PSO+: Algoritmo de Otimização Baseado em Inteligência de Enxame de Partículas para Problemas com Restrições Lineares e Não Lineares

Manoela Kohler[1], Marley Vellasco[1], Ricardo Tanscheit[1]

[1]Departamento de Engenharia Elétrica, Pontifícia Universidade Católica do Rio de Janeiro, RJ, Brasil

manoela@ele.puc-rio.br, marley@ele.puc-rio.br, ricardo@ele.puc-rio.br

Resumo. Este trabalho propõe um novo algoritmo baseado em otimização por enxame de partículas para solucionar problemas com restrições lineares e não lineares. Um operador de crossover e um novo método de atualização das partículas, chamado de Pontos de Apoio, foram desenvolvidos de forma a sempre garantir soluções totalmente válidas e uma melhor cobertura do espaço de busca válido, respectivamente. Além disso, uma heurística de inicialização de partículas válidas é proposta e aplicada a benchmarks com restrições de lineares e não lineares. O algoritmo foi testado em 13 benchmarks propostos em uma competição de algoritmo de otimização. Resultados experimentais mostram que o algoritmo é competitivo já que consegue aumentar a eficiência do PSO e melhorar a velocidade de convergência.

Palavras-chave: Otimização por Enxame de Partículas; Inteligência de Enxames; Restrições Lineares; Restrições Não Lineares.

1 Introdução

Muitos problemas de otimização envolvem uma série de restrições que as soluções de apoio à decisão precisam satisfazer. O objetivo da otimização restrita é buscar soluções viáveis, ou seja, válidas com melhores valores objetivos [1]. Geralmente, um problema de otimização restrita pode ser descrito da seguinte maneira:

Encontre x que minimize f(x)

$$\text{Sujeito à } g_i(\boldsymbol{x}) \leq 0, i = 1,2,\dots,n, \qquad h_j(\boldsymbol{x}) = 0, j = 1,2,\dots,p, \tag{1}$$

onde $\boldsymbol{x}$=[$x1$,$x2$,…,xd] denota o vetor de solução de decisão, n é o número de restrições de desigualdade e p é o número de restrições de igualdade. Em uma prática comum, uma restrição de igualdade $h_j(x) = 0$ pode ser substituída por um par de

restrições de desigualdade $h_j(x) \leq \delta$ and $h_j(x) \geq \delta$ (onde δ é uma pequena tolerância). Portanto, todas as restrições podem ser transformadas em $N = n + 2p$ restrições de desigualdade. Uma solução $\boldsymbol{x}$ é chamada de solução viável se satisfizer todas as restrições.

Com base na simulação de modelos sociais simplificados, como a movimentação de um bando de pássaros e de cardumes de peixes, o PSO foi introduzido pela primeira vez em [2] como uma nova técnica de computação evolutiva com o mecanismo de melhoria individual, cooperação populacional e competição. O PSO tem sido usado em vários problemas de otimização nos últimos anos [3–7]. No PSO, múltiplos indivíduos candidatos, chamados de partículas, coexistem e colaboram simultaneamente, e a posição de cada partícula denota um vetor de decisão para o problema original. A trajetória de cada partícula no espaço de busca é ajustada dinamicamente atualizando sua velocidade, de acordo com sua própria experiência de "voo", bem como com a experiência de partículas vizinhas (construídas através do rastreamento e memorização da melhor posição encontrada). Portanto, o PSO combina a técnica de busca local (pela experiência da própria partícula) e o método de busca global (pela experiência vizinha) para equilibrar a exploração global e local, de forma a finalmente atingir o ótimo global [8].

O método de penalização é a técnica de manipulação de restrições mais popular devido à sua simplicidade e facilidade de implementação. As violações das restrições das soluções são incorporadas na função objetivo, de modo que os problemas restritos originais são transformados em problemas sem restrições. O algoritmo PSO multiagente evolutivo apresentado em [3] resolve fluxos de energia ótimos com restrições de segurança que geram termos de penalidade para a função objetivo. Em [9], um novo PSO para resolver problemas de otimização restrita é proposto. Um fator de penalidade é introduzido para garantir que a aptidão das soluções inválidas seja sempre pior do que qualquer solução válida. Xu et al. [10] propõem três algoritmos PSO diferentes para melhorar a cobertura das redes de câmeras. Um dos algoritmos adiciona uma penalidade adaptativa à cobertura quando uma câmera viola a restrição de distância.

