Algoritmo de vaga-lumes com predação em sistema multi-processado

Eduardo Fávero Pacheco da Luz[1], **José Carlos Becceneri**[2], **Haroldo Fraga de Campos Velho**[2]

[1]Programa de Doutorado em Computação Aplicada – CAP
Instituto Nacional de Pesquisas Espaciais – INPE

[2]Laboratório Associado de Computação e Matemática Aplicada – LAC
Instituto Nacional de Pesquisas Espaciais – INPE

{eduardo.luz, becce, haroldo}@email.br

Abstract. *A modification of the optimization method inspired in the behaviour of fireflies is obtained with the introduction of a predation operator. This new algorithm is implemented in a multi-processed environment with shared memory and OpenMP is used to implement the parallel computation. Analysis of the performance shows a super-linear efficiency for the processing gain curve (speed-up), maintaining the numerical capacity of the algorithm so search for solutions.*

Resumo. *Uma modificação do método de otimização inspirado no comportamento natural de vaga-lumes é realizada com a introdução de um operador de predação. Esta nova forma do algoritmo é implementada num ambiente multi-processado de memória compartilhada e o padrão OpenMP é empregado para computação paralela. A análise de desempenho mostram uma eficiência super-linear para a curva de ganho de processamento (speed-up), mantendo a capacidade numérica do algoritmo para a busca de soluções.*

Palavras-chave: *metaheurística, vaga-lumes, bio-inspirado, otimização.*

1. Introdução

O desenvolvimento de aplicações científicas que demandam uma grande capacidade computacional se valeu de um avanço significativo com o advento da última evolução do hardware computacional, a popularização de multi-processadores, que são processadores com mais de um núcleo de equivalente poder computacional aos desktops de alguns anos atrás.

Este trabalho apresenta os resultados obtidos pelo desenvolvimento de um método de otimização estocástico baseado no comportamento de vaga-lumes, adicionado de um mecanismo de predação, e implementado em um ambiente paralelo, multi-processado, de memória compartilhada.

2. Algoritmo de vaga-lumes

O algoritmo de vaga-lumes (*Firely Algorithm*, FA) foi proposto por Xin-She Yang na Universidade de Cambridge em 2007. Este algoritmo é baseado na característica bioluminescente de vaga-lumes, insetos coleópteros notórios por suas emissões luminosas. A biologia ainda não tem um conhecimento completo para determinar todas as utilidades

que esta luminescência pode trazer ao vaga-lume, mas pelo menos três funções já foram identificadas [Yang 2010]: (i) como uma ferramenta de comunicação e atração para potenciais parceiros na reprodução; (ii) como uma isca para atração de eventuais presas para o vaga-lume; e (iii) como um mecanismo de alerta para potenciais predadores: lembrando-os de que vaga-lumes tem um "gosto amargo".

A intensidade de emissão de luz por parte de um vaga-lume é proporcional à função objetivo, i.e., $I(x) \propto J(x)$, porém a intensidade de luz percebida por um vaga-lume decai em função da distância entre os vaga-lumes, dada a absorção da luz pelo meio. Logo, a intensidade percebida por um vaga-lume é dada por: $I(r) = I_0 e^{\gamma r^2}$, em que I_0 é a intensidade da luz emitida; r é a distância Euclidiana entre os vaga-lumes i e j, sendo i o vaga-lume mais brilhante e j o vaga-lume menos brilhante; e γ é o parâmetro de absorção da luz pelo meio. Desta maneira o fator de atratividade β pode ser formulado como:

$$\beta = \beta_0 e^{-\gamma r^2}, \tag{1}$$

em que β_0 é a atratividade para uma distância $r = 0$, e pode ser fixo em $\beta_0 = 1$.

A movimentação em um dado passo de tempo t de um vaga-lume i em direção a um melhor vaga-lume j é definida como:

$$x_i^t = x_i^{t-1} + \beta \left(x_j^{t-1} - x_i^{t-1} \right) + \alpha \left(rand - \frac{1}{2} \right), \tag{2}$$

em que, o segundo termo do lado direito da equação insere o fator de atratividade β enquanto que o terceiro termo, regulado pelo parâmetro α, regula inserção de certa aleatoriedade no caminho percorrido pelo vaga-lume, $rand$ é um número aleatório entre 0 e 1.

2.1. Algoritmo de vaga-lumes com predação

A inspiração biológica advinda da observação do comportamento dos vaga-lumes pode ser incrementada com a adição de um mecanismo de seleção natural baseado na predação dos indivíduos menos aptos.

Desta maneira, este trabalho apresenta esta variante do *Firefly Algorithm* adicionado de um componente de seleção natural baseado em predação. Esta implementação pode seguir o esquema apresentado na Fig. 1.

Dois novos parâmetros são necessários para a implementação deste esquema de seleção natural: a quantidade de vaga-lumes que não serão predados, representado por n_{selNat}; e a frequência com que o esquema de seleção natural será aplicada, representado por $tSelNat$.

Os novos vaga-lumes são reinicializados em novas posições aleatórias dentro do espaço de buscas e o algoritmo itera convencionalmente até o seu critério de parada final ou até que outro passo de seleção natural seja ativado.

Os resultados observados demonstram a viabilidade da utilização deste esquema. Esta característica permite que o algoritmo escape de ótimos locais em um tempo menor do que sua versão canônica.

