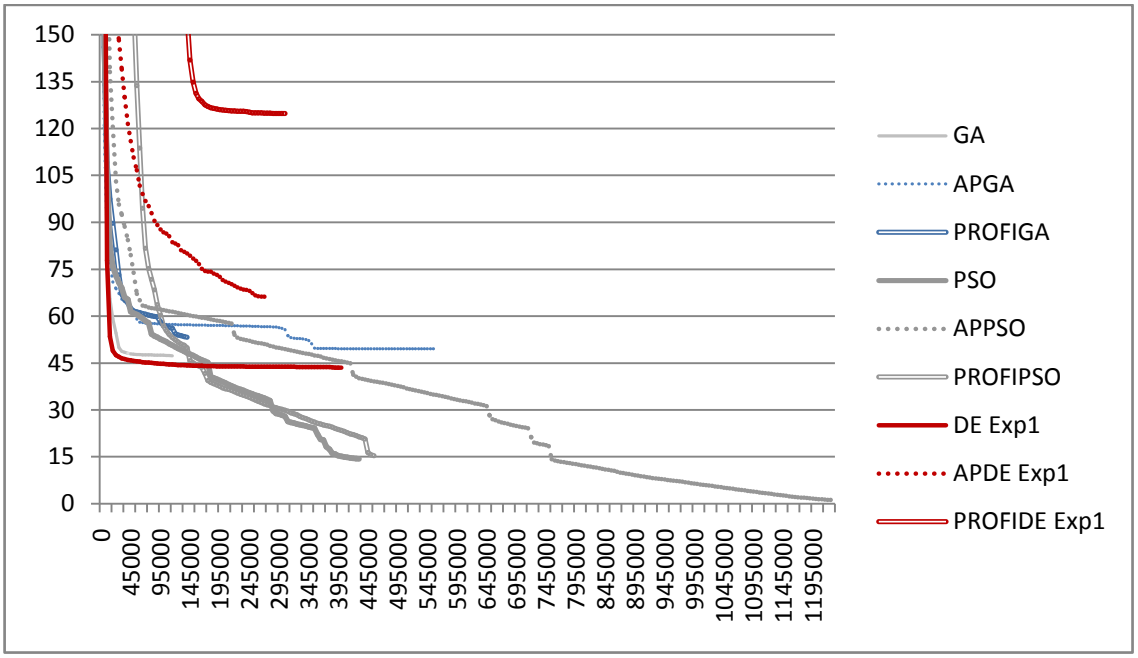

Figura 22 - Comportamento médio do melhor valor de aptidão em função do número de avaliações para a função Rosembrock com dimensão 50 e tamanho da população inicial 20.

3.2. Descoberta de protótipos para agrupamento de dados

Agrupamento (*clustering*) de dados refere-se à organização de padrões (geralmente representados como vetores de medidas) em grupos (*clusters*) com base na sua similaridade (Jain, Murty e Flynn 1999). Esta é uma tarefa comumente usada para fins de análise estatística de dados e classificação não-supervisionada, sendo adotada em vários campos de estudo da computação, como: aprendizado de máquina, mineração de dados, reconhecimento de padrões, análise de imagens e bioinformática (Jain, Murty e Flynn 1999).

Algoritmos de *clustering* podem ser hierárquicos ou particionais. Os hierárquicos encontram sucessivos *clusters* (subconjuntos) partindo sempre da partição encontrada anteriormente, enquanto os particionais trabalham com um nível de partição (Berkhin 2002), sendo este o tipo investigado neste trabalho.

Um algoritmo de *clustering* particional determina um subconjunto de soluções maximizando o grau de similaridade entre os objetos pertencentes a cada sub-conjunto, ao mesmo tempo que minimizando o grau de similaridade entre objetos pertencentes a diferentes

grupos (Jain, Murty e Flynn 1999). Desta forma, podemos modelar esse problema como um problema de otimização, utilizando-se como função de avaliação critérios estatísticos que consideram a variância dentro e entre os subconjuntos.

Possivelmente, o algoritmo mais conhecido dentre os de *clustering* particional é o *k-means* (Jain, Murty e Flynn 1999), por sua facilidade de implementação e eficiência. Porém, este possui a limitação de convergir para ótimos locais e alta sensibilidade às condições iniciais, fazendo-se necessárias às vezes várias execuções a partir de pontos iniciais diferentes para se encontrar uma solução adequada (Paterlini e Krink 2006).

A utilização de algoritmos evolutivos para *clustering* vem se apresentando como uma alternativa mais eficaz em relação ao *k-means*, sendo que essa abordagem não é tão nova, como atestam Paterlini e Krink (2006). Nesse artigo, além de discutirem sobre trabalhos anteriores investigando a adoção de algoritmos genéticos para a tarefa de *clustering*, os autores propõem a adoção dos algoritmos DE e PSO para o mesmo fim, demonstrando empiricamente, via uma série de testes comparativos, que estes algoritmos são mais eficazes (i.e., geram partições mais adequadas) do que os AGs.

Para este estudo de caso, nós seguimos a abordagem implementada por Paterlini e Krink (2006), em que cada indivíduo da população de uma metaheurística representa um conjunto de pontos especiais, conhecidos como centróides ou pontos representativos. É a partir destes pontos que são criados os subconjuntos (*clusters*) contendo os objetos da base de dados que se deseja agrupar. A partição resultante é então submetida à função de avaliação, que calcula o valor de aptidão do indivíduo de acordo com a qualidade da partição gerada. A representação dos pontos representativos em cada indivíduo da população é realizada de uma forma direta, ou seja, cada indivíduo codifica todas as *d* coordenadas (referentes aos atributos) de todos os *k clusters* da partição a ser formada, sendo *k* atribuído previamente pelo projetista e *d* estabelecido de acordo com a base de dados sob análise. Na Figura 23, trazemos um exemplo fictício de um indivíduo para um problema em que $k = 3$ e $d = 3$.

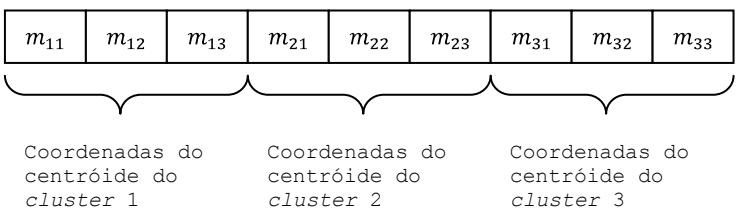

Figura 23 - Representação de um indivíduo fictício com 3 *clusters* e 3 atributos.

Os *clusters* são criados com base na regra de Forgy (1965), segundo a qual os elementos que compõem cada *cluster* são escolhidos de acordo com sua "proximidade" aos pontos representativos. Tanto no trabalho de Paterlini e Krink (2006) quanto no presente trabalho, a "proximidade" entre um objeto e um ponto representativo é calculada utilizando a distância euclidiana. Por outro lado, a avaliação dos *clusters* criados é feita de acordo com qualquer um dos métodos que serão apresentados na próxima subseção. Vale ressaltar de antemão que, se algum dos *clusters* criados por um indivíduo não tiver nenhum objeto da base a ele alocado, este indivíduo receberá um valor de *fitness* muito ruim como penalidade.

3.2.1. Funções de avaliação para clustering

Diferentes critérios estatísticos foram propostos para medir o grau de adequação das partições geradas permitindo compará-las entre si (Jain, Murty e Flynn 1999). Alguns destes critérios envolvem operações simples, como o cálculo do traço ou determinante das matrizes de espalhamento intra-grupo (*W*) e de espalhamento entre grupos (*B*) (Paterlini e Krink 2006).

A matriz de espalhamento intra-grupo é definida como $W = \sum_{l=1}^{g} W_k$, onde W_k denota a variância da matriz de características (atributos) dos objetos alocados ao *cluster* $C_k (k = 1, \dots, g)$. Então, considerando cada objeto $x_l^{(k)}$ de um *cluster* C_k e que n_k é a quantidade de elementos que compõem o *cluster* C_k, temos:

$$\boldsymbol{W_k} = \sum_{l=1}^{\boldsymbol{n_k}} \left(\boldsymbol{x_l^{(k)}} - \boldsymbol{\bar{x}^{(k)}}\right)\left(\boldsymbol{x_l^{(k)}} - \boldsymbol{\bar{x}^{(k)}}\right)', \quad \textbf{(7)}$$

onde $\bar{x}^{(k)} = (\sum_{l=1}^{n_k} x_l^{(k)})/n_k$ é o centróide (ou seja, o vetor-médio dos indivíduos) do *cluster* C_k.

Já a matriz de espalhamento entre grupos é definida como:

$$\boldsymbol{B} = \sum_{k=1}^{g} \boldsymbol{n_k}\left(\boldsymbol{\bar{x}^{(k)}} - \boldsymbol{\bar{x}}\right)\left(\boldsymbol{\bar{x}^{(k)}} - \boldsymbol{\bar{x}}\right)', \quad \textbf{(8)}$$

onde $\bar{x} = (\sum_{i=1}^{n} x_i)/n$.

Assim, pode-se definir a matriz de espalhamento total das *n* observações que compõem a base como $T = B + W$.

Particularmente, neste trabalho, foram considerados três critérios estatísticos para mensurar a adequação das partições criadas para o cálculo do valor de aptidão dos indivíduos, embora os resultados apresentados na próxima subseção são referentes a apenas um deles. O primeiro é TRW[9] (Friedman e Rubin 1967), definido como $minimizar_G\ traço(W)$, sendo *G* o conjunto de todas as partições factíveis. Esse critério assume implicitamente uma baixa correlação entre as medições dos atributos, dando igual importância às variâncias dentro dos *clusters*. O processo utilizado pelo *k-means* corresponde matematicamente a essa minimização do traço de *W*.

O segundo critério utilizado é o da razão de variâncias VCR[10] (Calinski e Harabasz 1974). Como se trata de um critério de maximização, em nossos testes, ele foi convertido para minimização (multiplicando-se o valor obtido por cada partição por -1). Esse critério é definido como $maximizar\ \frac{traço(B)/(g-1)}{traço(W)/(n-g)}$, onde $(n-g)$ denota o número de graus de liberdade associado à matriz W e $(g-1)$ é o número de graus de liberdade da matriz *B*.

Por fim, o terceiro critério é o de Marriott, denotado por MC (Marriott 1971), podendo ser definido como $minimizar\ g^2\frac{\det(W)}{\det(T)}$. O critério de Marriott é comumente utilizado quando se almeja formar *clusters* caracterizados por uma alta correlação interna.

Para maiores detalhes sobre esses critérios, pode-se consultar (Everitt 1993)(Paterlini e Krink 2006). A seguir, são apresentados os testes realizados para este estudo de caso e os resultados obtidos ao se adotar os algoritmos heurísticos com população de tamanho variável.

3.2.2. Resultados

Para efeito de comparação, nossos experimentos foram conduzidos com base na metodologia de avaliação seguida por Paterlini e Krink (2006). Esses autores realizaram vários testes empíricos, comparando o desempenho de AG, PSO e DE padrão sobre dados artificiais e dados reais. Para este trabalho, foram realizados apenas testes sobre dados reais. Foram utilizadas quatro bases de dados já bastante conhecidas, retiradas do repositório de

[9] Do inglês, *trace-within criterion*.
[10] Do inglês, *variance ratio criterion*.

aprendizado de máquina da Universidade da Califórnia em Irvine (Asuncion e Newman 2007). São elas:

- Iris $(n = 150, p = 4, g = 3)$, que consiste de três tipos diferentes de espécies de flores de Iris: *Iris setosa*, *Iris virginica* e *Iris versicolour*. Para cada espécie, são 50 exemplos com quatro características cada (tamanho da sépala, largura da sépala, tamanho da pétala e largura da pétala).
- Câncer $(n = 683, p = 9, g = 2)$, que consiste de 683 exemplos caracterizados por nove atributos: densidade da massa, tamanho uniforme da célula, formato uniforme da célula, aderência da borda, tamanho da célula epitelial, núcleo exposto, cromatine suave, nucléolo normal e mitoses. As categorias são duas: malignas (444 exemplos) e benignas (239 exemplos).
- Vidro $(n = 214, p = 9, g = 6)$ é um conjunto de dados com seis tipos de vidro diferentes: janela para prédio processada (70 exemplos), janela para prédio não-processada (76 exemplos), janela para veículos processada (17 exemplos), contêiner (13 exemplos), aparelho de jantar (9 exemplos) e lâmpada (29 exemplos). Cada amostra tem com nove características: índice refrativo, de sódio, magnésio, alumínio, silicone, potássio, cálcio, bário, e de ferro.
- Vogal $(n = 871, p = 3, g = 6)$, que consiste de 871 sons de vogais dos índios Telugu. Cada exemplo possui três características que correspondem à primeira, segunda e terceira freqüências das vogais. A base possui seis classes: δ (72 exemplos), *a* (89 exemplos), *i* (172 exemplos), *u* (151 exemplos), *e* (207 exemplos) e *o* (180 exemplos).

As configurações dos algoritmos estudados foram mantidas da seção anterior, exceto pelo tamanho mínimo da população que é igual a 10 e pela forma de iniciação da população. Em vez de gerarmos novos indivíduos aleatoriamente, escolhemos da própria base de dados indivíduos aleatórios para compor a população inicial, como sugere o trabalho de Paterlini e Krink (2006). A dimensão da solução (indivíduo) é sempre a mesma e depende da base de dados, como mencionado na subseção anterior. Estudos preliminares indicaram que os resultados obtidos com os critérios TWC e VRC seguem qualitativamente o mesmo perfil daqueles obtidos com o critério MC. Assim, na seqüência, serão apresentados os resultados apenas para este último critério.

Os algoritmos utilizados em nosso estudo comparativo para este estudo de caso foram o PSO, APPSO, PRoFIPSO, DE, APDE e PRoFIDE. Embora o trabalho de Paterlini e Krink (2006) já contenha os resultados para o PSO e DE, refizemos os experimentos com esses algoritmos para efeito de constatação. Para os algoritmos DE, APDE e PRoFIDE, utilizamos apenas a versão DE/rand/1/exp (ver Subseção 3.1.1). Não foram realizados testes com as versões DE/rand/2/exp e a DE/rand/1/bin, pois os resultados obtidos com esses esquemas nas funções *benchmark* foram inconstantes, possuindo muitas vezes um desvio-padrão do valor de aptidão da melhor solução encontrada bem elevado.