A abordagem do operador de viabilidade define um operador para tornar a solução válida no caso de a partícula sair dos limites de restrições. Em [6], os autores propõem um operador para limitar o vetor de velocidade do PSO. Eles propõem três métodos para lidar com as restrições de domínio: *Absorbing Walls* – quando uma partícula atinge o limite do espaço da solução em uma das dimensões, a velocidade naquela dimensão é zerada e a partícula é eventualmente trazida de volta ao espaço da solução válido. No sentido do limite, as "paredes" absorvem a energia das partículas tentando escapar do espaço de solução. *Reflecting Walls* – quando uma partícula atinge o limite em uma das dimensões, o sinal da velocidade naquela dimensão é alterado e a partícula é refletida de volta ao espaço da solução. *Invisible Walls* – as partículas podem "voar" sem qualquer restrição física. No entanto, as partículas que vagam fora do espaço de solução válido têm sua função objetiva avaliada.

Esses métodos funcionam bem com restrições de domínio – quando as restrições são impostas a cada variável individualmente –, mas não podem lidar com restrições de igualdade e desigualdade lineares e não lineares em problemas de otimização.

Hu e Eberhart [11] propõem um método onde a estratégia de preservação de viabilidade é empregada para lidar com restrições. Duas modificações no PSO original são feitas: (i) durante a geração de partículas iniciais válidas, todas as partículas são inicializadas repetidamente até satisfazer todas as restrições; (ii) ao calcular os melhores valores da partícula e do enxame, apenas são consideradas posições válidas. Mostra-se que o PSO resolve a maior parte dos problemas de otimização não linear com restrições não lineares de desigualdade, mas é sempre necessário reinicializar aleatoriamente partículas que não satisfaçam as restrições. Dependendo do problema, isso pode ser demorado e até impraticável. Em [12], o autor propõe uma nova técnica para tratar problemas restritos não-lineares como um problema de otimização biobjetivo: um objetivo é o objetivo original do problema e o outro é o grau de violação de restrições.

Este artigo propõe um novo algoritmo PSO, denominado PSO+, que utiliza um operador de cruzamento aritmético de viabilidade e um método de atualização de partículas para melhorar a cobertura do espaço de busca. Uma heurística de inicialização de enxame também é proposta para acelerar a inicialização de partículas. O algoritmo proposto é adequado para problemas de otimização com restrições lineares e não lineares.

2 PSO Original

PSO é uma técnica de computação evolutiva baseada em inteligência de enxame. A partícula em um enxame é um ponto em um espaço de busca multidimensional. A posição nesse espaço representa uma solução potencial para o problema de otimização. A velocidade do "voo" determina a direção e o passo da busca. A partícula "voa" no espaço de busca a uma velocidade que é ajustada dinamicamente de acordo com suas próprias experiências de "voo" e do enxame (ou vizinhos). A direção e a velocidade da aproximação são constantemente ajustadas ao traçar a melhor posição encontrada até agora pelas próprias partículas e por todo o enxame. As partículas rastreiam as melhores posições atuais, movem-se gradualmente para uma região melhor e, finalmente, tendem a chegar à melhor posição de todo o espaço de busca [13].

No PSO, cada partícula que representa uma solução potencial é mantida dentro de um enxame. Em termos simples, as partículas se movimentam através de um espaço de busca multidimensional onde a posição de cada partícula é ajustada de acordo com as suas experiências e dos seus vizinhos. A posição da partícula *i* no espaço de busca no passo *t* é representada por $x_i(t)$. A posição da partícula é alterada pela adição de um vetor de velocidade $v_i(t+1)$ à posição atual, i.e.

