J Glob Optim (2007) 39: 459–471 DOI
10.1007 / s10898-007-9149-x

PAPEL ORIGINAL

Um algoritmo poderoso e eficiente para otimização de função numérica: algoritmo arti fi cial bee colony (ABC)

Dervis Karaboga · Bahriye Basturk

Recebido: 31 de maio de 2006 / Aceito: 12 de fevereiro de 2007 / Publicado online: 13 de abril de 2007 © Springer Science + Business Media BV 2007

Abstrato A inteligência de enxame é um ramo de pesquisa que modela a população de agentes em interação ou enxames que são capazes de se auto-organizar. Uma colônia de formigas, um bando de pássaros ou um sistema imunológico é um exemplo típico de sistema de enxame. O enxame de abelhas ao redor de sua colmeia é outro exemplo de inteligência de enxame. Arti fi cial Bee Colony Algorithm (ABC) é um algoritmo de otimização baseado no comportamento inteligente do enxame de abelhas. Neste trabalho, o algoritmo ABC é usado para otimizar funções multivariáveis e os resultados produzidos por ABC, Algoritmo Genético (GA), Algoritmo de Enxame de Partículas (PSO) e Algoritmo de Partícula SwarmInspiredEvolutionary (PS-EA) foram comparados. Os resultados mostraram que o ABC supera os outros algoritmos.

Palavras-chave Inteligência de enxame · Colônia de abelhas artificiais · Algoritmo genético do swar de partículas · Otimização de função numérica

1. Introdução

Vários algoritmos heurísticos modernos foram desenvolvidos para resolver problemas de otimização combinatória e numérica [1] Esses algoritmos podem ser classificados em grupos diferentes dependendo dos critérios considerados, como baseado na população, baseado em iteração, estocástico, determinístico, etc. Enquanto um algoritmo que trabalha com um conjunto de soluções e tenta melhorá-las é chamado de baseado na população, aquele o uso de múltiplas iterações para abordar a solução buscada é denominado algoritmo iterativo. Se um algoritmo emprega uma regra probabilística para melhorar uma solução, ele é chamado de probabilístico ou estocástico. Outra classificação pode ser feita dependendo da natureza

D. Karaboga · B. Basturk (B)
Departamento de Engenharia da Computação, Universidade Erciyes, Kayseri, Turquia
e-mail: bahriye@erciyes.e du.tr

D. Karaboga
e-mail: karaboga@erciyes.edu.tr

de fenômeno simulado pelo algoritmo. Este tipo de classificação tem principalmente dois grupos importantes de algoritmos baseados em população: algoritmos evolutivos (EA) e algoritmos baseados em inteligência de enxame. O EA mais popular é o algoritmo genético (GA). GA tenta simular o fenômeno da evolução natural. Na evolução natural, cada espécie busca adaptações benéficas em um ambiente em constante mudança. À medida que as espécies evoluem, os novos atributos são codificados nos cromossomos de membros individuais. Essa informação muda por mutação aleatória, mas a verdadeira força motriz por trás do desenvolvimento evolutivo é a combinação e troca de material cromossômico durante a reprodução. Embora tentativas esporádicas tenham sido feitas para incorporar esses princípios nas rotinas de otimização desde o início dos anos 1960, 2] O termo enxame é usado de uma maneira geral para se referir a qualquer coleção restrita de agentes ou indivíduos que interagem. O exemplo clássico de um enxame é o enxame de abelhas ao redor de sua colméia, mas a metáfora pode ser facilmente estendida a outros sistemas com uma arquitetura semelhante. Uma colônia de formigas pode ser considerada um enxame cujos agentes individuais são formigas; um bando de pássaros é um bando de pássaros. Um sistema imunológico [3] é um enxame de células e moléculas enquanto uma multidão é um enxame de pessoas. O algoritmo de otimização de enxame de partículas (PSO) que simula o comportamento social de bando de pássaros ou escolaridade de peixes foi introduzido por Eberhart e Kennedy em 1995 [4] PSO é uma técnica de otimização estocástica baseada em população e bem adaptada para a otimização de funções não lineares em espaço multidimensional. O PSO recebeu um interesse significativo de pesquisadores que estudam em diferentes áreas de pesquisa e foi aplicado a vários problemas do mundo real [5] Uma versão melhorada do algoritmo PSO é o Algoritmo Evolucionário Inspirado por Enxame de Partículas (PS-EA), que é um modelo híbrido de EA e PSO [21] Ele compensa as limitações do PSO ao resolver problemas do mundo real. A fim de evitar indivíduos inviáveis resultantes de atualizações falhas, o PS-EA incorpora PSO com heurísticas de EA no gerador de população e no operador de mutação, mantendo o funcionamento do PSO.