Os resultados obtidos para os algoritmos PSO e DE foram iguais aos apresentados no trabalho de Paterlini e Krink (2006). Com relação às versões com população de tamanho variável, estas não apresentaram resultados superiores aos modelos convencionais em termos de eficácia. Nas Tabela 2 a Tabela 5, destacamos os resultados obtidos com as bases de dados Iris, Vogal, Câncer e Vidro, respectivamente, assumindo que o tamanho da população inicial era igual a 100. Nessas tabelas, para cada algoritmo, existem duas linhas, sendo a primeira a média dos melhores resultados e a segunda o desvio-padrão associado. Para a base Iris (Tabela 2), o melhor índice de desempenho obtido é praticamente igual para todos, apenas com alguma diferença nas últimas casas decimais. Porém, o APPSO e APDE conseguiram atingir esse índice, utilizando menos avaliações, enquanto o PRoFIPSO e PRoFIDE precisaram utilizar bem mais avaliações.

Tabela 2 - Resultados da base Iris (média e desvio-padrão)

	Média Fitness	*Tamanho população*	*Média de avaliações*
PSO	0,1983569	100	5510
	7,981E-16	0	8003,117
APPSO	0,198357	165,9	1062,7
	0,000955	3,634709	217,3548
PRoFIPSO	0,1983569	785,1	14689,7
	8,868E-16	283,1405	16034,05
DE	0,1983569	100	7350
	2,926E-17	0	4905,836
APDE	0,198357	10	2187,3
	0,0008336	0	537,8077
PRoFIDE	0,1983569	901	36652,3
	2,926E-17	311,3091	30452,3

Já para a base de dados Vogal (Tabela 3), os algoritmos com população variável conseguiram resultados piores, embora muito próximos. O APPSO e APDE mais uma vez foram os algoritmos que utilizaram menos avaliações para encontrar suas soluções.

Tabela 3 - Resultados da base Vogal (média e desvio-padrão)

	Média Fitness	*Tamanho população*	*Média de avaliações*
PSO	0,304212	100	60120
	0,009226	0	20989,67
APPSO	0,31569	10,4	41594,3
	0,015181	0,843274	29178,92
PRoFIPSO	0,300571	1000	96084,3
	0,008399	0	8211,692
DE	0,293855	100	86320
	0,001684	0	9983,631
APDE	0,305049	10	71870,3
	0,014303	0	35030,52
PRoFIDE	0,325316	999,5	85466
	0,007406	0,527046	14835,97

Para a base Câncer (Tabela 4), repetindo os resultados das bases anteriores, os índices de desempenho foram todos muito próximos, embora as metaheurísticas com tamanho da população variável tenham utilizado mais avaliações para alcançar seus resultados.

Tabela 4 - Resultados da base Câncer (média e desvio-padrão)

	Média Fitness	*Tamanho população*	*Média de avaliações*
PSO	0,352618	100	12820
	7,13E-05	0	14371,95
APPSO	0,376185	11,8	27297,2
	0,0301	3,794733	39942,97
PROFiPSO	0,352663	844,2	22878
	0,000109	300,8591	13116,27
DE	0,352618	100	43420
	7,13E-05	0	33685,8
APDE	0,352685	10	9694,7
	0,000116	0	12920,98
PROFiDE	0,352797	999,5	67017,4
	0,000185	0,527046	15997,79

Para a base Vidro (Tabela 5), os resultados das metaheurísticas com tamanho da população variável foram piores que aqueles obtidos por suas versões padrão, e mais uma vez os algoritmos PRoFIPSO e PRoFIDE utilizaram bem mais avaliações.

Tabela 5 - Resultados da base Vidro (média e desvio-padrão)

	Média Fitness	*Tamanho população*	*Média de avaliações*
PSO	0,03619	100	63960
	4,65E-08	0	24342,52
APPSO	0,03989	10	32780,3
	3,87E-08	0	20458,41
PROFiPSO	0,06125	999,3	83157,5
	1,89E-07	0,483046	10678,67
DE	0,01996	100	11640
	1,73E-06	0	8335,76
APDE	0,02931	10	2943,2
	3,51E-06	0,632456	2951,512
PROFiDE	0,03321	999,5	83157,5
	1,07E-07	0,483046	10678,67

Em geral, os resultados obtidos para este estudo de caso sugerem que a utilização do conceito de população variável para problemas de *clustering* não traz ganhos significativos para as metaheurísticas populacionais estudadas, embora o espaço amostral (número de problemas) tenha sido pequeno. Assim, mais estudos são necessários para que possamos obter resultados mais conclusivos. A grande quantidade de variáveis (características das bases *versus* parâmetros de controle dos algoritmos) envolvidas precisa ser estudada de forma mais profunda, pois influencia no resultado final do algoritmo de forma direta.

3.3. Treinamento de redes neurais feedforward

As redes neurais artificiais (RNAs) são sistemas não-lineares que simulam o funcionamento do cérebro humano. Formalmente, uma RNA compreende uma estrutura em grafo direcionado onde cada nó representa um neurônio e realiza alguma tarefa de processamento (Haykin 1999) (Coelho 2004). Esses nós são interligados por arestas (conexões sinápticas) que possuem pesos, cuja função é identificar níveis de excitação ou

inibição entre os neurônios. São os pesos das conexões sinápticas que guardam o "conhecimento" da rede. Esse conhecimento é adquirido através de um processo de aprendizagem que se resume a adaptar os pesos das conexões em consonância com os estímulos recebidos do ambiente (Braga, Carvalho e Ludermir 2007). As redes neurais se caracterizam por sua arquitetura, envolvendo o modelo (tipo) dos neurônios e seu padrão de interconexão (número de camadas), e por sua dinâmica, definindo as suas propriedades operacionais (como aprender, associar e recuperar padrões) (Coelho 2004).

Nesta seção trataremos do treinamento de RNAs *feedforward* com múltiplas camadas a serem projetadas para a resolução de problemas de classificação. Uma instância de RNA aprende a classificar um padrão desconhecido (amostra) em uma dentre várias classes ou categorias, ou seja, trata-se de uma tarefa de aprendizado supervisionado (Mitchell 1997). Na Figura 24, temos a arquitetura típica de uma rede *feedforward*, apresentando uma única camada escondida.

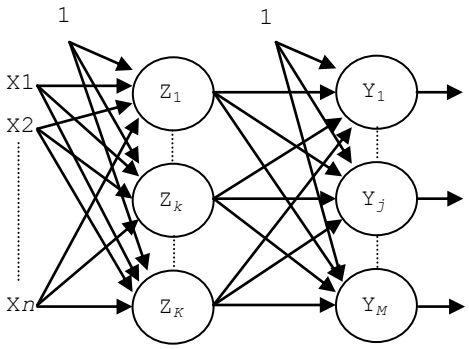

Figura 24 - Arquitetura típica de uma rede *feedforward* com uma única camada escondida.

Na camada de entrada deste tipo de rede, os neurônios $x_i, i = 1,..,n$ representam os elementos de entrada já previamente tratados, normalizados. *Z* representa a camada escondida que possui *K* neurônios. Existem algumas abordagens heurísticas para se dimensionar a quantidade de neurônios da camada escondida; neste trabalho, o tamanho *K* da camada escondida *Z* foi calculado pela média entre a quantidade de características de cada amostra (ou seja, *n*) e a quantidade de neurônios da camada de saída *Y*. Por outro lado, a quantidade de neurônios da camada de saída, ou seja, *M*, foi sempre igual à quantidade de classes relacionadas ao problema de classificação em análise. Na Figura 24, o neurônio com valor fixo "1" representa um neurônio de polarização (*bias*) que atua simplesmente na geração dos

thresholds dos neurônios escondidos e de saída, de acordo com o modelo generalizado do neurônio artificial de McCulloch e Pitts (Haykin 1999).

Dentre as metodologias de treinamento de RNA, o algoritmo supervisionado *backpropagation* pode ser considerado como o mais estudado, implementado e estendido (Coelho 2004). A característica mais saliente desse processo de treinamento é que ele pode ser modelado como um problema típico de otimização numérica sem restrições em que é minimizado um critério de erro (tipicamente, o erro quadrático médio).

Como alternativa ao algoritmo *backpropagation*, o uso de abordagens evolutivas para o treinamento de RNAs vem ganhando bastante espaço nas últimas décadas (Yao 1999). No âmbito deste trabalho, utilizamos os algoritmos propostos com tamanho da população variável para realizar o treinamento de redes *feedforward*. As configurações adotadas para esses algoritmos e os resultados por ele obtidos estão expostos na próxima seção.

3.3.1. Resultados

Para este cenário, utilizamos as mesmas quatro bases de dados do estudo de caso sobre *clustering*. Para podermos validar os resultados, utilizamos o método conhecido como *K-fold cross-validation* (Mitchell 1997). Segundo esse método, para cada base de dados, são inicialmente embaralhadas as amostras e depois separadas em K sub-conjuntos (*folds*) de mesmo tamanho. Tipicamente, $K = 10$. Desta forma, utiliza-se sempre nove *folds* para realizar o treinamento da rede e um *fold* para avaliar a qualidade do modelo encontrado durante o processo de treinamento. No nosso caso, após rotular os *folds* de 1 a 10, procedemos inicialmente o treinamento e o teste da seguinte forma: utilizamos os *folds* de 1 a 9 e executamos os algoritmos heurísticos para o treinamento (cada um sendo executado por 10 vezes), guardando o resultado do melhor indivíduo de cada execução. Em seguida, cada um dos dez melhores modelos (indivíduos) encontrados por cada algoritmo são testados sobre as amostras do *fold* 10 para validar a sua qualidade em termos de generalização. Esse processo foi repetido nove vezes mais, para cada um dos *folds* restantes.

Cada indivíduo da população representa uma configuração de valores para os parâmetros livres (pesos) da rede. Sendo assim, a dimensão do cromossomo de cada indivíduo

será dada por: $(1 + N)K + (1 + K)M$, onde *N* denota o número de atributos do problema em questão, *K* denota o número de neurônios na camada interna e *M* denota número de neurônios da camada de saída (classes do problema). Foram mantidas as mesmas configurações de parâmetros de controle dos algoritmos heurísticos utilizadas nos testes com funções *benchmark*, com exceção do tamanho inicial da população, que foi sempre igual a 100, a dimensão do cromossomo, que depende de cada problema, e o tamanho mínimo da população, que é igual a 10. Já a função de *fitness* adotada foi a taxa de erro de classificação.

Nas Tabela 6 e Tabela 7, temos os resultados obtidos com a base de dados Iris em termos de taxa de acerto. A coluna *A* representa a média dos melhores valores de *fitness* encontrados nas 10 execuções de cada algoritmo de treinamento da rede neural. A coluna *B* representa a média dos melhores valores de *fitness* encontrados durante o teste, utilizando os modelos encontrados na fase de treinamento. As colunas numeradas de 1 a 10 representam o *fold* de teste. Por exemplo, sob a coluna 1, a coluna *A* guarda o resultado médio do treinamento da rede neural utilizando os *folds* 2 a 10; já a coluna *B* guarda o resultado médio encontrado no teste sobre os elementos do *fold* 1.

Tabela 6 - Resultados da base Iris para *folds* de teste de 1 a 5 (taxa de acerto)

		1		*2*		*3*		*4*		*5*	
		A	B	A	B	A	B	A	B	A	B
PSO	Média	100	91,2	100	91,2	99,8	94	99,8	94	99,6	92,6
	D. P.	0	3,83	0	6,26	0,45	0	0,45	0	0,55	3,13
PROFiPSO	Média	100	92,6	100	94	99,8	94	100	94	99,6	94
	D. P.	0	3,13	0	0	0,45	0	0	0	0,55	0
APPSO	Média	100	92,6	100	94	99	91,2	99,4	89,8	99,2	88,6
	D. P.	0	3,13	0	0	0	6,26	0,55	6,26	0,45	8,71
DE	Média	100	94	100	94	100	94	99,8	94	100	94
	D. P.	0	0	0	0	0	0	0,45	0	0	0
PROFiDE	Média	100	94	100	94	100	94	100	94	100	94
	D. P.	0	0	0	0	0	0	0	0	0	0
APDE	Média	81,4	70	79,8	68,6	92,2	83,2	97,8	92,6	99,8	94
	D. P.	17	21,9	16,3	20,1	13,6	16,7	3,83	3,13	0,45	0

Tabela 7 - Resultados da base Iris para *folds* de teste de 6 a 10 (taxa de acerto)

		5		*6*		*7*		*8*		*9*	
		A	B	A	B	A	B	A	B	A	B
PSO	Média	99,8	94	99,6	94	99,8	92,6	99,8	92,6	99,8	94
	D. P.	0,45	0	0,55	0	0,45	3,13	0,45	3,13	0,45	0
PROFiPSO	Média	100	94	99,8	94	100	92,6	100	94	100	94
	D. P.	0	0	0,45	0	0	3,13	0	0	0	0
APPSO	Média	99,2	92,6	99,4	94	99,2	94	99,4	92,6	99,2	94
	D. P.	0,45	3,13	0,55	0	0,45	0	0,55	3,13	0,45	0
DE	Média	99,8	94	100	94	100	94	99,8	94	100	94
	D. P.	0,45	0	0	0	0	0	0,45	0	0	0
PROFiDE	Média	100	94	100	94	100	94	100	94	100	94
	D. P.	0	0	0	0	0	0	0	0	0	0
APDE	Média	98,8	90	80,6	67,2	99,6	94	97,2	88,4	99,4	92,6
	D. P.	1,64	8,94	17,3	18,7	0,55	0	4,02	7,67	0,55	3,13

Tabela 8 - Teste Wilcoxon pareado para a base de dados Iris (*p-value*)

		1	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*
Treinamento	DE x PROFIDE	NA	NA	NA	0.4237	NA	0.4237	NA	NA	0.4237	NA
	DE x APDE	0.0668	0.02315	0.02537	0.232	0.4237	0.232	0.02315	0.177	0.02092	0.0668
	PSO x PROFIPSO	NA	NA	0.4237	0.4237	0.6005	0.177	0.6005	0.177	0.0668	0.6005
	PSO x APPSO	NA	NA	0.003977	0.2703	0.09296	0.2703	0.2703	0.2703	1	0.09296
Teste	DE x PROFIDE	0.177	0.4237	NA	NA	NA	NA	0.4237	NA	0.177	0.4237
	DE x APDE	0.0668	0.02315	0.0707	0.4237	0.4237	0.02315	0.6513	NA	0.177	0.4237
	PSO x PROFIPSO	0.6005	0.4237	NA	NA	0.4237	NA	NA	1	0.4237	NA
	PSO x APPSO	0.6005	0.4237	0.4237	0.1797	0.5186	0.4237	NA	0.4237	1	NA

Para a base Iris, no geral, o PRoFIDE foi quem conseguiu encontrar os melhores resultados e o APDE os piores. O PRoFIDE conseguiu, para todos os *folds* de treinamento, 100% de taxa de acerto e, para todos os *folds* de teste, encontrou sempre 6% de taxa de erro (desvio-padrão nulo). O APPSO apresentou desempenho ruim quando comparado ao PRoFIPSO e ao PSO padrão. O PRoFIPSO foi quem encontrou os melhores resultados dentre todos. Pôde-se observar que o tamanho da população do APPSO e do APDE cai rapidamente logo no começo da execução, fazendo com que o espaço de busca não seja bem explorado. Na

Tabela 8, podemos verificar que, ao se aplicar um teste estatístico não-paramétrico[11], muitas vezes, o desempenho dos novos algoritmos se mostrou idêntico àquele apresentado pelos algoritmos com população de tamanho fixo para essa base de dados.