$$x_i(t+1) = x_i(t) + v_i(t+1) \qquad (2)$$

O processo de otimização é conduzido pelo vetor de velocidade, refletindo tanto o conhecimento da partícula (componente cognitivo) como a informação socialmente trocada com a vizinhança da partícula (componente social). O método de otimização baseado no melhor global, que exibe taxas de convergência muito rápidas (Engelbrecht 2007), é usado neste trabalho, sendo assim, a vizinhança para cada partícula é

todo o enxame. Portanto, a informação social é a melhor posição encontrada pelo enxame, referido como *gbest*. A velocidade da partícula *i* é computada como:

$$v_i = w * v_i + c_1 * rand_1 * (pbest_i - x_i) + c_2 * rand_2 * (gbest - x_i) \quad (3)$$

Onde v_i é a velocidade da partícula; w é o peso de inércia; c_1 e c_2 são constantes de aceleração positivas utilizadas para escalar a contribuição dos componentes cognitivos e sociais, respectivamente; $rand_1$ e $rand_2$ são valores aleatórios no intervalo [0,1] amostrados a partir de uma distribuição uniforme contínua; $pbest_i$ indica a melhor posição pessoal associada à partícula *i*; e *gbest* é a melhor posição global do enxame.

3 PSO+: PSO para Problemas com Restrições

Como o algoritmo PSO original não é capaz de lidar com problemas de otimização com restrições, propõe-se, neste trabalho, uma nova implementação do PSO de forma a assegurar a preservação da viabilidade das soluções. As características do PSO+ são apresentadas nas seções a seguir.

3.1 Inicialização da População Válida

Como alguns problemas podem ter restrições de equação de igualdade não linear, que dificultam a inicialização da população com apenas partículas válidas, uma heurística é proposta. Pelo menos uma partícula válida é inicializada aleatoriamente. Em seguida, o resto do enxame é inicializado. Se a próxima partícula for válida, ela é salva; se não o for, procura-se um ponto que seja colinear às soluções válidas e inválidas de forma a trazer a partícula inválida para o espaço de busca válido.

3.2 Pontos de Apoio

Um dos principais motivos que dificultam a localização da solução global é que os sistemas evolutivos não têm capacidade de pesquisar precisamente a área de fronteira entre regiões válidas e não válidas do espaço de busca. Esta habilidade é bastante importante no caso de problemas de otimização com restrições de igualdade ou até mesmo com restrições de desigualdade ativas no ótimo [14]. O número de restrições ativas no ótimo é, naturalmente, importante: se mais restrições estiverem ativas no ótimo, os algoritmos que têm a capacidade de explorar os limites da região válida são mais propensos ao sucesso.

As técnicas de computação evolutiva têm um enorme potencial para incorporar operadores especializados que buscam o limite entre regiões válidas e inválidas de forma eficiente. Como em muitos problemas de otimização com restrições, algumas restrições são ativas no ótimo global, ou seja, na fronteira do espaço válido. Uma vez que restringir o tamanho do espaço de busca em algoritmos evolutivos geralmente é

benéfico, parece natural restringir a busca da solução ao limite da região válida do espaço de busca no contexto de otimização com restrições.

Um novo conceito é introduzido neste trabalho para melhorar a convergência do PSO. Pontos de Apoio são pontos inválidos que são usados para redirecionar partículas com uma certa probabilidade. Eles auxiliam o algoritmo a alcançar soluções que estão na borda do espaço de busca viável. Além disso, também mantêm alguma diversidade no enxame, pois mudam a tendência das partículas de "voar" na direção da melhor partícula do enxame. A equação de redirecionamento é mostrada na equação abaixo:

$$Z = a * X + (1 - a) * F \qquad (4)$$

Onde Z é a nova partícula, a é um número aleatório entre 0 e 1, X é a partícula inválida e F é a partícula válida.

Os pontos de apoio diferenciam-se de partículas do enxame no sentido de que não se movem no espaço de busca – em vez disso, são trocados de vez em quando (o intervalo de tempo é previamente definido) –, não são avaliados e não pertencem a um enxame. Além disso, são inicializados aleatoriamente e não devem pertencer ao espaço de busca válido. Toda vez que uma partícula viável se move no espaço de busca, há uma chance (20%) de ser redirecionada e influenciada por um ponto escolhido aleatoriamente do grupo de pontos de apoio.