Várias abordagens foram propostas para modelar os comportamentos inteligentes específicos de enxames de abelhas e aplicadas para resolver problemas de tipo combinatório [6 - 14] Tereshko e Loengarov consideravam uma colônia de abelhas como um sistema dinâmico que coleta informações do ambiente e ajusta seu comportamento de acordo com ele. Eles estabeleceram uma ideia robótica sobre o comportamento de forrageamento das abelhas. Normalmente, todos esses robôs são física e funcionalmente idênticos, de modo que qualquer robô pode ser substituído aleatoriamente pelos outros. O enxame possui uma tolerância significativa; a falha em um único agente não interrompe o desempenho de todo o sistema. Os robôs individuais, como os insetos, têm capacidades limitadas e conhecimento limitado do ambiente. Por outro lado, o enxame desenvolve inteligência coletiva. Os experimentos mostraram que robôs semelhantes a insetos são bem-sucedidos em tarefas robóticas reais [6] Eles também desenvolveram um modelo mínimo de seleção de forragem que leva ao surgimento da inteligência coletiva que consiste em três componentes essenciais: fontes de alimento, forrageadoras empregadas e forrageadoras desempregadas. Este modelo define dois modos principais do comportamento: recrutamento para uma fonte de néctar e abandono de uma fonte [7 , 8] Teodorović´ sugeriu usar a inteligência do enxame de abelhas no desenvolvimento de sistemas arti fi ciais destinados a resolver problemas complexos de tráfego e transporte [9 , 10] Teodorović´ também propôs a Metaheurística de Otimização de Colônias de Abelhas (BCO) que é capaz de resolver problemas combinatórios determinísticos, bem como problemas combinatórios caracterizados pela incerteza [11] Drias et al. introduziu uma nova abordagem inteligente ou meta-heurística chamada Bees Swarm Optimization (BSO), que é inspirada no comportamento de

abelhas reais e eles o adaptaram às características do problema de satisfação máxima ponderada (max-sat) [12] Da mesma forma, Benatchba et al. introduziu uma meta-heurística para resolver um problema de 3 satélites com base no processo de reprodução das abelhas [13] Wedde et al. apresentou um novo algoritmo de roteamento chamado BeeHive, que foi inspirado nos métodos e procedimentos comunicativos e avaliativos de abelhas. No algoritmo BeeHive, os agentes apícolas viajam através de regiões de rede chamadas zonas de forrageamento. No caminho, suas informações sobre o estado da rede são entregues para atualizar as tabelas de roteamento local [14]

Os trabalhos dados no parágrafo anterior incluem o tipo combinatório de problemas. Existe apenas um algoritmo de otimização numérica na literatura com base no comportamento inteligente do enxame de abelhas [15] Na Ref. [15], Yang desenvolveu um algoritmo de abelha virtual (VBA) para resolver as otimizações de funções numéricas. Para as funções com dois parâmetros, um enxame de abelhas virtuais é gerado e o enxame começa a se mover aleatoriamente no espaço de fase. Essas abelhas interagem quando encontram algum néctar-alvo correspondente aos valores codificados da função. A solução para o problema de otimização pode ser obtida a partir da intensidade das interações das abelhas [15] Para otimizar funções multivariáveis, Karaboga descreveu um algoritmo de colônia de abelhas arti fi cial (ABC) [16] que é diferente do algoritmo de abelha virtual. Basturk e Karaboga compararam o desempenho do algoritmo ABC com o desempenho do GA na Ref. [17]

Neste artigo, comparamos o desempenho do algoritmo ABC com o do GA, PSO e PS-EA em um conjunto de problemas de otimização numérica multidimensional. Esses algoritmos são escolhidos porque também são algoritmos de inteligência de enxame e baseados em população, como o algoritmo ABC. Na seção 2, o algoritmo ABC proposto é descrito, na Seção 3 experimentos e resultados são apresentados.

2 Algoritmo de colônia de abelhas artificial proposto (ABC)

No algoritmo ABC, a colônia de abelhas artificiais contém três grupos de abelhas: abelhas empregadas, observadores e batedores. Uma abelha que aguarda na área de dança para tomar a decisão de escolha de uma fonte de alimento é chamada de espectador e uma abelha que vai até a fonte de alimento visitada anteriormente é chamada de abelha empregada. Uma abelha que realiza uma busca aleatória é chamada de batedor. No algoritmo ABC, a primeira metade da colônia consiste em abelhas artificiais empregadas e a segunda metade constitui os espectadores. Para cada fonte de alimento, existe apenas uma abelha empregada. Em outras palavras, o número de abelhas empregadas é igual ao número de fontes de alimento ao redor da colmeia. A abelha empregada, cuja fonte de alimento é exaurida pelas abelhas empregadas e observadoras, torna-se escuteira. As principais etapas do algoritmo são fornecidas a seguir:

- Inicializar.
- REPETIR.
  (a) Colocar as abelhas empregadas nas fontes de alimento da memória; (b) Coloque as abelhas observadoras nas fontes de alimento na memória;
  (c) Envie os batedores para a área de busca para descobrir novas fontes de
- alimento. ATÉ (os requisitos são atendidos).

No algoritmo ABC, cada ciclo da busca consiste em três etapas: envio das abelhas empregadas às fontes de alimento e, em seguida, mensuração da quantidade de néctar; seleção das fontes de alimentos pelos espectadores após compartilhar as informações dos empregados