Tabela 9 - Resultados da base Câncer para *folds* de 1 a 5 (taxa de acerto)

		1		*2*		*3*		*4*		*5*	
		A	B	A	B	A	B	A	B	A	B
PSO	Média	99	92,8	98,8	92,8	98,8	93,6	98,6	93,2	99	93,4
	D. P.	0	1,3	0,45	2,17	0,45	1,34	0,55	1,1	0	1,52
PROFiPSO	Média	99	91,4	99	92,6	99	93,8	98,8	93,8	99	93,4
	D. P.	0	1,34	0	0,55	0	1,1	0,45	1,1	0	1,52
APPSO	Média	98,6	92,7	98,8	92,1	98,5	92,8	98,4	92,7	98,1	92,9
	D. P.	0,52	1,49	0,42	3,07	0,53	1,69	0,52	1,7	0,32	2,18
DE	Média	99	92,6	99	91,7	99	93	99	93,4	99	92,7
	D. P.	0	1,51	0	1,25	0	0,82	0	1,65	0	1,25
PROFiDE	Média	99	91,6	99	92,2	99	94,2	99	93,4	99	93,3
	D. P.	0	0,89	0	0,45	0	1,1	0	1,52	0	1,49
APDE	Média	99	93,3	98,7	92,9	98,4	94,6	98,6	94,4	99	94,9
	D. P.	0	1,7	0,48	1,45	0,52	0,84	0,52	1,51	0	0,74

Tabela 10 - Resultados da base Câncer para *folds* de 6 a 10 (taxa de acerto)

		5		*6*		*7*		*8*		*9*	
		A	B	A	B	A	B	A	B	A	B
PSO	Média	98,8	95,2	98,8	93,8	98,8	93,8	98,4	93,6	98,4	94
	D. P.	0,45	0,45	0,45	1,1	0,45	1,1	0,55	2,19	0,55	1,41
PROFiPSO	Média	98,4	93,2	98,8	94,8	98,4	93,6	98,2	94,6	98,4	94,6
	D. P.	0,55	1,1	0,45	1,1	0,55	1,34	0,45	0,89	0,55	0,89
APPSO	Média	99	93,5	98,5	93,4	98,5	92,9	98,4	93,7	98,7	93,3
	D. P.	0	1,08	0,53	1,17	0,53	2,18	0,52	1,83	0,58	2,89
DE	Média	98,7	94	98,8	91,7	98,4	92,7	98,8	94	98,8	94,5
	D. P.	0,48	1,33	0,42	1,49	0,52	2,21	0,42	1,76	0,46	0,93
PROFiDE	Média	99	94,2	99	94,9	99	94,7	99	94,6	99	94,5
	D. P.	0	1,02	0	0,91	0	1,01	0	1,31	0	1,42
APDE	Média	98,5	93	98,5	94,2	98,6	94,1	98,4	94,3	98,8	94,5
	D. P.	0,53	2,16	0,53	1,03	0,52	1,2	0,52	1,16	0,46	0,24

[11] O teste de Wilcoxon é o equivalente não-paramétrico do teste-t pareado (Hollander e Wolfe 1999). Nos experimentos, adotou-se o software R e α=5% como nível de significância, o que corresponde a um nível de confiança de 95%. Nas Tabela 8, Tabela 11, Tabela 14 e Tabela 17, "NA" indica que o teste não pôde ser aplicado naquela situação, uma vez que as duas amostras de resultados eram idênticas.

Para a base de dados Câncer (Tabela 9 e Tabela 10), observamos, mais uma vez, que os algoritmos APPSO e APDE obtiveram resultados piores, na maioria dos casos, que os demais algoritmos. Mais uma vez, o tamanho da população apresentou queda abrupta logo nas primeiras gerações de cada algoritmo, o que acarretou em pouca exploração do espaço de busca. O melhor desempenho obtido na fase de treinamento foi 99% de taxa de acerto, sendo que essa marca foi atingida para todos os *folds* de treinamento pelo PRoFIDE. O PRoFIPSO também conseguiu bons resultados em termos de taxa de acerto, quando comparado aos algoritmos com tamanho da população fixo; na maioria dos casos, o PRoFIPSO obteve resultados próximos aos do PRoFIDE. A Tabela 11 nos traz a comparação par-a-par obtida com a aplicação do teste não-paramétrico. Como anteriormente, pode-se observar que os níveis de desempenho apresentados para alguns *folds* foram semelhantes entre os algoritmos com ou sem tamanho da população variável; por outro lado, em alguns outros, observamos que o ganho foi significativo.

Tabela 11 - Teste Wilcoxon pareado para a base de dados Câncer (*p-value*)

		1	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*
Treinamento	DE x PROFIDE	NA	NA	NA	NA	NA	0.07672	0.1675	0.005016	0.1675	0.1675
	DE x APDE	NA	0.005016	0.07672	0.03359	NA	0.398	0.1851	0.4076	0.08274	0.1851
	PSO x PROFIPSO	NA	0.3681	0.3681	0.1675	NA	0.3681	0.1675	0.1675	0.03359	0.03359
	PSO x APPSO	0.03359	0.5828	0.06362	0.08274	9.66e-05	0.3681	0.1851	0.1851	0.4076	1
Teste	DE x PROFIDE	0.04241	0.6858	0.05011	0.846	0.001238	0.3480	0.0008153	0.1429	0.9313	1
	DE x APDE	0.3994	0.06146	0.002295	0.1864	0.001238	0.3480	0.0008153	0.1429	0.9313	1
	PSO x PROFIPSO	0.1231	1	0.8174	0.4028	1	0.02384	0.2040	0.8174	0.5186	0.5637
	PSO x APPSO	1	1	0.3003	0.3329	0.753	0.01329	0.4969	0.599	1	0.8957

Os resultados encontrados com a base de dados Vidro evidenciou um maior equilíbrio entre os algoritmos. Todos os algoritmos conseguiram um melhor desempenho para algum *fold* de teste específico (ver Tabela 12 e Tabela 13). Mesmo os algoritmos APPSO e APDE conseguiram resultados bastante próximos aos dos outros algoritmos, fato que não tinha ocorrido para as bases de dados previamente discutidas. Em geral, as taxas de erro na fase de testes foram altas, passando dos 60% em várias ocasiões. Mais uma vez, os algoritmos PRoFIPSO e PRoFIDE foram os que utilizaram a maior quantidade de avaliações para se

alcançar o melhor valor de *fitness*, ao passo que o tamanho da população no momento em que o melhor indivíduo foi encontrado estava sempre no limite máximo (1.000 indivíduos). A semelhança entre os algoritmos ficou bem evidente também quando observamos os resultados encontrados com o Teste Wilcoxon (Tabela 14).

Tabela 12 - Resultados da base Vidro para *folds* de 1 a 5 (taxa de acerto)

		1		*2*		*3*		*4*		*5*	
		A	B	A	B	A	B	A	B	A	B
PSO	Média	57	39,1	59,4	35,4	57,4	40,1	58,9	36,7	57,1	32,5
	D. P.	2,05	10,49	2,757	6,022	1,955	7,795	2,51	4,95	1,97	7,03
PROFiPSO	Média	58,6	29,2	57,7	36,5	56,6	37,7	61,4	36,8	59	43,6
	D. P.	1,65	13,97	1,636	8,462	1,174	11,15	1,82	11,7	1,41	11,2
APPSO	Média	58,4	34	58,4	29	57,8	36	54,4	37,8	55,8	45,4
	D. P.	4,67	10,51	3,362	10,51	2,49	11,85	1,82	6,1	1,3	11,7
DE	Média	57,6	33,5	57,6	32,1	56,7	37	57	36,3	57,2	40,9
	D. P.	1,51	8,343	1,713	9,515	1,252	9,615	1,7	11,3	2,39	6,71
PROFiDE	Média	57,2	27,4	57,2	41,4	58,2	25,4	57,6	33,2	57,8	25,4
	D. P.	1,1	9,864	0,837	7,057	1,304	15,49	0,55	13	1,1	10,9
APDE	Média	55	35,8	56,2	47,4	55,8	41,6	53,6	31,8	57,6	36,8
	D. P.	1	5,357	3,271	12,4	0,837	13,45	5,68	14,8	1,52	12,6

Tabela 13 - Resultados da base Vidro para *folds* de 6 a 10 (taxa de acerto)

		5		*6*		*7*		*8*		*9*	
		A	B	A	B	A	B	A	B	A	B
PSO	Média	58,6	29,7	59,2	32,6	59	31,2	58,4	33,8	58,6	37,7
	D. P.	2,59	6,516	2,53	7,575	1,826	9,09	2,22	11,2	2,17	15
PROFiPSO	Média	60,4	44,4	58,8	44	59,6	43,2	61,8	36,6	61,6	40,7
	D. P.	1,52	18,04	2,588	2,236	3,05	12,87	2,59	9,91	2,99	10,8
APPSO	Média	56,7	41	56,8	31	57	34	55,2	28	56	44,2
	D. P.	1,44	10,65	1,789	12,37	2	9,899	1,92	4,18	2,83	3,83
DE	Média	56,8	33,5	57,6	35,7	55,8	29,6	56,6	32	56,5	39,7
	D. P.	0,92	11,72	1,35	8,782	1,476	16,56	1,35	11,5	1,43	9,31
PROFiDE	Média	57,6	34	57	27,2	57,5	24,5	57,7	31,9	57,9	34,7
	D. P.	1,14	6,124	1,225	11,12	0,707	6,364	0,16	7,02	1,01	10,7
APDE	Média	55,2	40,6	56	37,8	51,6	36,8	56	37,8	53	32
	D. P.	1,79	9,864	2,915	13,52	7,162	13,59	2,74	6,1	6,89	14,3

Finalmente, para a base de dados Vogal (Tabela 15 e Tabela 16), os algoritmos DE e PRoFIDE foram os que prevaleceram, sendo que, no geral, o PRoFIDE foi aquele com o melhor desempenho médio em termos de taxa de acerto. O PRoFIPSO não conseguiu resultados melhores que o DE, mas conseguiu superar o PSO. Mais uma vez, o APPSO e o APDE não conseguiram obter bons índices de desempenho. Para os problemas de classificação utilizando redes neurais artificiais, a fase de exploração do espaço de busca se mostra muito importante, o que justifica o fato dos algoritmos PRoFIPSO e PRoFIDE estarem encontrando sempre melhores resultados. O PRoFIPSO foi o algoritmo que alcançou os resultados mais significativos, como podemos observar na Tabela 17.