3.3 Operador de Crossover Aritmético

Dois enxames são separadamente criados, mas uma evolução em um deles influencia as avaliações de partículas no outro. O primeiro é chamado de enxame de referência e consiste em partículas totalmente válidas, isto é, aquelas que satisfazem todas as restrições do problema. O segundo, chamado de enxame de fronteira, consiste em partículas que devem pelo menos satisfazer as restrições de domínio do problema. O enxame de fronteira atua como um enxame de busca, pois pode ser parcialmente inválido. Esta abordagem é baseada em Genocop III - Algoritmo Genético para Otimização Numérica de Problemas com Restrições [15].

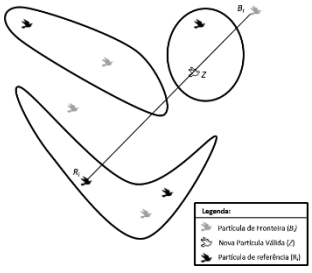

Fig. 1. Procedimento de Crossover

As partículas de referência são diretamente avaliadas no início do processo de otimização, assim como o são as partículas de fronteira válidas. No entanto, partículas de fronteira inválidas são "reparadas" antes da avaliação. Este processo de reparação

consiste na seleção aleatória de uma partícula de referência (*Ri*) e na aplicação de um operador de cruzamento entre a partícula de fronteira inválida (*Bi*) e *Ri* até que seja encontrada uma nova partícula válida (*Z*), que é posteriormente avaliada. Além disso, se a avaliação de *Z* for melhor que a avaliação de *Ri*, esta partícula substitui *Ri* na população de referência. Além disso, *Z* pode substituir *Bi* com certa probabilidade. A **Fig. 1** ilustra o procedimento do operador de crossover aritmético, onde os espaços de busca possíveis estão dentro das formas fechadas.

4 Resultados e Discussões

Foram utilizados treze problemas de *benchmark* de otimização numérica com restrições lineares e não lineares nos experimentos [16]. As principais características desses *benchmarks* estão resumidas na **Tabela 1**.

Tabela 1. Principais características dos *benchmarks* (D é a dimensão do problema, LI, NE, NI são, respectivamente, o número de restrições lineares de desigualdade, restrições não lineares de igualdade e não lineares de desigualdade, e A é o número de restrições ativas no ótimo / número total de restrições)

	Ótimo	***D***	***Min/ Max***	***Tipo de f(x)***	***LI***	***NE***	***NI***	***A***
G01	-15.0000	13	Min	quadrática	9	0	0	6/9
G02	0.803619	20	Max	não linear	1	0	1	1/2
G03	1	10	Max	polinomial	0	1	0	1/1
G04	-30665.539	5	Min	quadrática	0	0	6	2/6
G05	5126.4981	4	Min	cúbica	2	3	0	3/5
G06	-6961.81388	2	Min	cúbica	0	0	2	2/2
G07	24.306291	10	Min	quadrática	3	0	5	6/8
G08	0.095825	2	Max	não linear	0	0	2	0/2
G09	680.630057	7	Min	polinomial	0	0	4	2/4
G10	7049.25	8	Min	linear	3	0	3	3/6
G11	0.75	2	Min	quadrática	0	1	0	1/1
G12	1	3	Max	quadrática	0	0	9^3	$0/9^3$
G13	0.0539498	5	Min	exponential	0	3	0	3/3

4.1 Melhores Resultados do PSO+

A **Tabela 2** apresenta os melhores resultados para o PSO+ com e sem a implementação do conceito de pontos de apoio. Pode-se observar que a implementação e utilização dos pontos de apoio ajuda na convergência do algoritmo em problemas com restrições ativas no ótimo global. As condições de cada otimização feita seguem: Em cada função foram realizados 10 experimentos independentes; foram estipuladas três condições de parada do algoritmo: i) alcançar o mínimo global da função com precisão de 1e-10; ou ii) atingir o número máximo de 30.000 avaliações; ou iii) completar 10 gerações seguidas sem melhora na função objetivo.