abelhas e determinação da quantidade de néctar dos alimentos; determinar as abelhas batedoras e, em seguida, enviá-las para possíveis fontes de alimento. No estágio de inicialização, um conjunto de posições da fonte de alimento é selecionado aleatoriamente pelas abelhas e suas quantidades de néctar são determinadas. Em seguida, essas abelhas entram na colmeia e compartilham as informações sobre o néctar das fontes com as abelhas que aguardam na área de dança dentro da colmeia. Na segunda etapa, após o compartilhamento das informações, cada abelha empregada vai para a área de alimentação que ela mesma visitou no ciclo anterior, visto que aquela fonte de alimento existe em sua memória, e então escolhe uma nova fonte de alimento por meio de informação visual do bairro. do presente. No terceiro estágio, um observador prefere uma área de fonte de alimento dependendo da informação de néctar distribuída pelas abelhas empregadas na área de dança. À medida que a quantidade de néctar de uma fonte de alimento aumenta, a probabilidade de que essa fonte de alimento seja escolhida por um observador também aumenta. Conseqüentemente, a dança das abelhas empregadas carregando mais néctar recruta os espectadores para as áreas de fonte de alimento com maior quantidade de néctar. Depois de chegar à área selecionada, ela escolhe uma nova fonte de alimento na vizinhança daquela que está na memória, dependendo das informações visuais. As informações visuais são baseadas na comparação das posições das fontes de alimentos. Quando o néctar de uma fonte de alimento é abandonado pelas abelhas, uma nova fonte de alimento é determinada aleatoriamente por uma abelha exploradora e substituída pela abandonada. Em nosso modelo, a cada ciclo no máximo um batedor sai em busca de uma nova fonte de alimento e o número de abelhas empregadas e observadoras era igual. a probabilidade com que aquela fonte de alimento seja escolhida por um observador também aumenta. Portanto, a dança das abelhas empregadas carregando mais néctar recruta os espectadores para as áreas de fonte de alimento com maior quantidade de néctar. Depois de chegar à área selecionada, ela escolhe uma nova fonte de alimento na vizinhança daquela que está na memória a partir de informações visuais. As informações visuais são baseadas na comparação das posições das fontes de alimentos. Quando o néctar de uma fonte de alimento é abandonado pelas abelhas, uma nova fonte de alimento é determinada aleatoriamente por uma abelha exploradora e substitu

No algoritmo ABC, a posição de uma fonte de alimento representa uma possível solução do problema de otimização e a quantidade de néctar de uma fonte de alimento corresponde à qualidade (aptidão) da solução associada. O número de abelhas empregadas ou de abelhas curiosas é igual ao número de soluções na população. Na primeira etapa, o ABC gera uma população inicial distribuída aleatoriamente $P(G = 0)$ de soluções SN (posições de fontes de alimentos), onde SN denota o tamanho da população.
Cada solução (fonte de alimento) x_{eu} ($i = 1, 2, \ldots, SN$) é um D- vetor dimensional. Aqui, D é o número de parâmetros de otimização. Após a inicialização, a população de as posições (soluções) são submetidas a ciclos repetidos, $C = 1, 2, \ldots, C_{max}$, dos processos de busca das abelhas empregadas, das abelhas observadoras e das abelhas escoteiras. Um arti fi cial
A abelha empregada ou observadora produz probabilisticamente uma modificação na posição (solução) em sua memória para encontrar uma nova fonte de alimento e testa a quantidade de néctar (valor de aptidão) da nova fonte (nova solução). No caso de abelhas reais, a produção de novas fontes alimentares é baseada em um processo de comparação das fontes alimentares de uma região em função das informações coletadas, visualmente, pela abelha. Em nosso modelo, a produção de uma nova posição de fonte de alimento também é baseada em um processo de comparação de posições de fonte de alimento. Porém, no modelo, as abelhas arti fi ciais não utilizam nenhuma informação na comparação. Eles selecionam aleatoriamente uma posição de fonte de alimento e produzem uma modificação naquele existente em sua memória, conforme descrito em (2,2) Desde que a quantidade de néctar da nova fonte seja maior que a anterior, a abelha memoriza a nova posição e esquece a antiga. Caso contrário, ela mantém a posição do anterior. Depois que todas as abelhas empregadas completam o processo de busca, elas compartilham as informações sobre o néctar das fontes de alimento (soluções) e suas informações de posição com as abelhas espectadoras na área de dança. Uma abelha observadora avalia as informações sobre o néctar obtidas de todas as abelhas empregadas e escolhe uma fonte de alimento com uma probabilidade relacionada à sua quantidade de néctar. Como no caso da abelha empregada, ela produz uma modificação na posição (solução) em sua memória e verifica a quantidade de néctar da fonte candidata (solução). Desde que seu néctar seja maior que o anterior, a abelha memoriza a nova posição e esquece a antiga.

Uma abelha observadora escolhe uma fonte de alimento dependendo do valor de probabilidade associado a essa fonte de alimento, $p_{eu,}$ calculado pela seguinte expressão (2,1):

$$p_i = \frac{\text{caber}_{eu,}}{\sum_{n=1}^{SN} \text{caber}_n} \tag{2.1}$$

onde caber_{eu} é o valor de aptidão da solução *eu* avaliada por sua abelha empregada, que é proporcional à quantidade de néctar da fonte de alimento na posição *eu* e SN é o número de fontes de alimento que é igual ao número de abelhas empregadas (BN). Desta forma, as abelhas empregadas trocam suas informações com os curiosos.

A fim de produzir uma posição de alimento candidata a partir da antiga, o ABC usa a seguinte expressão (2,2):

$$v_{ij} = x_{ij} + \varphi_{eu\,j}(x_{eu\,j} - x_{kj}), \tag{2.2}$$

Onde $k \in \{1, 2, \ldots, BN\}$ e $j \in \{1, 2, \ldots, D\}$ são índices escolhidos aleatoriamente. Apesar *k* é determinado aleatoriamente, tem que ser diferente de *eu*. $\varphi_{eu\,j}$ é um número aleatório entre [- 1, 1]. Ele controla a produção de uma posição de fonte de alimento vizinho ao redor $x_{eu\,j}$ e a modificação representa a comparação das posições dos alimentos vizinhos visualmente pela abelha. Equação 2,2 mostra que como a diferença entre os parâmetros do $x_{eu\,j}$ e $x_{k,\,j}$ diminui, a perturbação na posição $x_{eu\,j}$ diminui também. Assim, à medida que a busca se aproxima da solução ótima no espaço de busca, o comprimento do passo é reduzido adaptativamente.