Firefly Algorithm com Seleção Natural

Início
Definir a função objetivo $J(x)$, $x = (x_1, \ldots, x_d)^T$
Definir os parâmetros $n, \alpha, \gamma, \beta_0, MaxGeracoes, n_{selNat}, tSelNat$
Gerar a população inicial de vagalumes x_i $(i = 1, 2, \ldots, n)$
Para $t = 1$ até $MaxGeracoes$
Se $tSelNat$
Para $i = n_{selNat} + 1$ até n
Gera um novo vagalume
Fim-Para
Fim-Se
Calcular a intensidade da luz I_i para x_i proporcionalmente a $J(x_i)$
Para $i = 1$ até n
Calcular o fator de atratividade β de acordo com $e^{-\gamma r^2}$
Mover o vagalume i em direção aos vagalumes mais brilhantes
Verificar se o vagalume está dentro dos limites
Fim-Para
Fim-Para
Pós-processar e visualizar os resultados
Fim

Figura 1. Pseudocódigo para o *Firefly Algorithm* com Seleção Natural.

3. Resultados

Para a análise dos resultados, foram utilizadas as funções bidimensionais (2D) de *benchmark* de Ackley, Beale e Rastrigin. Os resultados aqui demonstrados são a média de 10 (dez) experimentos computacionais independentes. Para as funções de Ackley e Rastrigin, foram utilizados 20 vaga-lumes (soluções candidatas), destes, 5 são mantidos após o procedimento de predação, que o corre a cada 500 iterações. O critério de parada é de 100.000 iterações. Para a função de Beale, o critério de parada é o de precisão para o valor da função objetivo, neste caso 1×10^{-8}.

A Tabela 1 apresenta os resultados numéricos para a solução com *Firefly Algorithm* canônico e com o *Firefly Algorithm with Predation* e implementação com OpenMP. As funções de Ackley e Rastrigin tem seu ótimo global em $\{0; 0\}$ e a função de Beale tem seu ótimo em $\{3,0; 0,5\}$. A linha de FO apresenta o resultado obtido para a média da função objetido e a linha DP apresenta o desvio padrão.

Tabela 1. Resultados numéricos obtidos pelo FA canônico e o FAP com OpenMP.

Função		FA	FAP
Ackley	FO	$\{0,77; 0,21\}$	$\{-3,04 \times 10^{-5}; -3,2 \times 10^{-5}\}$
	DP	$\{1,37; 1,53\}$	$\{1,17 \times 10^{-4}; 1,68 \times 10^{-4}\}$
Beale	FO	$\{2,99; 0,50\}$	$\{3,00; 0,50\}$
	DP	$\{3,75 \times 10^{-4}; 1,01 \times 10^{-4}\}$	$\{3,45 \times 10^{-4}; 9,78 \times 10^{-5}\}$
Rastrigin	FO	$\{3,45 \times 10^{-3}; -0,10\}$	$\{6,30 \times 10^{-3}; -2,87 \times 10^{-3}\}$
	DP	$\{1,24; 1,18\}$	$\{1,83 \times 10^{-2}; 1,03 \times 10^{-2}\}$

A função de Beale apresentou resultados equivalentes, porém a condição parada foi satisfeita após uma média de 239.012 avaliações para o FA e 218.570 avaliações para o FAP.

A Tabela 2 apresenta o *speed up* e a eficiência obtida pela análise do algoritmo de vagalumes com predação para a função de Ackley, em 250.000 iterações e 100 vaga-lumes. O *speed up* é super-linear para dois e três processadores.

Tabela 2. *Speed up* e eficiência do FAP com OpenMP.

Threads	*Speed up*	Eficiência
2	2,97	1,48
3	3,09	1,03
4	3,79	0,94

A Figura 2 apresenta a curva de *speed up* baseada nos dados da Tab. 1.

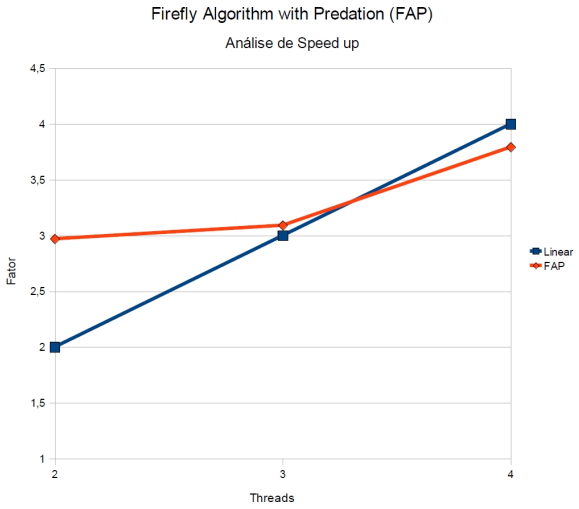

Figura 2. Fator de *speed up* obtido para 2, 3 e 4 processadores executando o FAP.

4. Conclusões

A nova variante com predação mostra uma melhora no desempenho do algoritmo, permitindo um escape mais rápido de ótimos locais. Já a sua implementação em ambiente de memória paralela reduziu o tempo de execução do algoritmo em alguns casos levando a uma curva de *speed up* super-linear.

Próximos passos incluem a obtenção de resultados para a solução de problemas inversos com o uso desta nova implementação e um maior estudo com o objetivo de compreender a obtenção do resultado super-linear. Testes com um maior número de processadores também serão levados em conta.

5. Agradecimentos

Os autores agradecem o apoio da CAPES pelo financiamento na forma de bolsa de doutorado. Também agradecem à CAP/INPE e ao LAC/INPE.

Referências

Yang, X.-S. (2010). *Nature-Inspired Metaheuristic Algorithms*. Luniver Press, Frome, UK, second edition edition.