A introdução dos pontos de apoio proporcionou a obtenção de melhores resultados para a grande maioria dos problemas. Todas as funções que obtiveram resultados melhores, envolvem restrições ativas nas soluções ótimas globais (o número de restrições ativas no ótimo global para cada função é mostrado na **Tabela 1**. Nas funções

G1, G6, G8, G11 e G12, os dois algoritmos chegaram ao mesmo resultado ótimo (G8 e G12 não possuem restrições ativas no ponto ótimo). Percebe-se também que na implementação com pontos de apoio, a média dos melhores entre todos os experimentos está mais estável, igual ou bem próximo ao ótimo, indicando uma robustez do algoritmo em encontrar o ótimo. Além disso, a diversidade garantida pelo uso dos pontos de apoio permite o PSO+ convergir mais rapidamente para um ótimo global em todas as funções, exceto na G11, em que a média de avaliações do algoritmo sem pontos de apoio foi muito menor (60 vs. 20790 avaliações). O PSO+ conseguiu chegar ao ótimo global ou bem próximo dele em todos os *benchmarks*.

Tabela 2. Melhores Resultados para o PSO+

	Ótimo		***P.A.***	***Melhor***	***Pior***	***Média dos Melhores***	***Desvio Padrão***	***Avaliações***
G1	-15.00	Min	Não	**-15.000**	-15	-12	1.55	570
			Sim	**-15.000**	**-11**	**-15.0**	**0**	**330**
G2	0,8036	Max	Não	0.6274	0.6272	0.5966	1.33e-15	30000
			Sim	**0.7967**	**0.7743**	**0.7967**	**1.59e-6**	**9510**
G3	1	Max	Não	0.6176	0.4969	0.4776	5.55e-16	30000
			Sim	**1.0000**	**0.0939**	**0.6739**	**1.16e-15**	**3900**
G4	-30665,5	Min	Não	-30649.4	-30648.4	-30563.17	2.91e-11	30000
			Sim	**-30665.5**	**-30665.5**	**-30661.09**	**5.09e-11**	**2190**
G5	5126,4981	Min	Não	5936.5	5936.4	6075.77	9.83e-3	2760
			Sim	**5126.49**	**5905.7**	**5126.49**	**1.66e-10**	**2760**
G6	-6961.81	Min	Não	**-6961.8**	-6763.70	-6675.34	3.21e-12	5160
			Sim	**-6961.8**	**-6783.1**	**-6961.7**	**3.18e-1**	**2490**
G7	24.3062	Min	Não	29.44	48.89	56.78	1.5e-6	30000
			Sim	**24.310**	**42.25**	**26.46**	**8.52e-14**	**1860**
G8	0.09582	Max	Não	**0.09582**	0.0910	0.0957	3.12e-2	3570
			Sim	**0.09582**	**0.08544**	**0.09582**	**1.89e-4**	**1200**
G9	680.6300	Min	Não	681.2795	684.49	684.28	1.15e-3	30000
			Sim	**680.6300**	**681.47**	**681.6134**	**9.45e-1**	**30000**
G10	7049.25	Min	Não	7892.7588	9333.63	9099.17	201.91	30000
			Sim	**7050.2600**	**7222.77**	**7050.26**	**8.13e-3**	**28980**
G11	0.75	Min	Não	**0.75**	**0.8027**	**0.75**	**8.88e-2**	**60**
			Sim	**0.75**	1.00	0.75	1.33e-15	20790
G12	1	Max	Não	**1.0000**	0.74	0.9482	8.25e-3	150
			Sim	**1.0000**	**0.52**	**1.0000**	**1.27e-10**	**60**
G13	0.0539	Min	Não	0.0609	0.3157	0.3018	3.82e-3	30000
			Sim	**0.0539**	**0.0622**	**0.0539**	**4.13e-17**	**2000**

4.2 Comparação com outros algoritmos PSO

A **Tabela 3** compara os resultados do PSO+ com os de quatro outras implementações de PSO: PSO_Eb [11], MPSO [5], PSO_CO [9] e COPSO [17]. No PSO_Eb, uma estratégia de preservação de viabilidade é empregada para lidar com restrições. Pode-se perceber que o PSO+ apresenta melhor desempenho – tanto em soluções ótimas quanto em número de simulações até a convergência – para todos os problemas apresentados ao PSO_Eb. O MPSO, é uma implementação para a solução de problemas com restrições que adotam um pequeno tamanho de população. O PSO+ supera o Micro-PSO para os problemas G1 e G4 ao G13, enquanto o Micro-PSO produz resultados superiores para G2 e G3. O PSO_CO, baseia-se no pressuposto de que