Se um parâmetro produzido por esta operação exceder seu limite predeterminado, o parâmetro pode ser definido com um valor aceitável. Neste trabalho, o valor do parâmetro que excede seu limite é definido como seu valor limite.

A fonte de alimento cujo néctar é abandonado pelas abelhas é substituída por uma nova fonte de alimento pelos batedores. No algoritmo ABC, isso é simulado produzindo aleatoriamente uma posição e substituindo-a pela abandonada. No algoritmo ABC, se uma posição não pode ser melhorada ainda mais por meio de um número predeterminado de ciclos chamados *limite* então essa fonte de alimento é considerada abandonada.

Após cada posição de fonte candidata $v_{eu\,j}$ é produzido e então avaliado pelo abelha artificial, seu desempenho é comparado com o de $x_{eu\,j.}$ Se o novo alimento tiver néctar igual ou melhor do que a fonte anterior, ele será substituído pelo antigo na memória. Caso contrário, o antigo é mantido. Em outras palavras, um mecanismo de seleção ganancioso é empregado como a operação de seleção entre as fontes alimentares antigas e atuais.

O algoritmo ABC, de fato, emprega quatro processos de seleção diferentes: (1) um processo de seleção global usado pelas abelhas observadoras artificiais para descobrir regiões promissoras, conforme descrito por (2,1), (2) um processo de seleção local realizado em uma região pelas abelhas arti fi ciais empregadas e os espectadores, dependendo das informações locais (no caso de abelhas reais, esta informação inclui a cor, forma e fragrância das flores) (as abelhas não ser capaz de identificar o tipo de fonte de néctar até chegar ao local certo e discriminar entre as fontes *crescente* lá com base em seu cheiro) para determinar uma fonte de alimento vizinho em torno da fonte na memória, conforme definido em (2,2), (3) um processo de seleção local denominado processo de seleção ganancioso realizado por todas as abelhas em que se a quantidade de néctar da fonte candidata for melhor do que a atual, a abelha esquece o presente e memoriza a fonte candidata. Caso contrário, a abelha mantém o presente na memória. (4) um processo de seleção aleatório realizado por olheiros.

É claro a partir da explicação acima que existem três parâmetros de controle usados no ABC básico: O número de fontes de alimento que é igual ao número de abelhas empregadas ou observadoras (SN), o valor de *limite* e o número máximo do ciclo (MCN).

No caso das abelhas, a taxa de recrutamento representa um *medir* da rapidez com que a colônia de abelhas encontra e explora uma fonte de alimento recém-descoberta. O recrutamento artificial pode representar de forma semelhante a *medição* da velocidade com que as soluções viáveis ou o *boa qualidade* soluções para os difíceis problemas de otimização podem ser descobertas. A sobrevivência e o progresso da colônia de abelhas dependem da rápida descoberta e da utilização eficiente dos melhores recursos alimentares. Da mesma forma, a solução bem-sucedida de problemas difíceis de engenharia está ligada à descoberta relativamente rápida de *boas soluções* principalmente para os problemas que precisam ser resolvidos em tempo real. Em um processo de pesquisa robusto, os processos de exploração e aproveitamento devem ser realizados em conjunto. No algoritmo ABC, enquanto observadores e abelhas empregadas realizam o processo de exploração no espaço de busca, os batedores controlam o processo de exploração.

3 experimentos

3.1 Funções de benchmark

Uma função é multimodal se tiver dois ou mais ótimos locais. Uma função de variáveis é separável se puder ser reescrita como uma soma de funções de apenas uma variável [18] A separabilidade está intimamente relacionada ao conceito de epistasia ou inter-relação entre as variáveis da função. O problema é ainda mais difícil se a função também for multimodal. O processo de busca deve ser capaz de evitar as regiões em torno dos mínimos locais para se aproximar, na medida do possível, do ótimo global. O caso mais complexo aparece quando os ótimos locais são distribuídos aleatoriamente no espaço de busca. A dimensionalidade do espaço de busca é outro fator importante na complexidade do problema [19] Um estudo do problema de dimensionalidade e suas características foi realizado por Friedman [20] A fim de comparar o desempenho do ABC proposto com PSO, PS-EA e GA, usamos cinco funções de benchmark clássicas, conforme apresentado na Ref. [21]

A primeira função é a função Griewank, cujo valor é 0 em seu mínimo global (0,0, ..., 0) (3,1) O intervalo de inicialização para a função é [- 600.600]. A função de Griewank tem um termo de produto que introduz a interdependência entre as variáveis. O objetivo é superar o fracasso das técnicas que otimizam cada variável de forma independente. Os ótimos da função Griewank são regularmente distribuídos. Como o número de ótimos locais aumenta com a dimensionalidade, essa função é fortemente multimodal. A multimodalidade desaparece para dimensionalidades suficientemente altas ($n > 30$) e o problema parece unimodal.