Tabela 14 - Teste Wilcoxon pareado para a base de dados Vidro (*p-value*)

		1	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*
Treinamento	DE x PROFIDE	0.7965	1	0.07686	0.5298	0.4939	0.1998	0.3791	0.1438	0.6091	0.3157
	DE x APDE	0.006848	0.3120	0.1290	0.1473	0.6642	0.09232	0.2909	0.1438	0.6091	0.3157
	PSO x PROFIPSO	0.05865	0.1261	0.4735	0.09584	0.07939	0.1681	0.665	0.852	0.03649	0.137
	PSO x APPSO	0.4188	0.5374	0.8354	0.005383	0.2046	0.1204	1	0.07883	0.02612	0.137
Teste	DE x PROFIDE	0.2361	0.09042	0.1327	0.4005	0.01844	0.5746	0.2037	0.4604	0.4227	0.2119
	DE x APDE	0.8012	0.09042	0.7504	0.6636	0.2855	0.233	0.8025	0.4604	0.4911	0.3522
	PSO x PROFIPSO	0.09941	0.497	0.7005	0.8457	0.03246	0.1173	0.001961	0.1323	0.8517	0.5381
	PSO x APPSO	0.3446	0.1958	0.3809	0.8412	0.01574	0.4768	0.0001512	0.8457	0.1301	0.2165

Tabela 15 - Resultados da base Vogal para *folds* de 1 a 5 (taxa de acerto)

		1		*2*		*3*		*4*		*5*	
		A	B	A	B	A	B	A	B	A	B
PSO	Média	74,3	66,2	73,5	65,2	76,1	66,9	67	58,5	77,9	66,8
	D. P.	9,43	7,67	8,61	7,67	9,06	9,6	10,6	9,97	4,38	5,81
PROFiPSO	Média	78,4	69,5	78,8	68,5	76,5	67,1	76,6	64,3	79,4	68,6
	D. P.	7,47	7,88	7	6,28	11	13,8	8,58	7,65	5,78	6,52
APPSO	Média	68	59,6	70,5	60,6	64,9	55,2	69	59,8	71,5	63,8
	D. P.	7,07	9,64	9,24	8,5	10,6	9,03	9,49	10,6	7,91	9,9
DE	Média	80,5	70	82	68,8	79	72,3	80,8	73	80,5	73
	D. P.	0,84	2,97	0,82	5,68	1,67	2,8	0,98	1,22	1,05	1,67
PROFiDE	Média	81,8	73,3	81,7	73,8	81,6	74	81,3	74,3	82	75,2
	D. P.	0,5	3,1	0,95	2,91	1,76	2,91	0,31	2	0,89	1,94
APDE	Média	53,6	47,9	52,4	46	56,6	45	50,4	42,6	59,4	49,9
	D. P.	8,34	9,32	6,21	5,66	7,9	7,02	7,66	8,38	7,13	7,2

Tabela 16 - Resultados da base Vogal para *folds* de 6 a 10 (taxa de acerto)

		5		*6*		*7*		*8*		*9*	
		A	B	A	B	A	B	A	B	A	B
PSO	Média	78,1	68,2	71,2	62,9	76,5	67,7	70,9	63,8	77,1	60,7
	D. P.	8,39	9,2	8,24	8,74	8,02	9,43	9,98	9,09	8,99	10,2
PROFiPSO	Média	79,1	69,8	78,3	69,1	77,3	68,6	77,1	66,5	79,9	67,9
	D. P.	7,21	8,1	7,21	7,9	7,9	8,72	8,01	8,3	7,18	7,92
APPSO	Média	63,9	55,5	70,7	63,1	71,1	59,4	70	56,9	70	55,9
	D. P.	8,2	11	9,66	7,37	7,91	11,8	8,11	10,2	7,31	9,02
DE	Média	81,7	72,8	79	72,4	80,2	71,7	80,3	73	78,5	72,5
	D. P.	0,82	1,47	1,63	2,94	1,83	3,27	0,82	2,61	0,71	2,12
PROFiDE	Média	81,7	73,3	81,8	72	80,9	71,9	81,7	74,2	81,1	71,6
	D. P.	1,03	2,42	2,14	3,52	0,69	3,85	0,82	2,23	0,69	3,15
APDE	Média	55,8	50,6	59,5	50,6	56,9	49,9	58,8	50,8	58,4	50
	D. P.	6,69	8,08	6,99	5,37	6,02	5,99	6,91	7	6,34	6,58

Tabela 17 - Teste Wilcoxon pareado para a base de dados Vogal (*p-value*)

		1	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*
Treinamento	DE x PROFIDE	0.01065	1	0.008733	0.1519	0.03939	1	0.01313	0.7097	0.02568	0.2258
	DE x APDE	0.005349	0.005597	0.003275	0.004566	0.007686	0.005494	0.005597	0.007827	0.007408	0.007827
	PSO x PROFIPSO	0.5428	0.1839	0.8792	0.03058	0.3703	0.2246	0.008901	0.002278	0.00154	0.001521
	PSO x APPSO	0.07434	0.4244	0.02308	0.5935	0.03670	0.001612	1	0.8201	0.7048	0.2718
Teste	DE x PROFIDE	0.04316	0.05084	0.1566	0.01184	0.04197	0.4657	0.9425	1	0.6273	0.8268
	DE x APDE	0.005681	0.005514	0.007827	0.01167	0.007546	0.00797	0.005681	0.007827	0.004329	0.007
	PSO x PROFIPSO	0.04401	0.01582	0.01023	0.003312	0.02422	0.4779	0.01614	0.1058	0.1252	0.623
	PSO x APPSO	0.2714	0.1707	0.01701	0.8488	0.4431	0.01709	0.8201	0.2106	0.1111	0.01880

Na Tabela 18, apresenta-se um resumo dos experimentos contendo o resultado médio obtido pelos algoritmos para todos os *folds* das quatro bases de dados utilizadas neste estudo de caso. No geral, o algoritmo heurístico PRoFIDE foi quem obteve os melhores resultados. Apenas para a base de dados Vidro, o PRoFIPSO o superou. O DE foi quem, dos algoritmos com tamanho da população fixo, conseguiu melhores resultados. O APPSO e o APDE encontraram sempre resultados próximos ou piores aos demais algoritmos.

Tabela 18 – Sumário dos resultados obtidos para as quatro base de dados (taxa de acerto)

		Iris		*Câncer*		*Vidro*		*Vogal*	
		A	B	A	B	A	B	A	B
PSO	Média	99,8	93,02	98,74	93,62	58,36	34,88	74,26	64,69
	D. P.	0,378	1,949	0,388	1,367	2,259	8,562	8,571	8,732
PROFiPSO	Média	99,92	93,72	98,7	93,58	59,55	39,27	78,14	67,99
	D. P.	0,144	0,626	0,298	1,092	2,042	11,04	7,739	8,308
APPSO	Média	99,4	92,34	98,55	93	56,65	36,04	68,96	58,98
	D. P.	0,343	3,062	0,445	1,929	2,362	9,159	8,549	9,708
DE	Média	99,94	94	98,85	93,03	56,94	35,03	80,25	71,95
	D. P.	0,134	0	0,231	1,42	1,509	10,34	1,117	2,675
PROFiDE	Média	100	94	99	93,76	57,57	30,51	81,54	73,34
	D. P.	0	0	0	1,112	0,912	9,765	0,979	2,804
APDE	Média	92,66	84,06	98,65	94,02	55	37,84	56,18	48,33
	D. P.	7,514	10,04	0,406	1,203	3,38	11,6	7,019	7,06

3.4. Conclusão

Neste capítulo, foram discutidos três estudos de caso conduzidos com o intuito de avaliar o desempenho dos modelos heurísticos propostos nesta dissertação. Os resultados experimentais obtidos nos mostram que, para as funções *benchmark*, o PRoFIPSO e APPSO foram os algoritmos que alcançaram os melhores índices de desempenho em termos de eficácia (ou seja, capacidade de localizar soluções mais satisfatórias). Já para o problema de descoberta de protótipos em agrupamento de dados, os algoritmos com população de tamanho variável não conseguiram apresentar ganhos de eficácia que compensem a perda em termos de eficiência. Por outro lado, no treinamento de rede neurais artificiais para classificação, o PRoFIDE foi o algoritmo que produziu na maioria das vezes o melhor índice de desempenho.

O comportamento evolutivo do tamanho da população também se mostrou diferente nos três estudos de caso. Para as funções *benchmark*, os modelos baseados no PRoFIGA obtiveram um grande crescimento do tamanho da população no começo, atingindo sempre o tamanho máximo, e permanecendo assim durante grande parte das gerações, diminuindo próximo ao fim do algoritmo. Já os modelos com a variação do tamanho da população baseados no APGA encontraram sempre um tamanho em torno do qual ficou variando. Em relação ao problema de descoberta de protótipos em agrupamento de dados, em se tratando

dos modelos baseados no PRoFIGA, o tamanho da população também chegou logo ao máximo continuando assim até o final. Já para os modelos baseados no APGA, o tamanho da população tendeu a cair vagarosamente sempre, terminando as execuções com o tamanho próximo do mínimo. Por último, em relação ao treinamento das redes neurais, os modelos baseados no PRoFIGA, mais uma vez, quase sempre atingiram o limite máximo do tamanho da população logo de início, continuando assim por grande parte da execução e diminuindo mais para o fim. Já os modelos baseados no APGA geralmente tiveram uma queda brusca logo no início da execução, permanecendo em torno do tamanho mínimo até o final. No geral, os resultados apontam que, para cada tipo de problema estudado, um tipo de algoritmo com população de tamanho variável se apresenta como o mais apropriado, não existindo um que seja o melhor para todos os casos.

Com os resultados exibidos neste capítulo, fica clara também a importância de uma calibração apropriada dos parâmetros de controle dos algoritmos estudados para cada tipo de problema. Pôde-se observar que, nem sempre a configuração utilizada para um modelo em um tipo de problema alcança bons resultados em outros tipos de problemas, algo que já era esperado. Isso evidencia a necessidade de se fazer uma calibração mais refinada dos novos parâmetros de controle introduzidos em substituição ao tamanho da população. Por outro lado, os resultados corroboram a perspectiva de que, ao se adotar o conceito de população com tamanho variável, pode-se produzir novos algoritmos adaptativos cujos comportamentos de busca são bastante diferentes daqueles já existentes, obtendo-se, eventualmente, ganhos de desempenho consideráveis em termos de eficácia.

4. Considerações finais

A criação de quatro novas metaheurísticas populacionais de tamanho variável para resolução de problemas de busca e otimização se constituiu no propósito e principal contribuição deste trabalho.

Três cenários de otimização numérica foram utilizados para validar a qualidade dos algoritmos em termos de eficiência e eficácia. A utilização dos algoritmos para a otimização de funções *benchmark* e para o treinamento de redes neurais artificiais apresentou bons resultados em termos de eficácia em se comparando com algoritmos de população de tamanho fixo. Já os testes realizados com o problema de descoberta de protótipos para agrupamentos de dados não nos evidenciaram ganhos para os novos algoritmos, já que estes conseguiram no máximo repetir os mesmos índices de desempenho obtidos pelos algoritmos com tamanho da população fixo, utilizando, entretanto, mais avaliações. Acreditamos que esta tarefa, dadas as suas características peculiares, necessite de uma calibração mais bem otimizada dos parâmetros de controle dos algoritmos propostos. Aquela que adotamos foi a mesma utilizada nos experimentos com as funções *benchmark*. Esse estudo já se encontra em andamento.

Junto com estes parâmetros calibrados com as funções *benchmark*, encontra-se um parâmetro que, quando varia, também impacta no resultado final. É o tamanho inicial da população. Não só o comportamento deste parâmetro, como também, daqueles inseridos pelos novos modelos, precisam ser mais bem estudados, para que se possa saber o real ganho, em termos de eficiência e eficácia, da utilização destes modelos com tamanho da população variável, no lugar de suas versões com tamanho da população fixo.

Uma das linhas de investigação futura é a de avaliar o desempenho dos algoritmos heurísticos com população de tamanho variável no contexto de otimização discreta. Para

tanto, será necessário utilizar modelos estendidos dos algoritmos PSO e DE, já que estes foram propostos inicialmente para o domínio de otimização numérica. Um exemplo de modelo de PSO adaptado para problemas combinatórios é aquele proposto recentemente por Jarboui et al. (2007).

Uma outra linha de atuação seria a de investigar outros modelos de AEs com tamanho da população variável no contexto do PSO e DE, tais como os algoritmos RVPS e PLGA. Por outro lado, também seria possível realizar esse estudo acerca do conceito de tamanho de população variável com outras metaheurísticas populacionais, tais como aquela conhecida por Otimização por Colônia de Formigas (ACO[12]), proposta no contexto de otimização discreta mas que vem sendo adaptada mais recentemente para o contexto de otimização numérica (Bilchev e Parmee 1995).

Finalmente, vale mencionar uma outra linha de investigação que já se encontra em curso e que envolve a extensão dos algoritmos propostos neste trabalho para operarem em um ambiente paralelo. Nesse contexto, Cantú-Paz (1998) já conduziu um estudo sobre os diferentes métodos de paralelização de AGs. Em (Minetti e Alfonso 2005) é utilizado o modelo paralelo de ilhas em que duas ou mais sub-populações (ilhas) de AGs com tamanho de população variável evoluem concorrentemente e interagem entre si, com certa freqüência, para realizar trocas de indivíduos. No trabalho de Minetti e Alfonso, foram realizados testes somente com o GAVaPS e PRoFIGA, em que todas as ilhas continham o mesmo algoritmo. Em vez de trabalharmos com esse modelo homogêneo, nosso intuito vem sendo o de explorar dois ou mais algoritmos heurísticos com populações de tamanho variável (por exemplo, uma ilha com APDE e outra ilha com PRoFIDE, ou mesmo uma ilha com APDE e outra com PRoFIPSO) para cooperarem entre si de forma paralela. Estamos denominando essa estratégia de heterogênea. Acreditamos que resultados satisfatórios podem ser obtidos com esse modelo heterogêneo para problemas de otimização mais complexos, uma vez que ele possibilita combinar as características de exploração e refinamento específicas de cada um dos modelos estudados.

[12] Do inglês: *Ant Colony Optimization*.

Bibliografia

Arabas, J., Michalewicz, Z. e Mulawka, J. GAVaPS – A genetic algorithm with varying population size. *Proceedings of the First IEEE Conference on Evolutionary Computation*, vol. 1, pp. 73-78, Piscataway, IEEE Press, 1994.

Asuncion, A. e Newman, D. UCI Machine Learning Repository. School of Information and Computer Science, Universitu of California, Irvine, 2007. http://www.ics.uci.edu/~mlearn/MLRepository.html (acessado em junho de 2008).

Bäck, T. *Evolutionary Algorithms Theory and Practice: Evolution Strategies, Evolutionary Pragramming, Genetic Algorithms.* Oxford University Press, 1996.

Bäck, T., Eiben, A. e Vaart, N. An empirical study on GAs without parameters. *Proceedings of Parallel Problem Solving from Nature, PPSN VI, Lecture Notes in Computer Science*, vol. 1917, pp. 315-324, Springer, 2000.

Bäck, T., Fogel, D. e Michalewicz, Z. *Evolutionary Computation 1: Basic Algorithms and Operators.* IOP Press, 2000.

Bäck, T., Fogel, D. e Michalewicz, Z. *Evolutionary Computation 2: Advanced Algorithms and Operations.* IOP Press, 2000.

Bäck, T., Fogel, D. e Michalewicz, Z. *Handbook of Evolutionary Algorithms.* Oxford Press, 1997.

Berkhin, Pavel. *Survey of Clustering Data Mining Techniques.* San Jose, USA: Springer, 2002.

Bilchev, G. e Parmee, I. The ant colony metaphor for searching continuous design spaces. *Lecture Notes in Computer Science*, vol. 993, pp. 25-39, Springer, 1995.

Blum, C. e Roli, A. Metaheuristics in Combinatorial Optimization: Overview and Conceptual Comparison. *ACM Computing Surveys*, vol. 35, pp. 268-308, 2003.