qualquer solução válida é melhor do que qualquer solução inválida. Nos problemas G1, G4, G5, G6, G8, G9 e G11, ambos os algoritmos mostram comportamento semelhante. Finalmente, o COPSO, é um PSO que utiliza operadores de perturbação para evitar convergência prematura e uma nova estrutura de vizinhança em anel. Além disso, a técnica de controle de restrições utilizada é baseada na validade das partículas e na soma das violações das restrições. O PSO+ supera esse algoritmo para os problemas G1, G3 ao G6, G8 ao G10, G12 e G13 seja em solução ótima ou em número de avaliações até a convergência ao ótimo. Ambos algoritmos tiveram desempenho bem próximos em G2, G7 e G11.

Tabela 3. Comparação de resultados com diferentes implementações de PSO

	Ótimo Conhecido		***Melhor***	***Média***	***Pior***	***Desvio Padrão***	***Avaliações***
G1	15.0000	PSO+	**-15.000**	**-15.0**	**-11**	**0**	**330**
		PSO_Eb	-15.000	-	-	-	25000
		MPSO	-15.000	-13.2734	-	-	240000
		PSO_CO	-15.000	-15.00	-	-	-
		COPSO	-15.000	-15.000	-15.000	0	90800
G2	0.803619	PSO+	0.7967	0.7967	0.7743	1.59e-6	**9510**
		MPSO	0.8036	0.78	-	-	240000
		COPSO	**-0.8036**	**-0.8013**	**-0.7865**	**4.5e-3**	**142900**
G3	1	PSO+	**1.0000**	**0.6739**	**0.0939**	**1.16e-15**	**3900**
		MPSO	**1.0004**	0.9936	-	-	240000
		COPSO	-1.0000	-1.0000	-1.0000	3.15e-7	315100
G4	-30665.539	PSO+	**-30665.5**	**-30661.09**	**-30665.5**	**5.09e-11**	**2190**
		PSO_Eb	-30665.5	-	-	-	25000
		MPSO	-30665.5	-30665.53	-	-	240000
		PSO_CO	-30665.5	-30665.59	-	-	-
		COPSO	-30665.5	-30665.5	-30665.5	0	59600
G5	5126.4981	PSO+	**5126.49**	**5126.49**	**5905.7**	**1.66e-10**	**2760**
		PSO_Eb	-	-	-	-	-
		MPSO	5126.6467	5495.23	-	-	240000
		PSO_CO	5126.49	5129.30	-	-	-
		COPSO	5126.49	5126.49	5126.49	0	315100
G6	-6961.81388	PSO+	**-6961.8**	**-6961.7**	**-6783.1**	**3.18e-1**	**2490**
		PSO_Eb	-6961.7	-	-	-	25000
		MPSO	-6961.7	-6961.83	-	-	240000
		PSO_CO	-6961.8	-6961.81	-	-	-
		COPSO	-6961.8	-6961.8	-6961.8	0	47100
G7	24.306209	PSO+	24.310	26.46	42.25	8.52e-14	**1860**
		PSO_Eb	24.4420	-	-	-	25000
		MPSO	24.3278	24.6996	-	-	240000
		PSO_CO	-	-	-	-	-
		COPSO	**24.306**	**24.306**	**24.306**	**3.3e-6**	**185500**
G8	0.095825	PSO+	**0.0958**	**0.09582**	**0.08544**	**1.89e-4**	**1200**
		PSO_Eb	0.0958	-	-	-	25000
		MPSO	0.0958	0.0958	-	-	240000
		PSO_CO	0.0958	0.0959	-	-	-
		COPSO	0.0958	0.0958	0.0958	0	3600
G9	680.630057	PSO+	**680.6300**	**681.6134**	**681.47**	**9.45e-1**	**30000**
		PSO_Eb	680.6570	-	-	-	25000
		MPSO	680.6307	680.64	-	-	240000
		PSO_CO	680.6300	680.65	-	-	-
		COPSO	680.6300	680.6300	680.6300	0	69900
G10	7049.25	PSO+	7050.2600	**7050.26**	**7222.77**	**8.13e-3**	28980