$$f_1(x) = \frac{1}{4.000}\left(\sum_{i=1}^{D}(x_{eu}^2)\right) - \prod_{i=1}^{D}\cos\left(\frac{x_{eu}}{\sqrt{eu}}\right) + 1 \qquad (3.1)$$

A segunda função é a função Rastrigin, cujo valor é 0 em seu mínimo global (0,0,..., 0) (3,2) O intervalo de inicialização para a função é [- 15,15]. Esta função é baseada na função de esfera com a adição de modulação cosseno para produzir muitos

mínimo. Assim, a função é multimodal. As localizações dos mínimos são regularmente distribuídas. A parte difícil de encontrar soluções ótimas para essa função é que um algoritmo de otimização pode facilmente ser preso em um ótimo local em seu caminho em direção ao ótimo global.

$$f_2(x) = \sum_{i=1}^{D} (x_{eu}^2 - 10\cos(2\pi x_{eu}) + 10). \tag{3,2}$$

A terceira função é a função de Rosenbrock cujo valor é 0 em seu mínimo global (1,1, ..., 1) (3,3) O intervalo de inicialização para a função é [- 15,15]. O ótimo global está dentro de um vale plano longo, estreito e parabólico. Como é difícil convergir para o ótimo global, as variáveis são fortemente dependentes e os gradientes geralmente não apontam para o ótimo, este problema é usado repetidamente para testar o desempenho dos algoritmos de otimização.

$$f_3(x) = \sum_{i=1}^{D} 100(x_{eu}^2 - x_{eu+1})^2 + (1 - x_{eu})^2 \tag{3,3}$$

A quarta função é a função Ackley, cujo valor é 0 em seu mínimo global (0,0,..., 0) (3,4) O intervalo de inicialização para a função é [- 32.768,32.768]. Ackley tem um termo exponencial que cobre sua superfície com vários mínimos locais. A complexidade desta função é moderada. Um algoritmo que usa apenas a descida mais íngreme do gradiente ficará preso em um ótimo local, mas qualquer estratégia de busca que analise uma região mais ampla será capaz de cruzar o vale entre os ótimos e obter melhores resultados. Para obter bons resultados para esta função, a estratégia de busca deve combinar os componentes exploratório e exploratório de forma eficiente.

$$f_4(x) = 20 + e - 20\,e^{\left(-0{,}2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_{eu}^2}\right)} - e^{\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_{eu})} \tag{3.4}$$

A quinta função é a função de Schwefel, cujo valor é 0 em seu mínimo global (420,9867, 420,9867,..., 420,9867) (3,5) O intervalo de inicialização para a função é [- 500, 500]. A superfície da função de Schwefel é composta por um grande número de picos e vales. A função tem um segundo melhor mínimo longe do mínimo global, onde muitos algoritmos de pesquisa estão presos. Além disso, o mínimo global está próximo dos limites do domínio.

$$f_5(x) = D * 418{,}9829 + \sum_{i=1}^{D} -x_{eu}\,\text{pecado}(\sqrt{|x_i|}). \tag{3,5}$$

3.2 Configurações para algoritmos

Os parâmetros de controle comuns dos algoritmos são o tamanho da população e o número de geração máxima. Nos experimentos, o número máximo de gerações foi 500, 750 e 1.000 para as dimensões 10, 20 e 30, respectivamente; e o tamanho da população era 125 como na Ref. [21] Outros parâmetros de controle dos algoritmos e os esquemas usados na Ref. [21] são apresentados a seguir; e também os valores desses parâmetros de controle empregados para GA, PSO e PS-EA na Ref. [21] são apresentados.

3.2.1 Configurações GA

O esquema GA usado apresentado na Ref. [21] é o seguinte: crossover uniforme de ponto único com a taxa de 0,95, mecanismo de seleção aleatória, mutação gaussiana com a taxa de 0,1 e função de ajuste de classificação linear são empregados. Um cromossomo filho é adicionado à população no esquema de produção infantil.

3.2.2 Configurações de PSO

As equações PSO são fornecidas com as expressões em (3,6) e (3,7)

$$\upsilon(t+1) = \omega\upsilon(t) + \varphi_1 \,\mathrm{rand}(0,1)(p(t) - x(t)) + \varphi_2 \,\mathrm{rand}(0,1)(g(t) - x(t)), \tag{3,6}$$

$$x(t+1) = x(t) + \upsilon(t+1), \tag{3,7}$$

Onde ω é o peso de inércia adicional, que varia de 0,9 a 0,7 linearmente com as iterações. Os fatores de aprendizagem, φ_1 e φ_2 são definidos como 2. Os limites superior e inferior para υ, $(\upsilon_{\min}, \upsilon_{\max})$ são definidos como os limites máximos superior e inferior de x, ie $(\upsilon_{\min}, \upsilon_{\max}) = (x_{\min}, x_{\max})$. Se a soma das acelerações causasse a velocidade nessa dimensão $\upsilon(t+1)$, para exceder $\upsilon_{\min}$ ou $\upsilon_{\max}$, então a velocidade nessa dimensão $\upsilon(t+1)$, é limitado a $\upsilon_{\min}$ ou $\upsilon_{\max}$, respectivamente [21]