Braga, A., Carvalho, A. e Ludermir, T. *Redes Neurais Artificiais.* 2ª ed. Rio de Janeiro, RJ: LTC, 2007.

Burke, E. e Kendall, G. *Search Methodologies: Introductory Tutorials in Optimization and Decision Support Techniques.* Springer, 2005.

Calinski, T. e Harabasz, J. A dendrite method for cluster analysis. *Communications in Statistics*, vol. 3, pp. 1-27, 1974.

Cantú-Paz, E. A survey of parallel genetic algorithms. *Calculateurs Parallèles, Réseaux et Systèmes Répartis*, vol. 10, pp. 141-171, 1998.

Carlisle, A e Dozier, G. An off-the-shelf PSO. *Proceedings of the Particle Swarm Optimization Workshop*, vol. 1, pp. 1–6, 2001.

Clerc, M. e Kennedy, J. The particle swarm - explosion, stability and convergence in a multidimensional complex space. *IEEE Transactions on Evolutionary Computation*, vol 1, pp. 58-73, 2002.

Coelho, A. L. V. e Oliveira, D. G. Dynamically tuning the population size in particle swarm optimization. *Proceedings of ACM Symposium on Applied Computing (SAC)*, vol. 3, pp. 1782-1787, 2008.

Coelho, A. L. V. Evolução, simbiose e hibridismo aplicados à engenharia de sistemas inteligentes modulares: investigações em redes neurais, comitês de máquinas e sistemas multiagentes. *Tese de doutorado, Faculdade de Engenharia Elétrica e Computação, UNICAMP*, 2004.

Costa, J., Tavares, R. e Rosa, A. An experimental study on dynamic random variation of population size. *Proceedings of IEEE Systems, Man, and Cybernetics Conference*, vol. 1, pp. 607-612, Tokyo: IEEE Press, 1999.

Darwin, C. *On the Origin of Species by Means of Natural Selection or Preservation of Favoured Races in the Struggle fo Life.* London: Murray, 1859.

DeJong, K. An analysis of the behavior of a class of genetic adaptive systems. *Tese de doutorado, University of Michigan*, 1975.

Digalakis, J. e Margaritis, K. An experimental study of benchmarking functions for genetic algorithms. *International Journal of Computer Mathematics*, vol. 5, pp. 403-416, 2002.

Eberhart, R. e Kennedy, J. A new optimizer using particle swarm theory. *Proceedings of the Sixth International Symposium on Micromachine and Human Science*, vol. 1, pp. 39-43, Nagoya, Japan, 1995.

Eberhart, R., Simpson, P. e Dobbins, R. *Computational Intelligence PC Tools.* Academic Press, 1996.

Eberhat, R. e Shi, Y. Comparing Inertia weight and constriction factors in particle swarm optimization. *Proceedings of Congress on Evolutionary* Computation, vol. 1, pp. 84-88, San Diego, CA, 2000.

Eiben, A. e Smith, J. *Introduction to Evolutionary Computing.* Springer, 2003.

Eiben, A., Marchiori, E. e Valkó, V. 2004. Evolutionary Algorithms with on-the-fly population size adjustment. *Lecture Notes in Computer Science*, vol. 3242, pp. 41-50, Springer, 2004.

Everitt, B. *Cluster Analysis.* New York: Halsted Press, 1993.

Fernandes, C. e Rosa, A. Self-regulated population size in evolutionary algorithms. *Proceedings of Parallel Problem Solving from Nature*, *Lecture Notes in Computer Science*, vol. 4193, pp. 920-929, Springer, 2006.

Fernandes, C., Tavares, R. e Rosa, A. niGAVaPS – outbreeding in genetic algorithms. *Proceedings of the ACM Symposium on Applied Computing*, vol. 1, pp. 477-482, 2000.

Forgy, E. Cluster analysis of multivariate data: effeciency versus interpretability of classification, *Biometrics*, vol. 21, pp. 768-769, 1965.

Friedman, H. e Rubin, J. On some invariant criterion for grouping data. *Journal of the American Statistical Association*, vol. 62, pp. 1159-1178, 1967.

Glover, F. e Kochenberger, G. *Handbook of Metaheuristics.* The Netherlands: Kluwer Academic Publishers, 2003.

Glover, F. *Future paths for integer programming and links to artificial intelligence*, *Computers and Operations Research*, vol. 13, pp. 533-549, 1986.

Goldberg, D. E. *Genetic Algorithms in Search, Optimization and Machine Learning.* Addison Wesley, 1989.

Goldberg, D. E. Sizing populations for serial and parallel genetic algorithms. *Proceedings of the Third International Conference on Genetic Algorithms and Their Applications*, vol. 1, pp. 70-79, Morgan Kaufmann, 1989.

Grefenstette, J. Optimization of control parameters for genetic algorithms. *IEEE Transactions on Systems, Man and Cybernetics*, vol. 16, pp. 122-128, 1986.

Harik, G. e Lobo, F. A parameter-less genetic algorithm. *Proceedings of the Genetic and Evolutionary Computation Conference*, vol. 1, pp. 258-265, Orlando, 1999.

Haykin, S. *Neural Networks: A Comprehensive Foundation.* New Jersey: Prentice Hall, 1999.

Hiller, F. e Lieberman, G. *Introductions to operation Research.* 7a. ed. McGraw-Hill, 2002.

Holland, J. *Adaptation in Natural and Artificial Systems.* Ann Arbor: MIT Press, 1975.

Hollander, M. e Wolfe, D. A. *Nonparametric Statistical Methods*. 2a. ed., Wiley, 1999.

Hu, X., Shi, Y. e Eberhart, R. Recent advances in particle swarm. *Proceedings of IEEE Congress on Evolutionary Computation*, vol. 1, pp. 90-97, Portland, 2004.

Huang, T. e Mohan, A. Micro-particle swarm optimizer for solving high dimensional optimization problems. *Applied Mathematics and Computation*, vol. 181, pp. 1148-1154, 2006.

Jain, A., Murty, M. e Flynn, P. Data clustering: A review. *ACM Computing Surveys*, vol. 31, pp. 264-323, 1999.

Jarboui, B., Cheikh, M., Siarry, P. e Rebai A. Combinatorial particle swarm optimization (CPSO) for partitional clustering problem. *Applied Mathematics and Computation*, vol. 192, pp. 337-345, 2007.

Kennedy, J. The behavior of particles. *Proceedings of Evolutionary Programming VII. Lecture Notes in Computer Science*, vol 1447, pp. 581-590, Springer ,1998.

Kennedy, J. e Eberhart, R. Particle swarm optimization. *Proceedings IEEE International Conference on Neural Networks*, vol. 12, pp. 1942-1948, Piscataway: IEEE Press, 1995.

Kennedy, J. e Eberhart, R. *Swarm Intelligence.* Morgan Kaufmann Publishers, 2001.

Kirkpatrick, S., Gelatt, C. e Vecchi, M. *Optimization by simulated annealing*, vol. 220, pp. 671-680, 1983.

Koumousis, V. e Katsaras, C. A saw-tooth genetic algorithm combining the effects of variable population size and reinitialization to enhance performance. *IEEE Transactions on Evolutionary Computation*, vol. 10, pp. 19-28, 2006.

Linden, R. *Algoritmos Genéticos.* Rio de Janeiro: Brasport, 2006.

Liping, Z., Huanjun, Y. e Shangxu, H. A new approach to improve particle swarm optimization. *Proceedings of Genetic and Evolutionary Computation*, vol. 1, pp. 134-139, Springer, 2003.

Lobo, F. e Lima, C. A review of adaptive population sizing schemes in genetic algorithms. *Proceedings of the 2005 Workshop on Parameter Setting in Genetic and Evolutionary Algorithms*, vol. 1, pp. 228-234, 2005.

Marriott, F. Practical problems in a method of cluster analysis. *Biometrics*, vol. 27, pp. 501-514, 1971.

Michalewicz, Z. *Genetic Algorithms + Data Structures = Evolution Programs.* 3ª. ed., Springer, 1996.

Minetti, G. and Alfonso, H. Variable size population in parallel evolutionary algorithms. *Proceedings of the 5th International Conference on Intelligent Systems Design and Applications*, vol. 1, pp. 350-355, IEEE Computer Society Press, 2005.

Mitchell, M. *Machine Learning.* McGraw-Hill, 1997.

Paterlini, S. e Krink, T. Differential evolution and particle swarm optimisation in partitional clustering, vol. 50, pp. 1220-1247, 2006.

Price, K., Storn, R. e Lampinen, J. *Differential Evolution: A Practical Approach to Global Optimization.* Springer, 2005.

Rezende, S. *Sistemas Inteligentes Fundamentos e Aplicações.* Barueri - SP: Manole, 2005.

Shi, Y. e Eberhart, R. A modified particle swarm optimizer. *Proceedings of the 1998 IEEE Conference on Evolutionary Computation*, vol. 1, pp. 4-9, 1998.

Shi, Y. e Eberhart, R. Experimental study of particle swarm optimization. Proceedings of the World Multiconference on Systemics, Cybernetics, and Informatics, vol. 3, pp. 1950-1958, 2000.

Shi, Y. e Eberhart, R. Parameter selection in particle swarm optimization. *Evolutionary Programming VII. Lecture Notes in Computer Science*, vol. 1, pp. 611-616, Springer, 1998.

Shi, Y. e Eberhat, R. Fuzzy adaptive particle swarm optimization. *Proceedings of the 2001 Congress on Evolutionary Computation*, vol. 1, pp. 101-106, 2001.

Smith, R. e Smuda, E. Adaptively resizing populations: Algorithm, analysis, and first results. *Complex Systems*, vol. 1, pp. 47-72, 1995.

Spinosa, E. e Pozo, A. Controlling the population size in genetic programming. *Proceedings of Simpósio Brasileiro de Inteligência Artificial. Lecture Notes in Artificial Intelligence*, vol. 2507, pp. 345-354, Springer, 2002.

Storn, R. e Price, K. Differential Evolution - A simple and efficient adaptive scheme for global optimization over continuous spaces. *Journal of Global Optimization*, vol. 11, pp. 341-359, 1997.

Teo, J. Differential evolution with self-adaptive populations. *Proceedings og Knowledge-Based Intelligent Information and Engineering Systems, Lecture Notes in Computer Science*, vol. 3681, pp. 1284-1290, Springer, 2005.

Valkó, V. A. *Self-calibration in evolutionary algorithms: Adaptive population size.* Dissertação de mestrado, Free University Amsterdam, 2003.

Yao, X. Evolving artificial neural networks. *Proceedings of IEEE,* vol. 87, pp. 1423-1447, 1999.

Yoshida, H., Kawata, K., Fukuyama, Y., Nakanishi, Y. A particle swarm optimization for reactive power and voltage control considering voltage stability. *Proceedings of the International Conference Intelligent System Application to Power Systems,* vol. 1, pp. 177-121, Rio de Janeiro, 1999.

Apêndice I – Resultados comparativos utilizando funções *benchmark*

As seguintes tabelas guardam os resultados obtidos nos experimentos realizados sobre funções de otimização numérica com os modelos metaheurísticos de população variável. As colunas dessas tabelas estão nomeadas como *A*, *B*, *C* e *P*, que denotam, respectivamente, a média dos melhores valores de *fitness* encontrados, a média das quantidades de avaliações necessárias para se encontrar o melhor valor de *fitness*, a média do tamanho da população quando se encontrou o melhor valor de *fitness*, e o tamanho da população inicial. Para os algoritmos sem tamanho da população variável, a coluna *C* não se faz necessária.

Resultados da função Ackley com dimensão 10.

			Padrão		*APGA, APPSO ou APDE*			*PRoFIGA, PRoFIPSO ou PROFIDE*		
	P		A	B	A	C	B	A	C	B
GA	20	Média	0,0919	32299	0,2844	16	38078	0,6182	15	23359
		Des. P.	0,0645	10925	0,2369	3	25322	0,7048	0	9762
	50	Média	0,4643	45084	0,6284	16	31602	0,7194	22	21154
		Des. P.	0,4468	19220	1,2000	5	14204	0,5312	5	8588
	100	Média	0,9840	58681	0,3840	15	33220	0,8418	98	23070
		Des. P.	0,8301	31928	0,3289	0	15111	0,7618	27	10211
PSO	20	Média	0,1540	13240	0,0000	46	22365	0,0000	20	96952
		Des. P.	0,4065	1882	0,0000	10	1131	0,0000	4	3635
	50	Média	0,0000	20167	0,0000	45	24725	0,0000	25	101255
		Des. P.	0,0000	492	0,0000	9	1137	0,0000	5	3649
	100	Média	0,0000	28360	0,1097	44	28098	0,0000	39	101666
		Des. P.	0,0000	820	0,4251	9	1588	0,0000	12	3948
DE BIN	20	Média	1,4281	14835	0,0005	30	19115	1,8317	486	117905
		Des. P.	1,3149	2305	0,0008	15	1593	1,0500	2	34986
	50	Média	0,6687	35610	0,0005	34	21369	2,9560	494	89114
		Des. P.	0,5858	3361	0,0019	13	2376	2,2697	19	44276
	100	Média	1,1051	53593	0,0284	36	22705	1,9929	485	105755
		Des. P.	0,8133	17032	0,1096	12	1311	1,0469	27	34205
DE EXP1	20	Média	2,0110	11587	0,0032	29	19785	1,6328	426	135374
		Des. P.	1,6627	1999	0,0085	13	1742	1,1050	167	43844
	50	Média	0,6439	25450	0,0005	38	21732	2,4712	437	107816
		Des. P.	0,6460	833	0,0017	17	1405	1,7211	166	56894
	100	Média	0,4295	40720	0,0003	29	24647	2,7248	431	109272
		Des. P.	0,2986	1869	0,0012	14	1458	2,1105	157	63286
DE EXP2	20	Média	2,3560	19765	0,0868	22	18299	3,8973	480	92492
		Des. P.	2,0977	736	0,3257	11	2163	1,2433	35	31278
	50	Média	1,2396	37623	0,0021	19	19007	4,3500	480	88732
		Des. P.	1,1508	2215	0,0027	7	2182	1,9924	43	41177
	100	Média	0,4428	64433	0,0108	18	21443	3,8277	474	99195
		Des. P.	0,4172	1941	0,0312	7	2161	1,3003	65	34355

Resultados da função Ackley com dimensão 20.