3.2.3 Configurações PS-EA

O Algoritmo Evolucionário Inspirado por Enxame de Partículas (PS-EA) é um algoritmo hibridizado que combina conceitos de PSO e EA. O principal módulo do PS-EA é o Mecanismo de Autoatualização (SUM), que faz uso da Árvore de Probabilidade de Herança (PIT) para fazer a operação de atualização de cada indivíduo da população. Um Ajustador de probabilidade de herança dinâmica (DIPA) é incorporado em SUM para ajustar dinamicamente as probabilidades de herança em PIT com base na taxa de convergência ou status do algoritmo em uma iteração particular [21] O objetivo de definir as probabilidades de herança iniciais de forma diferente é observar se o DIPA funciona corretamente para ajustar as probabilidades de herança de volta para que garanta uma pesquisa eficaz. Um conjunto inviável de probabilidades de herança inicial é usado para testar o desempenho do módulo DIPA do PS-EA [21]

3.2.4 Configurações ABC

O algoritmo ABC tem alguns parâmetros de controle: Número máximo de ciclos (MCN) é igual ao número máximo de geração e o tamanho da colônia é igual ao tamanho da população, ou seja, 125, como no estudo apresentado na Ref. [21] A porcentagem de abelhas observadoras era de 50% da colônia, as abelhas empregadas eram 50% da colônia e o número de abelhas batedoras foi selecionado como um só. O aumento do número de batedores incentiva a exploração à medida que o aumento de curiosos em uma fonte de alimento aumenta a exploração. Cada um dos experimentos foi repetido 30 vezes com diferentes sementes aleatórias. Os valores médios das funções das melhores soluções encontradas pelos algoritmos para diferentes dimensões foram registrados. A média e os desvios padrão dos valores da função obtidos pelo ABC, GA, PSO, PS-EA são dados na Tabela 1 .

tabela 1 Resultados obtidos pelos algoritmos GA, PSO, PS-EA e ABC

f	Alg.	GA [21]		PSO [21]		PS-EA [21]		ABC1		ABC2	
	Dim	Mau	SD	Mau	SD	Mau	SD	Mau	SD	Mau	SD
*f*1	10	0,050228	0,029523	0,079393	0,033451	0,222366	0,0781	0,00087	0,002535	0,000329	0,0018
	20	1.0139	0,026966	0,030565	0,025419	0,59036	0,2030	2.01E-08	6,76E-08	0	0
	30	1.2342	0,11045	0,011151	0,014209	0,8211	0,1394	2.87E-09	8.45E-10	0	0
*f*2	10	1,3928	0,76319	2,6559	1,3896	0,43404	0,2551	0	0	0	0
	20	6.0309	1,4537	12,059	3,3216	1.8135	0,2551	1.45E-08	5.06E-08	0	0
	30	10,4388	2.6386	32.476	6,9521	3.0527	0,9985	0,033874	0,181557	0	0
*f*3	10	46,3184	33,8217	4,3713	2,3811	25,303	29,7964	0,034072	0,045553	0,012522	0,01263
	20	103,93	29,505	77.382	94,901	72,452	27,3441	0,13614	0,132013	0,014458	0,010933
	30	166,283	59,5102	402,54	633,65	98,407	35.5791	0,219626	0,152742	0,020121	0,021846
*f*4	10	0,59267	0,22482	9,8499E-13	9.6202E-13	0,19209	0,1951	7,8E-11	1.16E-09	4.6E-11	5.4E-11
	20	0,92413	0,22599	1.1778E-6	1.5842E-6	0,32321	0,097353	1.6E-11	1.9E-11	0	1E-12
	30	1.0989	0,24956	1.4917E-6	1.8612E-6	0,3771	0,098762	3E-12	5E-12	0	0
*f*5	10	1,9519	1,3044	161,87	144,16	0,32037	1.6185	1.27E-09	4E-12	1.27E-09	4E-12
	20	7,285	2,9971	543,07	360,22	1,4984	0,84612	19.83971	45,12342	0,000255	0
	30	13,5346	4,9534	990,77	581,14	3.272	1.6185	146,8568	82,3144	0,000382	1E-12

GA, PSO, PS-EA e ABC1 indicam os resultados obtidos após 500, 750 e 1.000 ciclos com uma população de 125 indivíduos, enquanto ABC2 indica os resultados obtidos após 1.000, 1.500 e 2.000 ciclos

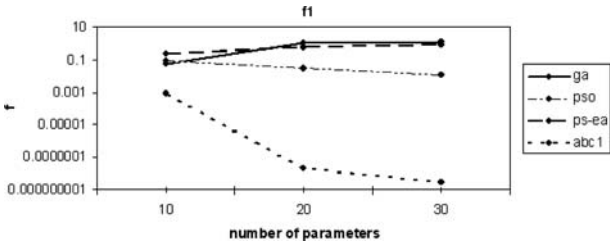

Figura 1 Os melhores valores médios obtidos para $f1$ por GA, PSO, PS-EA e ABC1 após 500, 750 e 1.000 ciclos para dimensões 10, 20 e 30

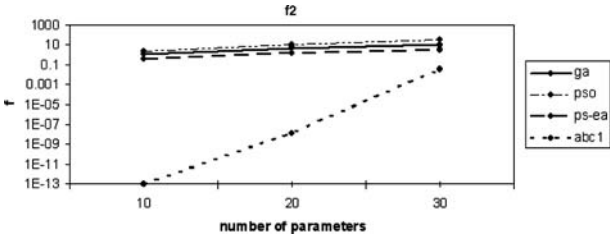

Figura 2 Os melhores valores médios obtidos para $f2$ por GA, PSO, PS-EA e ABC1 após 500, 750 e 1.000 ciclos para dimensões 10, 20 e 30

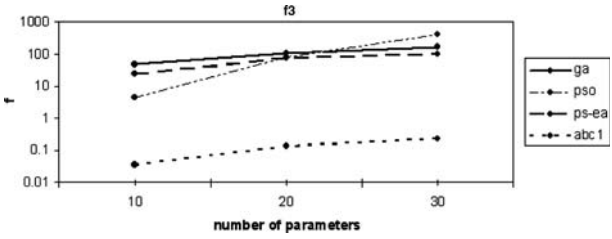

Fig. 3 Os melhores valores médios obtidos para $f3$ por GA, PSO, PS-EA e ABC1 após 500, 750 e 1.000 ciclos para dimensões 10, 20 e 30

A fim de mostrar o desempenho do algoritmo ABC de forma mais clara, as representações gráficas dos resultados na Tabela 1 são reproduzidos nas Figs. 1 - 5 . Mesa 1 e os gráficos indicam que, embora o desempenho do algoritmo PS-EA se deteriore na otimização de funções difíceis, como as funções Griewank e Akley, o ABCalgorithm mostra melhor desempenho nelas. No entanto, PS-EA e GA superam o algoritmo ABC apenas na função de Schwefel para as dimensões 20 e 30 por essas configurações. Depois de obter resultados para 500, 750 e 1.000 ciclos, o algoritmo ABC foi executado para 1.000, 1.500 e 2.000 ciclos em vez de 500, 750 e 1.000 ciclos para a dimensão 10, 20 e 30, respectivamente, para ver se o algoritmo ABC pode melhorar as soluções para esse problema quando o valor de MCN é aumentado. Os resultados produzidos neste caso são mostrados na coluna ABC2 da Tabela 1 . Como pode ser visto nos resultados desta coluna, o algoritmo ABC também pode convergir para o mínimo da função de Schwefel. Em outras palavras, isso prova que o algoritmo ABC tem a capacidade de sair de um mínimo local no espaço de busca e encontrar o mínimo global. No algoritmo ABC, enquanto o processo de exploração realizado por olheiros arti fi ciais é bom para

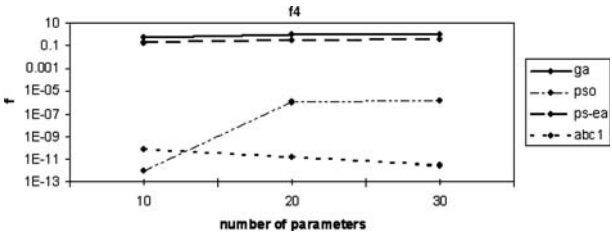

Fig. 4 Os melhores valores médios obtidos para $f4$ por GA, PSO, PS-EA e ABC1 após 500, 750 e 1.000 ciclos para dimensões 10, 20 e 30

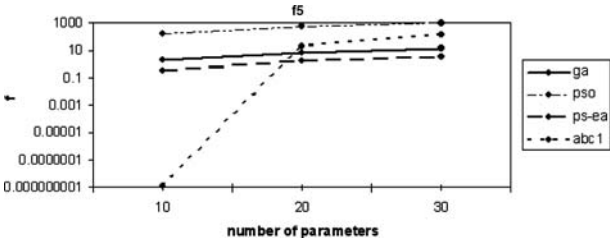

Fig. 5 Os melhores valores médios obtidos para $f5$ por GA, PSO, PS-EA e ABC1 após 500, 750 e 1.000 ciclos para dimensões 10, 20 e 30

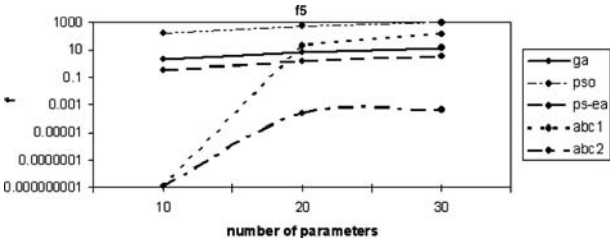

Fig. 6 Os melhores valores médios obtidos para $f5$ byGA, PSO, PS-EA, ABC1 após 500, 750 e 1.000 ciclos e ABC2 após 1.000, 1.500 e 2.000 ciclos

otimização, o processo de exploração gerenciado por observadores arti fi ciais e abelhas empregadas é muito eficiente para a otimização local. Portanto, o algoritmo ABC é bastante bem-sucedido na otimização de funções multivariáveis e multimodais. Na Fig. 6 , O significativo melhores valores de GA, PSO, PS-EA, ABC1 e ABC2 são dados graficamente para f_5 quando o valor de MCN é definido como 1.000, 1.500 e 2.000. Uma versão demo do Arti fi cial O algoritmo Bee Colony foi preparado e é apresentado em sua página inicial [22]

4. Conclusão

Este artigo comparou o desempenho do ABC com o do GA, PSO e PS-EA, que também são algoritmos de inteligência de enxame e baseados em população como o algoritmo ABC. Para demonstrar o desempenho do algoritmo ABC, os algoritmos PSO, PS-EA, GA e ABC foram testados em cinco funções de benchmark numéricas de alta dimensão que possuem multimodalidade. A partir dos resultados da simulação, concluiu-se que

o algoritmo proposto tem a capacidade de sair de um mínimo local e pode ser usado com eficiência para otimização multivariável de função multimodal. Existem várias questões que permanecem como escopos para estudos futuros, como a investigação do efeito dos parâmetros de controle sobre o desempenho do algoritmo ABC e a velocidade de convergência do algoritmo.