			Padrão		*APGA, APPSO ou APDE*			*PRoFIGA, PRoFIPSO ou PROFIDE*		
	P		A	B	A	C	B	A	C	B
GA	20	Média	0,2515	48558	0,1578	15	49222	0,2916	15	33565
		Des. P.	0,1982	22688	0,1000	0	22065	0,3792	0	14351
	50	Média	0,2986	84233	0,2121	15	46757	0,3748	23	37563
		Des. P.	0,3204	18127	0,2491	0	20036	0,3808	5	15042
	100	Média	0,4637	129619	0,1498	15	49921	0,4107	102	41491
		Des. P.	0,2739	24603	0,1098	0	14414	0,2307	28	13838
PSO	20	Média	1,9717	17171	0,5417	76	45746	0,0000	21	142597
		Des. P.	1,6271	2591	0,8578	12	13196	0,0000	2	5408
	50	Média	0,1898	27273	0,4043	75	44729	0,0000	28	142884
		Des. P.	0,5009	1712	0,7073	11	11594	0,0000	9	5753
	100	Média	0,0000	40240	0,7710	72	45111	0,0000	35	141625
		Des. P.	0,0000	1158	0,8104	9	8453	0,0000	16	6315
DE BIN	20	Média	1,8814	38689	0,1852	49	27014	4,8665	476	141982
		Des. P.	0,9573	14013	0,6958	13	2355	1,6176	42	34572
	50	Média	2,2031	41070	0,0029	43	29011	5,0550	473	144494
		Des. P.	1,1486	21647	0,0088	7	1487	2,5320	53	52954
	100	Média	2,9155	38187	0,0014	42	30331	4,2813	465	158363
		Des. P.	0,4031	15925	0,0053	8	1417	1,5054	61	46024
DE EXP1	20	Média	1,4047	20777	0,0805	42	31810	4,3350	406	188963
		Des. P.	1,3103	837	0,1916	9	1958	1,9254	139	64843
	50	Média	0,4916	40540	0,0150	40	34838	4,7019	415	178058
		Des. P.	0,4297	1825	0,0440	10	2417	2,2214	162	72219
	100	Média	0,6851	69620	0,0055	39	37424	7,2976	462	124679
		Des. P.	0,4860	2570	0,0118	8	2052	3,7087	103	62922
DE EXP2	20	Média	1,2853	31624	0,5973	30	27939	8,1595	453	128455
		Des. P.	1,1000	1658	0,8081	8	4119	2,3219	60	43554
	50	Média	0,9727	64430	0,4005	33	27446	9,1558	472	115989
		Des. P.	0,7134	3419	0,6433	6	2412	3,2482	59	54202
	100	Média	0,4396	113847	0,1443	32	30469	8,3105	463	130003
		Des. P.	0,3774	4784	0,2636	7	2279	2,2387	90	50684

Resultados da função Ackley com dimensão 50.

			Padrão		*APGA, APPSO ou APDE*			*PRoFIGA, PRoFIPSO ou PROFIDE*		
	P		A	B	A	C	B	A	C	B
GA	20	Média	0,3154	95936	0,0347	15	120190	0,1740	15	87341
		Des. P.	0,1529	22910	0,0250	0	25358	0,0513	0	23681
	50	Média	0,3495	204866	0,0394	15	113348	0,3160	25	70767
		Des. P.	0,2119	37080	0,0312	0	33335	0,1286	2	16450
	100	Média	0,3359	361687	0,0461	15	110475	0,4790	100	79336
		Des. P.	0,1857	45836	0,0356	0	32667	0,2480	27	23988
PSO	20	Média	7,5116	39693	2,7640	47	134754	0,0002	21	296420
		Des. P.	2,2244	4549	0,5116	4	34850	0,0004	3	27164
	50	Média	2,7769	50093	2,9507	50	150463	0,0982	27	289314
		Des. P.	1,1119	6094	0,5181	5	42859	0,3798	9	23644
	100	Média	2,1978	71627	2,8163	44	117689	0,2049	34	285085
		Des. P.	0,6364	6454	0,6420	5	37763	0,5499	15	28748
DE BIN	20	Média	0,8348	60303	0,2669	35	46537	19,2488	494	41589
		Des. P.	0,5991	15955	0,4194	4	20754	0,7429	19	27768
	50	Média	2,4436	79560	0,3890	38	43356	19,6420	512	25738
		Des. P.	1,2644	41858	0,4759	7	14873	0,8057	35	26119
	100	Média	3,6167	78333	0,2025	37	48263	19,2899	494	34511
		Des. P.	0,2897	12190	0,2724	7	19121	0,8257	29	24559
DE EXP1	20	Média	0,9677	37772	1,1589	58	77581	8,8071	326	290596
		Des. P.	0,8521	3995	0,6843	11	15405	5,8984	230	157614
	50	Média	0,3420	79133	0,9339	51	88295	12,8275	439	184525
		Des. P.	0,2629	2685	0,6239	12	20079	3,6736	129	100068
	100	Média	0,2871	146347	0,7837	47	86234	10,5701	375	243043
		Des. P.	0,2565	5046	0,5319	9	17866	5,3600	185	145105
DE EXP2	20	Média	0,8454	60664	1,1344	35	88844	11,8823	412	234644
		Des. P.	0,7401	2934	0,5267	13	46043	4,6562	168	120839
	50	Média	0,3865	133187	1,7604	43	64746	8,6866	238	316731
		Des. P.	0,3319	6396	1,0661	9	22833	7,1589	221	176689
	100	Média	0,1345	254440	1,2895	40	59150	11,0074	350	257387
		Des. P.	0,1375	15009	0,7043	4	17114	6,1536	205	157110

Resultados da função Ackley com dimensão 100.

			Padrão		*APGA, APPSO ou APDE*			*PRoFIGA, PRoFIPSO ou PROFIDE*		
	P		A	B	A	C	B	A	C	B
GA	20	Média	0,4257	206925	0,0292	15	320365	0,2735	15	153046
		Des. P.	0,1693	32818	0,0130	0	48223	0,1120	0	34609
	50	Média	0,4193	459294	0,0406	15	319028	0,3818	27	148267
		Des. P.	0,1496	34871	0,0218	0	60414	0,1983	14	57957
	100	Média	2,5828	470223	0,0390	15	312843	0,4300	85	157714
		Des. P.	0,1810	56984	0,0164	0	48339	0,2295	35	56463
PSO	20	Média	14,8632	89625	6,0830	23	228681	3,0353	20	392903
		Des. P.	1,0578	10157	0,4597	3	42984	0,6581	3	20050
	50	Média	8,6599	116060	6,0125	24	232543	2,8631	26	394516
		Des. P.	2,2811	14512	0,7084	2	45094	0,8218	9	23759
	100	Média	6,0165	163913	6,1517	25	239327	3,0525	37	400609
		Des. P.	1,8402	23374	0,8104	2	39395	1,1450	17	23955
DE BIN	20	Média	1,2849	90549	0,9668	32	144306	20,3270	381	17997
		Des. P.	0,3587	16761	0,5947	5	45305	0,3588	129	15224
	50	Média	3,1722	130223	0,8428	31	162201	20,3482	297	17418
		Des. P.	0,9406	48535	0,6773	8	54775	0,5409	135	17265
	100	Média	5,3771	179367	0,7534	34	166479	20,3121	349	16463
		Des. P.	3,7247	49580	0,5816	6	51869	0,3258	134	14666
DE EXP1	20	Média	0,4171	62972	2,8078	47	310450	9,5938	203	350445
		Des. P.	0,3414	3826	0,6220	12	83691	7,2539	224	174515
	50	Média	0,2922	139087	2,6814	52	258951	9,9277	228	330314
		Des. P.	0,2999	4807	0,3958	17	75225	7,7361	209	170515
	100	Média	0,1357	263753	2,5061	48	259021	5,4972	106	433461
		Des. P.	0,1331	11009	0,3069	13	71992	6,4535	152	139113
DE EXP2	20	Média	0,6081	105253	2,0043	32	323317	10,6070	231	338052
		Des. P.	0,7128	4743	0,3466	13	51746	6,6444	226	166891
	50	Média	0,2704	233073	2,0782	34	278893	8,5093	181	365338
		Des. P.	0,1982	6677	0,3404	14	76541	7,7438	208	171050
	100	Média	0,2100	438327	1,9497	31	283979	9,6873	242	345254
		Des. P.	0,1768	18916	0,5225	14	87221	7,4131	221	175958

Resultados da função Griewangk com dimensão 10

			Padrão		*APGA, APPSO ou APDE*			*PRoFIGA, PRoFIPSO ou PROFIDE*		
	P		A	B	A	C	B	A	C	B
GA	20	Média	0,1944	28241	0,3403	16	32631	0,3665	15	23174
		Des. P.	0,0950	11276	0,3057	5	13683	0,3270	0	14448
	50	Média	0,4214	44933	0,2591	16	28564	0,3735	22	22847
		Des. P.	0,2548	24318	0,1668	3	12087	0,2031	5	13843
	100	Média	0,5350	46426	0,2466	15	28768	0,6076	94	25362
		Des. P.	0,3240	24852	0,1827	0	15256	0,2412	33	12518
PSO	20	Média	0,1007	13025	0,1124	61	21998	0,1620	50	60829
		Des. P.	0,0814	3560	0,0633	14	2678	0,2323	121	10392
	50	Média	0,0795	23947	0,1337	64	24284	0,1187	28	68101
		Des. P.	0,0389	4916	0,1129	10	4312	0,0704	7	9172
	100	Média	0,0620	36833	0,1071	54	24370	0,1806	82	66404
		Des. P.	0,0285	9473	0,0511	11	4574	0,2175	141	15321
DE BIN	20	Média	0,8060	18932	0,0300	42	23295	0,0678	19	97500
		Des. P.	1,0246	3651	0,0353	35	12790	0,0433	4	7653
	50	Média	0,2948	33963	0,0830	63	38802	0,0396	29	111962
		Des. P.	0,2641	2412	0,0742	33	59501	0,0167	6	7586
	100	Média	0,1828	60060	0,0311	34	26142	0,0442	47	115702
		Des. P.	0,1368	12132	0,0282	22	21915	0,0287	20	7985
DE EXP1	20	Média	0,7440	19973	0,1429	42	23023	0,3619	59	106454
		Des. P.	0,9083	2653	0,1990	18	11391	1,2100	149	30009
	50	Média	0,1442	36640	0,0678	29	19808	0,0585	30	116051
		Des. P.	0,1097	3394	0,0645	17	6496	0,0389	6	10951
	100	Média	0,1110	61140	0,0426	25	27129	0,0379	49	120102
		Des. P.	0,1290	7246	0,0354	13	14344	0,0288	16	10302
DE EXP2	20	Média	0,7717	31653	0,1614	18	14504	0,6562	106	103090
		Des. P.	0,6603	4858	0,1691	8	4090	1,1160	179	37165
	50	Média	0,1855	58907	0,1183	17	14768	0,8990	143	106628
		Des. P.	0,1752	13062	0,1264	5	1253	1,4906	195	47231
	100	Média	0,1311	92413	0,0953	21	17160	0,2714	71	134434
		Des. P.	0,1460	14503	0,0872	13	1618	0,4967	116	35348

Resultados da função Griewangk com dimensão 20

			Padrão		*APGA, APPSO ou APDE*			*PRoFIGA, PRoFIPSO ou PROFIDE*		
	P		A	B	A	C	B	A	C	B
GA	20	Média	0,1845	66104	0,1635	15	61607	0,4438	15	38857
		Des. P.	0,1540	16858	0,1281	0	20828	0,3041	0	16313
	50	Média	0,5864	87769	0,2343	15	53710	0,5968	24	44679
		Des. P.	0,3429	26935	0,2963	0	17618	0,3116	12	12429
	100	Média	0,7732	110631	0,1441	15	72160	0,6738	61	44307
		Des. P.	0,2573	31934	0,1450	0	29477	0,3102	45	19438
PSO	20	Média	0,0231	16819	0,0336	75	33035	0,0212	18	86284
		Des. P.	0,0226	1064	0,0261	8	4870	0,0175	4	3972
	50	Média	0,0272	24543	0,0275	77	32719	0,0287	28	86597
		Des. P.	0,0254	1164	0,0254	5	2947	0,0169	9	4567
	100	Média	0,0318	36227	0,0378	78	33838	0,0382	39	89058
		Des. P.	0,0205	3304	0,0287	7	2975	0,0306	15	4452
DE BIN	20	Média	0,5750	20013	0,0052	43	22627	0,0209	18	144872
		Des. P.	0,9244	965	0,0162	7	523	0,0223	4	9573
	50	Média	0,4081	37757	0,0072	43	24581	7,4955	147	114092
		Des. P.	0,4040	2703	0,0178	8	2060	17,6329	211	46811
	100	Média	0,1306	69200	0,0028	42	26792	0,8105	70	132874
		Des. P.	0,1252	4891	0,0052	5	2551	2,0455	117	21300
DE EXP1	20	Média	1,2831	23779	0,1264	47	29959	1,7893	113	149909
		Des. P.	1,3478	2340	0,1172	16	6466	3,7137	198	42174
	50	Média	0,2517	48440	0,2364	51	30514	3,8694	130	158215
		Des. P.	0,1984	5838	0,5018	11	4554	8,4145	212	56881
	100	Média	0,1505	69867	0,0697	48	29984	0,7084	72	177019
		Des. P.	0,1600	8167	0,1130	9	2860	2,2944	117	39292
DE EXP2	20	Média	0,7172	36937	0,4878	32	24161	4,2266	139	168350
		Des. P.	0,6363	6747	0,5052	4	2186	6,3244	208	63596
	50	Média	0,2323	65893	0,1943	32	24499	1,0246	47	201014
		Des. P.	0,2129	9995	0,2985	9	1765	1,4589	90	41923
	100	Média	0,1812	101133	0,0739	33	27973	5,7469	162	174385
		Des. P.	0,1590	14811	0,0654	7	2393	11,2149	203	77999

Resultados da função Griewangk com dimensão 50

			Padrão		*APGA, APPSO ou APDE*			*PRoFIGA, PRoFIPSO ou PROFIDE*		
	P		A	B	A	C	B	A	C	B
GA	20	Média	0,6339	102757	0,0514	15	204064	0,5338	17	91214
		Des. P.	0,2985	29953	0,0396	0	54091	0,3309	5	27737
	50	Média	0,8021	176154	0,0520	15	188826	0,6722	30	84955
		Des. P.	0,2367	45098	0,0807	0	49214	0,2731	22	31524
	100	Média	0,9941	248282	0,0997	15	157267	0,7987	42	89145
		Des. P.	0,0474	35183	0,1626	0	61143	0,2197	42	31322
PSO	20	Média	0,6015	50121	0,0768	63	93093	0,0113	20	149131
		Des. P.	1,0292	8114	0,0692	4	8645	0,0204	4	9341
	50	Média	0,0562	51787	0,2165	60	92537	0,0122	25	149228
		Des. P.	0,0925	3865	0,4673	3	7126	0,0148	9	11411
	100	Média	0,0316	69427	0,0974	62	94592	0,0179	41	152778
		Des. P.	0,0480	3927	0,1245	7	14592	0,0227	20	7237
DE BIN	20	Média	0,4798	35151	0,1775	42	48297	309,7213	494	78991
		Des. P.	0,5390	2592	0,3360	8	21368	158,8116	19	45217
	50	Média	0,2782	92663	0,0930	39	37093	457,6146	501	44112
		Des. P.	0,1880	3823	0,1681	8	4040	183,4446	24	32892
	100	Média	0,0856	184413	0,1443	41	44167	431,2011	487	47724
		Des. P.	0,0664	4405	0,1929	6	13511	196,4168	6	43181
DE EXP1	20	Média	0,6097	42796	1,4027	43	134117	10,0013	49	345947
		Des. P.	0,6208	7833	1,4654	19	46606	34,7303	121	78905
	50	Média	0,2659	83587	0,7841	42	119581	17,7026	118	304271
		Des. P.	0,1928	8653	0,3949	13	28916	35,6705	184	98667
	100	Média	0,1476	137313	0,6895	39	116165	32,7851	215	300242
		Des. P.	0,1042	7157	0,2945	18	37159	39,4497	216	139049
DE EXP2	20	Média	0,7922	53205	0,2769	40	66467	86,7658	280	245017
		Des. P.	0,8591	8417	0,2538	11	11811	111,4807	254	129680
	50	Média	0,2303	103337	0,7847	41	60685	30,3818	120	341202
		Des. P.	0,1934	9695	0,9738	6	7866	61,8577	192	110077
	100	Média	0,1339	180920	0,4459	39	62661	94,8504	254	256131
		Des. P.	0,1112	7593	0,6447	9	8764	123,3539	233	147956

Resultados da função Griewangk com dimensão 100

			Padrão		*APGA, APPSO ou APDE*			*PRoFIGA, PRoFIPSO ou PROFIDE*		
	P		A	B	A	C	B	A	C	B
GA	20	Média	0,9891	200497	1,3297	15	240284	0,7038	18	157675
		Des. P.	0,0992	34991	0,4350	0	79569	0,2596	6	45296
	50	Média	0,9714	382074	1,2172	15	269099	0,8440	32	156614
		Des. P.	0,1155	63025	0,5516	0	94021	0,1734	24	27164
	100	Média	1,1092	478117	1,1102	15	255682	0,9342	48	159863
		Des. P.	0,1129	64982	0,6080	0	67702	0,1801	49	45777
PSO	20	Média	2,5817	150385	1,2150	44	289099	0,3709	21	289595
		Des. P.	3,4412	19266	1,3989	2	37177	0,3070	3	21163
	50	Média	0,9052	172347	1,0035	45	285746	0,4047	29	299008
		Des. P.	2,4078	19877	0,7954	3	41665	0,4162	9	24443
	100	Média	0,4182	184793	0,6369	44	282217	0,3916	43	308210
		Des. P.	0,6107	17713	0,5688	3	36604	0,3697	18	19370
DE BIN	20	Média	0,4990	67289	0,4867	30	90962	1787,8499	487	11587
		Des. P.	0,5831	5130	0,3954	4	15476	152,9405	8	12112
	50	Média	0,2043	241887	0,5310	30	92020	1765,5199	486	13547
		Des. P.	0,1685	10069	0,4339	5	23454	149,8832	53	6867
	100	Média	0,4022	399080	0,3989	30	90937	1764,5051	491	11323
		Des. P.	0,3354	95482	0,3497	5	17717	155,7095	23	5065
DE EXP1	20	Média	0,7001	65832	1,6615	34	419414	26,3010	46	478090
		Des. P.	0,5320	8791	0,6248	11	40592	85,7206	122	66987
	50	Média	0,3790	132617	1,4908	31	390543	147,9815	139	407127
		Des. P.	0,3662	8483	0,5517	16	54809	264,8092	200	145822
	100	Média	0,2309	238433	1,6989	33	378024	223,0359	236	356495
		Des. P.	0,2255	12800	0,3938	13	58507	287,6465	217	153021
DE EXP2	20	Média	0,8024	83203	0,4558	27	278591	434,9725	275	302508
		Des. P.	1,4967	10459	0,3838	10	48572	434,0544	229	172477
	50	Média	0,3167	173020	0,6645	28	265859	160,3623	147	414023
		Des. P.	0,2044	9793	0,3959	13	48791	277,9389	217	139675
	100	Média	0,0773	315927	0,7352	31	243822	286,8118	244	357109
		Des. P.	0,0654	12293	0,4895	11	65735	337,8906	219	160284

Resultados da função Rastrigin com dimensão 10

			Padrão		*APGA, APPSO ou APDE*			*PRoFIGA, PRoFIPSO ou PROFIDE*		
	P		A	B	A	C	B	A	C	B
GA	20	Média	0,0295	38849	0,0763	15	33365	0,2555	15	29963
		Des. P.	0,0443	13461	0,1682	0	11945	0,3326	0	8349
	50	Média	0,3867	42527	0,7386	17	34340	0,4158	22	26367
		Des. P.	0,4848	20324	1,7975	6	16518	0,3874	5	9427
	100	Média	0,9682	41892	0,2796	15	33768	0,5724	98	26905
		Des. P.	0,4757	17697	0,4128	1	13969	0,7142	31	11326
PSO	20	Média	7,5617	13951	1,0613	51	28848	3,5155	20	116631
		Des. P.	3,7190	2119	0,9563	11	6865	1,6762	4	14117
	50	Média	5,3064	25570	1,5256	44	26740	7,8540	119	99626
		Des. P.	2,7031	4434	1,4004	11	5685	9,0573	191	48313
	100	Média	3,4492	44613	0,9286	48	31183	6,7382	71	101674
		Des. P.	2,5189	8930	1,0276	12	7133	9,7246	116	30671
DE BIN	20	Média	1,6649	12799	3,1324	57	35651	25,5691	485	36213
		Des. P.	1,8117	1073	1,8815	32	22943	5,4495	28	12528
	50	Média	0,9291	33427	3,0528	44	38376	24,5456	501	40382
		Des. P.	0,9930	1983	1,7744	27	26942	7,9480	24	17560
	100	Média	1,7337	51567	2,4627	31	56441	27,8602	494	29368
		Des. P.	3,1322	15110	1,6485	26	66712	5,5416	19	12598
DE EXP1	20	Média	2,2078	10084	3,0247	24	26551	10,3648	359	87342
		Des. P.	1,3611	380	1,6482	12	15290	8,4853	216	56581
	50	Média	0,7232	23277	1,7119	27	26862	16,4096	483	52941
		Des. P.	0,8916	565	1,0401	11	12675	8,3858	35	32088
	100	Média	0,3152	36740	1,9272	28	33837	12,0467	451	67102
		Des. P.	0,3754	1268	1,3099	13	16987	9,4115	109	32307
DE EXP2	20	Média	2,3277	18917	3,1711	22	15044	14,4694	476	68945
		Des. P.	2,2813	371	2,3700	14	5151	5,1265	55	34592
	50	Média	0,8745	33653	2,8989	16	15850	16,5319	470	52758
		Des. P.	1,1435	1772	2,3560	3	4277	5,2868	81	17644
	100	Média	0,4832	57560	2,7741	22	22350	13,5994	512	63801
		Des. P.	0,4062	3177	2,3180	11	12070	3,8042	96	15551

Resultados da função Rastrigin com dimensão 20

			Padrão		*APGA, APPSO ou APDE*			*PRoFIGA, PRoFIPSO ou PROFIDE*		
	P		A	B	A	C	B	A	C	B
GA	20	Média	0,2969	62907	0,0410	15	65911	0,4181	15	45997
		Des. P.	0,6135	25084	0,0864	1	25200	0,4914	0	16904
	50	Média	0,8399	84688	0,0469	15	57335	0,7899	22	40792
		Des. P.	1,0295	20894	0,0701	0	13242	0,7321	7	13579
	100	Média	1,8873	94731	0,0203	15	56963	0,8546	95	43913
		Des. P.	1,7039	31725	0,0349	1	11654	0,8216	34	21413
PSO	20	Média	37,4104	17701	3,1849	59	54961	18,1082	20	147605
		Des. P.	11,1481	2906	2,0994	13	7676	6,1471	3	11733
	50	Média	24,2770	33413	4,2651	64	52031	29,4320	89	141127
		Des. P.	6,1310	6128	5,2799	13	11986	23,4400	162	43557
	100	Média	19,7665	49573	2,7859	55	54073	17,8429	37	160568
		Des. P.	7,3004	7777	2,1339	5	7690	5,5711	17	15248
DE BIN	20	Média	21,5550	48571	8,1785	57	57760	121,2601	501	37996
		Des. P.	19,7589	25149	3,4517	21	34307	16,6542	33	20695
	50	Média	55,1912	30925	8,6580	64	71561	122,1376	478	37116
		Des. P.	9,0189	10093	5,0597	19	64696	18,5179	50	20437
	100	Média	72,5425	30847	7,5076	55	81948	125,5069	482	35690
		Des. P.	12,8200	16834	5,7906	28	45507	17,8818	51	18071
DE EXP1	20	Média	1,5495	21292	6,9294	28	63691	52,4138	425	115267
		Des. P.	1,4254	472	4,2301	14	20760	36,9209	168	81702
	50	Média	0,9195	38843	6,4601	25	65521	49,6736	424	112718
		Des. P.	0,7680	1311	2,6438	8	21098	28,1714	165	68603
	100	Média	0,6178	68340	6,1477	25	68547	62,7005	484	80121
		Des. P.	0,4086	2038	3,3764	9	28125	22,7695	87	34464
DE EXP2	20	Média	1,4596	31563	6,3838	19	43589	63,9530	477	84713
		Des. P.	1,8924	1662	4,0351	7	17849	15,0533	85	33169
	50	Média	1,9009	62603	7,7620	24	44264	58,0184	453	109720
		Des. P.	1,4717	3262	4,1003	10	14467	24,9092	118	65236
	100	Média	0,6082	109967	5,4271	22	46683	68,8945	491	86445
		Des. P.	0,5730	3094	2,6343	7	15941	29,0351	14	36965

Resultados da função Rastrigin com dimensão 50

			Padrão		*APGA, APPSO ou APDE*			*PRoFIGA, PRoFIPSO ou PROFIDE*		
	P		A	B	A	C	B	A	C	B
GA	20	Média	1,3668	120159	0,1788	15	141154	0,5355	16	89304
		Des. P.	1,5737	33050	0,3580	0	44144	0,3118	5	20628
	50	Média	1,8360	201068	0,0690	15	163087	1,1758	23	82722
		Des. P.	1,9582	40404	0,0978	0	71445	0,9925	13	30006
	100	Média	4,2860	262283	0,0993	15	204999	1,4267	46	89708
		Des. P.	2,2179	51970	0,0999	0	75972	2,2811	42	27903
PSO	20	Média	142,4175	44847	19,0960	22	156772	107,7211	20	230170
		Des. P.	27,6425	5281	7,2339	5	24543	21,7413	3	17764
	50	Média	149,0444	53640	23,2461	21	133941	133,9768	57	212527
		Des. P.	36,3672	5982	6,2212	3	18933	100,4518	119	56419
	100	Média	131,1353	76140	21,5834	23	140784	141,6103	98	206776
		Des. P.	35,0242	9418	4,5876	4	14498	127,7541	159	77957
DE BIN	20	Média	286,2664	27328	7,7012	43	175186	587,7705	505	20474
		Des. P.	15,9576	9030	6,1474	8	57504	44,0391	34	14070
	50	Média	346,8399	28470	6,1828	41	188379	566,0141	498	29623
		Des. P.	22,5810	8983	5,1534	7	70466	34,9134	31	18626
	100	Média	378,9809	35520	9,1479	41	148544	599,2071	508	15871
		Des. P.	35,2122	16254	7,0844	9	58664	44,9379	27	12899
DE EXP1	20	Média	1,9382	41760	11,6848	16	137404	207,3894	341	224598
		Des. P.	1,6848	1259	6,1594	3	46714	121,6827	200	147236
	50	Média	1,0346	87980	11,8298	20	126904	231,9806	412	189702
		Des. P.	1,0319	3108	5,3718	11	25765	103,6913	162	113714
	100	Média	0,5615	164547	11,5254	22	121001	246,1538	443	171276
		Des. P.	0,5926	4376	4,0087	12	30777	84,3072	123	98489
DE EXP2	20	Média	3,0972	67216	9,2851	24	154412	274,4810	395	185060
		Des. P.	1,6865	1947	4,3522	17	30066	159,0977	183	140648
	50	Média	0,8821	146320	10,0176	15	134687	285,4880	437	170970
		Des. P.	0,6232	3303	2,9823	1	22177	143,7993	149	116756
	100	Média	0,7800	273327	11,1414	17	147301	334,0716	457	124157
		Des. P.	0,6475	7129	6,3545	9	40935	95,8357	66	57749