Referências

1 Pham, DT, Karaboga, D .: Intelligent Optimization Techniques. Springer, London (2000) Holland, JH:
2 Adaptation in Natural and Arti fi cial Systems. University of Michigan Press, Ann Arbor, MI (1975)

3 De Castro, LN, Von Zuben, FJ: Arti fi cial Immune Systems. Parte I. Teoria Básica e Aplicações. Relatório Técnico nº Rt Dca 01/99, Feec / Unicamp, Brasil (1999)
4 Kennedy, J., Eberhart, RC: Particle swarm optimization. In: Proceedings of the 1995 IEEE International Conference on Neural Networks, vol. 4, pp. 1942–1948. IEEE Service Center, Piscat away (1995)

5 Fukuyama, Y., Takayama, S., Nakanishi, Y., Yoshida, H .: Otimização de enxame de partículas para potência reativa e controle de tensão em sistemas de energia elétrica. In: Proceedings of the Genetic and Evolutionary Computation Conference, pp. 1523–1528. Orlando, Flórida, EUA (1999) Tereshko, V .: Modelo de
6 reação-difusão do comportamento de forrageamento de uma colônia de abelhas. In: Schoenauer M. (ed.) Parallel Problem Solving from Nature VI. Lecture Notes in Computer Science, vol. 1917, pp. 807-816, Springer, Berlin (2000)
7 Tereshko, V., Lee, T .: Como os padrões de mapeamento de informações determinam o comportamento de forrageamento de uma colônia de abelhas. Open Syst. Inf. Dyn. **9,** 181-193 (2002)
8 Tereshko, V., Loengarov, A .: Tomada de decisão coletiva na dinâmica de forrageamento de abelhas. Comput. Inf. Sys. J., **9 (** 3), 1–7 (2005)
9 Teodorović´, D .: Modelagem de Transporte por Sistemas Multi-Agente: Uma Abordagem de Intellgence de Enxame, Transporte. Plano. Technol. **26 (** 4), 289-312 (2003)
10 Lucic, P., Teodorović´, D .: Modelagem de Transporte: Uma Abordagem de Vida Arti fi cial. ICTAI, pp. 216-223. Washington DC (2002)
11 Teodorović´, D., Dell'Orco, M .: Otimização de colônias de abelhas - uma abordagem de aprendizagem cooperativa para problemas de transporte complexos. In: Proceedings of the 10th EWGT Meeting, Poznan, 13–16 de setembro de 2005
12 Drias, H., Sadeg, S., Yahi, S .: Enxame de abelhas cooperativas para resolver o problema de capacidade máxima ponderada de satisfação, inteligência computacional e sistemas bioinspirados. In: Proceedings of the 8th International Workshop on Arti fi cial Neural Networks, IWANN 2005, Vilanova i la Geltr, Barcelona, Spain, 8–10 de junho de 2005
13 Benatchba, K., Admane, L., Koudil, M .: Usando abelhas para resolver um problema de mineração de dados expresso como um max-sat one, inteligência artificial e aplicações de engenharia do conhecimento: uma abordagem bioinspirada. In: Proceedings of the First International Work-Conference on the Interplay Between Natural and Arti fi cial Computation, IWINAC 2005, Las Palmas, Ilhas Canárias, Espanha, 15-18 de junho de 2005
14 Wedde, HF, Farooq, M., Zhang, Y .: BeeHive: um algoritmo de roteamento tolerante a falhas e fi ciente inspirado pelo comportamento das abelhas, colônia de formigas, otimização e inteligência de enxame. In: Proceedings of the 4th International Workshop, ANTS 2004, Brussels, Belgium, 5-8 September 2004
15 Yang, XS: Otimizações de engenharia por meio de algoritmos de abelhas virtuais inspirados na natureza. Lecture Notes in Computer Science, pp. 317–323. Springer, GmbH (2005)
16 Karaboga, D .: Uma ideia baseada em enxames de abelhas para otimização numérica. Relatório Técnico - TR06, Universidade Erciyes, Faculdade de Engenharia, Departamento de Engenharia da Computação, 2005
17 Basturk, B., Karaboga, D .: Um algoritmo de colônia de abelhas arti fi cial (ABC) para otimização de função numérica. In: Proceedings of the IEEE Swarm Intelligence Symposium 2006, Indianapolis, Indiana, EUA, 12–14 de maio de 2006
18 Hadley, G .: Programação não linear e dinâmica. Addison Wesley, Reading, MA (1964) Boyer, DO, Martnez,
19 CH, Pedrajas, NG: Crossover Operator for Evolutionary Algorithms Based on Population Features. http://www.cs.cmu.edu/afs/cs/project/jair/pub/volume24/ortizbo- yer05a-html / Ortiz-Boyer.html.

20. Friedman, JH: Uma visão geral da aprendizagem preditiva e aproximação de função. From Statistics to Neural Networks, Theory and Pattern Recognition Applications, OTAN ASI Series F, vol. 136, pp. 1-61. Springer, Berlim (1994)
21. Srinivasan, D., Seow, TH: Evolutionary Computation, CEC '03, 8-12 de dezembro de 2003, 4 (2003), Canberra, Australia, pp. 2292-2297.
22. http://mf.erciyes.edu.tr/abc