Resultados da função Rastrigin com dimensão 100

			Padrão		*APGA, APPSO ou APDE*			*PRoFIGA, PRoFIPSO ou PROFIDE*		
	P		A	B	A	C	B	A	C	B
GA	20	Média	6,5495	256959	0,1129	15	331398	1,0976	18	171108
		Des. P.	3,9810	36551	0,0859	0	49636	0,9759	7	31673
	50	Média	13,0242	407769	1,7626	15	285243	2,4345	23	159326
		Des. P.	14,8109	95017	6,2637	0	40982	1,9990	14	59120
	100	Média	24,5715	407486	0,1082	15	305804	3,5785	58	154928
		Des. P.	9,6922	112720	0,0653	0	55091	2,2825	49	45819
PSO	20	Média	416,7590	131211	210,9509	17	254078	462,0782	114	292093
		Des. P.	46,2395	20542	29,6416	2	42962	308,7673	193	133494
	50	Média	357,6891	141047	201,7503	17	262822	302,1889	25	367148
		Des. P.	37,3191	13849	28,3589	2	38928	41,8118	6	25370
	100	Média	337,9541	173080	226,7422	17	245336	398,0075	99	331698
		Des. P.	55,9061	15763	40,3519	1	55511	271,6807	158	121524
DE BIN	20	Média	776,9349	35468	3,9169	32	286951	1436,6513	505	10218
		Des. P.	37,1549	12058	2,5226	4	104123	58,0600	34	9599
	50	Média	943,4627	46118	2,4458	31	302145	1423,0167	505	10840
		Des. P.	63,6544	16657	1,7917	6	66366	52,3377	34	7584
	100	Média	1146,3412	38587	2,5132	30	335063	1458,7002	516	5098
		Des. P.	98,6832	18584	2,5669	6	84053	31,3242	41	3307
DE EXP1	20	Média	1,8670	74535	18,2609	15	275899	349,3146	159	369855
		Des. P.	2,5169	1833	7,1998	0	52017	435,6741	214	181479
	50	Média	0,7005	168673	15,4439	15	272187	304,1268	186	390989
		Des. P.	0,5520	4276	7,0397	0	56678	337,9749	214	140425
	100	Média	0,4376	321540	15,8846	15	279307	529,1866	286	297978
		Des. P.	0,4710	5304	9,1093	0	59589	421,8863	223	174888
DE EXP2	20	Média	3,5109	131633	16,1682	15	257687	417,9605	222	353676
		Des. P.	2,9603	2946	7,3490	0	48775	426,0764	232	168843
	50	Média	1,4594	287560	14,0386	15	266217	468,8885	224	334794
		Des. P.	1,5496	7002	6,8263	0	44467	425,8873	206	166754
	100	Média	0,9364	499707	13,9617	15	272248	568,8177	298	289304
		Des. P.	0,8348	524	4,6822	0	35504	508,8344	212	186660

Resultados da função Rosenbrock com dimensão 10

			Padrão		*APGA, APPSO ou APDE*			*PRoFIGA, PRoFIPSO ou PROFIDE*		
	P		A	B	A	C	B	A	C	B
GA	20	Média	7,6396	70936	6,0729	17	69687	6,9852	17	54979
		Des. P.	0,6894	35255	3,0724	7	46732	1,9733	5	47021
	50	Média	8,1462	58670	7,3642	16	53562	7,5529	23	37115
		Des. P.	0,5008	22817	1,6209	3	40387	1,0794	13	27717
	100	Média	8,2660	59104	7,3976	17	70695	7,4479	85	35655
		Des. P.	0,2454	28435	5,0925	9	43718	1,0560	41	13962
PSO	20	Média	0,0121	123167	0,0002	72	162696	0,0000	24	137186
		Des. P.	0,0285	21745	0,0001	18	8857	0,0000	3	18703
	50	Média	0,0060	217803	0,0001	60	159906	0,0000	31	131991
		Des. P.	0,0134	54745	0,0000	13	10653	0,0000	7	12211
	100	Média	0,2713	381053	0,0002	72	159303	0,0021	39	144139
		Des. P.	1,0281	72586	0,0001	14	9079	0,0082	17	22390
DE BIN	20	Média	2,5728	321908	1,4548	152	1071790	17,8927	323	188239
		Des. P.	3,5307	182328	1,0682	94	1318776	17,2330	231	227968
	50	Média	0,7350	172917	1,2382	145	1282013	8,5494	250	255948
		Des. P.	1,6556	102755	1,0189	110	1285270	8,4420	225	236108
	100	Média	3,4442	128027	1,2248	139	1229893	11,6923	282	252218
		Des. P.	3,0639	83817	1,0291	91	1381733	13,9395	227	240168
DE EXP1	20	Média	1,5459	120005	6,9038	49	111638	9,5201	239	266657
		Des. P.	2,6252	87594	1,1589	12	117097	10,0611	247	231924
	50	Média	0,1491	95680	7,3426	49	108317	8,0864	178	340069
		Des. P.	0,2129	30333	1,8541	16	139051	12,4819	220	224807
	100	Média	1,4510	99200	6,7009	53	126190	4,5862	140	372832
		Des. P.	2,1108	35719	0,7649	17	66593	7,1204	186	207677
DE EXP2	20	Média	0,3690	108989	1,0642	44	107475	14,0798	202	149997
		Des. P.	0,9052	28499	0,3192	6	31426	15,8829	235	118020
	50	Média	1,5217	112863	1,0592	43	119440	9,9120	177	195796
		Des. P.	2,0080	53983	0,2797	6	13752	10,0378	221	162822
	100	Média	3,5984	101280	1,1013	40	120596	13,8722	290	216660
		Des. P.	2,0575	47439	0,1919	6	25401	15,9947	226	220307

Resultados da função Rosenbrock com dimensão 20

			Padrão		*APGA, APPSO ou APDE*			*PRoFIGA, PRoFIPSO ou PROFIDE*		
	P		A	B	A	C	B	A	C	B
GA	20	Média	17,4285	62063	16,0493	15	451070	17,4949	16	61801
		Des. P.	0,8248	34106	3,0803	1	1260268	0,7775	5	32889
	50	Média	18,0822	73344	16,4690	16	88788	16,7811	24	56733
		Des. P.	0,3860	33418	1,6911	2	26968	2,6373	16	43094
	100	Média	18,1440	118096	17,1760	15	211487	17,0161	59	66528
		Des. P.	0,2498	26448	0,9024	1	306442	1,5081	47	24879
PSO	20	Média	2,3739	218785	0,0009	114	480452	0,7974	24	215545
		Des. P.	3,5732	40703	0,0002	15	47092	1,6506	3	8551
	50	Média	1,1684	417310	0,0008	114	479716	0,0001	35	216066
		Des. P.	1,8150	77780	0,0004	12	59242	0,0000	8	8399
	100	Média	1,7834	499973	0,0007	120	484275	0,5316	34	230420
		Des. P.	2,1906	46	0,0002	8	111040	1,4028	14	11584
DE BIN	20	Média	9,4511	472408	2,4112	76	2072031	208,7782	390	130042
		Des. P.	4,6551	105337	4,7484	9	539345	163,7210	193	192405
	50	Média	8,0884	387877	1,1274	78	2133207	179,7841	242	271599
		Des. P.	5,4339	173815	0,1051	7	174466	325,7801	245	227519
	100	Média	13,8961	212427	3,1444	77	1957935	177,2624	455	56007
		Des. P.	3,0946	160220	5,3253	7	677216	106,5217	117	32710
DE EXP1	20	Média	14,0822	395587	17,0688	34	131916	30,1408	79	302173
		Des. P.	3,8698	150798	1,4158	11	114320	48,7983	158	149650
	50	Média	10,8909	416783	17,5962	29	66207	49,6096	121	227732
		Des. P.	2,9137	172068	1,0163	11	24558	92,7774	190	130989
	100	Média	14,0518	245293	16,7624	27	74098	19,6190	74	377774
		Des. P.	2,1690	198706	1,1214	7	30585	23,5227	116	142154
DE EXP2	20	Média	7,8515	445957	7,0441	35	542460	97,7438	217	137147
		Des. P.	9,5891	102146	3,0725	7	159958	106,6557	225	88618
	50	Média	13,9173	196220	8,1880	37	467066	58,2485	189	148870
		Des. P.	1,5643	140691	3,1582	5	131590	59,0987	213	78456
	100	Média	15,5218	133840	7,7485	36	503549	96,3623	253	135402
		Des. P.	0,7391	42707	3,0514	9	157887	112,6184	215	85864

Resultados da função Rosenbrock com dimensão 50.

			Padrão		*APGA, APPSO ou APDE*			*PRoFIGA, PRoFIPSO ou PROFIDE*		
	P		A	B	A	C	B	A	C	B
GA	20	Média	47,4770	117638	49,0977	15	571587	50,5647	20	142549
		Des. P.	0,6841	31952	11,4625	0	838833	14,1071	8	70328
	50	Média	47,7160	177789	46,8531	15	332760	47,0728	27	117605
		Des. P.	0,4487	42432	0,7603	0	206308	0,8633	20	34731
	100	Média	48,0089	241602	46,6460	15	377913	50,4353	42	135189
		Des. P.	0,2108	49266	0,8625	0	350013	12,9571	46	40378
PSO	20	Média	8,0558	576284	0,5332	83	1650854	14,7402	24	458448
		Des. P.	12,1445	229821	1,4028	6	213632	10,7065	4	87618
	50	Média	6,2142	1119993	0,2672	86	1687318	11,1778	30	491467
		Des. P.	10,3142	446000	1,0293	8	266602	5,8203	9	28766
	100	Média	8,9502	1915740	1,0647	84	1600357	13,8153	46	471254
		Des. P.	10,0354	605277	1,8248	9	345193	10,8151	14	97428
DE BIN	20	Média	44,9284	499664	49,0891	48	190314	5975,2045	505	26577
		Des. P.	10,2230	230	12,0167	7	148518	1463,6365	34	27807
	50	Média	41,9807	332837	46,1472	44	158314	5847,5221	482	27411
		Des. P.	3,5754	149944	1,5781	4	106411	1633,4903	72	23344
	100	Média	47,2696	200187	46,1791	47	158670	6103,9923	494	20529
		Des. P.	3,0340	75199	1,0583	6	85406	1625,0668	19	16130
DE EXP1	20	Média	43,4238	405632	64,7277	30	278906	124,7988	47	309837
		Des. P.	1,9091	169138	22,9347	13	85974	290,8730	122	69349
	50	Média	45,5197	119780	57,0163	23	241532	117,6616	62	319011
		Des. P.	1,2900	46908	25,9208	8	87737	262,5847	118	81160
	100	Média	45,7496	154260	55,3200	25	187290	375,5792	187	276033
		Des. P.	0,4474	24300	17,7062	10	39527	632,7857	204	127503
DE EXP2	20	Média	46,0749	133559	52,8819	19	153618	789,8334	226	233920
		Des. P.	2,6366	98333	19,2210	12	50692	1220,9868	232	132017
	50	Média	46,6103	141913	49,5759	15	125468	428,6565	167	261235
		Des. P.	1,3556	22765	10,5321	1	16686	598,9862	206	104837
	100	Média	46,4857	230740	46,4388	15	136640	761,7141	317	191420
		Des. P.	0,3862	19423	0,8273	0	21347	664,0056	213	109373

Resultados da função Rosenbrock com dimensão 100.

			Padrão		*APGA, APPSO ou APDE*			*PRoFIGA, PRoFIPSO ou PROFIDE*		
	P		A	B	A	C	B	A	C	B
GA	20	Média	100,2779	222595	104,9599	15	377271	103,4568	21	223839
		Des. P.	11,9289	35975	27,2086	0	170746	18,5414	9	85602
	50	Média	97,4498	361962	101,9016	15	271304	103,7133	27	187127
		Des. P.	0,4703	64312	14,1591	0	115816	18,5264	20	38578
	100	Média	98,2114	465068	97,3465	15	321753	96,7011	68	184191
		Des. P.	1,3776	82446	1,8944	0	75968	1,3561	52	44262
PSO	20	Média	37,0051	1433481	6,1591	43	3706805	121,2714	20	668008
		Des. P.	41,0994	500631	20,5842	5	986868	29,0560	4	245009
	50	Média	42,0737	2148877	21,0968	43	3264438	94,3274	26	984189
		Des. P.	34,0673	751852	31,5226	8	1420134	30,4579	8	538837
	100	Média	71,4640	2796513	8,6172	43	3783563	57,2344	40	1514291
		Des. P.	21,8128	1533941	17,5340	6	847569	40,5514	14	864631
DE BIN	20	Média	102,8271	479976	95,6574	33	172789	21709,2039	491	16438
		Des. P.	19,1018	75515	2,1312	4	75145	2587,3748	14	10529
	50	Média	96,1072	361090	98,5648	32	214106	21008,3509	484	12695
		Des. P.	4,3007	126972	12,9274	5	116601	2719,0275	94	8906
	100	Média	14966,7937	47307	96,5798	30	171927	20089,8511	505	10488
		Des. P.	12521,6513	48099	0,6305	3	72445	2650,2088	34	6212
DE EXP1	20	Média	98,3252	169617	152,1388	31	976786	2212,6915	134	386655
		Des. P.	6,0060	138094	29,9763	15	273845	3775,9659	207	172472
	50	Média	95,9072	159653	121,7625	38	721129	1627,9961	154	407693
		Des. P.	1,1571	22247	32,6678	15	141420	3209,9458	216	156672
	100	Média	95,8649	270567	111,9390	40	777986	1195,4637	108	440127
		Des. P.	0,3177	12899	23,7360	16	172943	3165,4038	176	135792
DE EXP2	20	Média	100,4006	141481	119,7270	17	421129	4438,2903	234	318050
		Des. P.	9,6052	20524	27,1696	4	87143	5433,3347	240	194463
	50	Média	97,7447	242310	111,7125	17	392271	2467,8944	206	368287
		Des. P.	6,0787	17979	23,7437	5	79296	3252,4264	230	166725
	100	Média	95,9592	434953	103,4230	16	359691	1338,3185	129	427026
		Des. P.	0,7488	23719	18,0962	2	79060	2352,2652	186	135345

Livros Grátis

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

Milhares de Livros para Download:

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