		PSO_Eb	7131.0100	-	-	-	25000
		MPSO	7090.4524	7747.63	-	-	240000
		COPSO	**7049.2486**	**7049.2500**	**7049.26**	**3.6e-3**	**167200**
G11	0.75	PSO+	**0.7500**	**0.75**	**1.00**	**1.33e-15**	**20790**
		PSO_Eb	0.7500	-	-	-	25000
		MPSO	**0.7499**	**0.7673**	-	-	**240000**
		PSO_CO	0.7499	0.7499	-	-	-
		COPSO	0.7499	0.7499	0.7499	**0**	315000
G12	1	PSO+	**1.0000**	**1.0000**	**0.52**	**1.27e-10**	**60**
		PSO_Eb	1.0000	-	-	-	25000
		MPSO	1.0000	1.00	-	-	240000
		COPSO	1.0000	1.0000	1.0000	0	400
G13	0.0539	PSO+	**0.0539**	**0.0539**	**0.0622**	**4.13e-17**	**2000**
		MPSO	0.0594	0.81	-	-	240000
		COPSO	0.0539	0.0539	0.0539	2.7e-6	315100

4.3 Comparação com outros algoritmos evolutivos

A **Tabela 4** mostra os resultados obtidos com o PSO+ e com outros algoritmos evolucionários. As três primeiras implementações, RGenIII, GenV e RGenV [18] são implementações revisadas do Genocop III com melhorias no tempo de processamento e em algumas limitações. O PSO+ supera todas estas implementações revisadas tanto em solução quanto em número de avaliações.

Colônia Artificial de Abelhas (ABC, do inglês *Artificial Bee Colony*) é um algoritmo de inteligência de enxames relativamente novo que tem mostrado ter desempenho competitivo quando comparado a outros algoritmos baseados em população. O algoritmo eABC [19] é proposto para resolver problemas de otimização com restrições. No processo de otimização, os indivíduos são inicializados aleatoriamente e a evolução é feita utilizando penalização competitiva. O PSO+ supera em resultados de otimização e número de avaliações os *benchmarks* G1, G3, G4, G6 ao G10, G12 e G13. O desempenho do eABC é ligeiramente melhor em G2. Em G5 e G11 o PSO+ chegou ao ótimo, mas o eABC superou o ótimo, e isso se deve à diferença na tolerância usada para as restrições lineares e não lineares de igualdade. O algoritmo Meta-Heurístico de Vaga-Lume (FA, do inglês *Firefly Algorithm*) é baseado no comportamento natural dos vaga-lumes e no fenômeno de bioluminescência. Para lidar com restrições, Deshpande [20] propôs um método de penalização competitiva. O PSO+ superou FA em todos os benchmarks, exceto G11, onde o resultado do FA foi melhor do que o ótimo, provavelmente devido às tolerâncias estabelecidas para as restrições de igualdade. SSO-C (do inglês *Social Spider Optimization with Constraints*) [21] é baseado na simulação do comportamento cooperativo de aranhas. Nesse algoritmo, os indivíduos emulam um grupo de aranhas que interagem entre si de acordo com as leis de cooperação de colônias cooperativas. A inicialização dos indivíduos é feita de forma aleatória e para lidar com restrições é utilizado o método de penalização competitiva. O PSO+ apresentou resultados melhores tanto em solução da otimização quanto em número de simulações para todos os *benchmarks* avaliados, exceto G2, onde o SSO-C superou o PSO+ e G11, onde o algoritmo obteve um resultado melhor que o ótimo pelo mesmo motivo do algoritmo FA, onde a tolerância definida para as equações de igualdade foi menor do que a usada neste trabalho.

O PSO+ convergiu, para a maioria dos *benchmarks*, com muito menos avaliações. No caso do G10, PSO_Eb apresentou melhor desempenho em termos de número de avaliações, mas deve-se recordar (**Tabela 1**) que o PSO+ superou com folga o PSO_Eb em termos de resultados. G9 e G11 foram superados pelo algoritmo FA em termos de número de avaliações, sendo que o PSO+ superou FA em termos de resultado para G9; já para G10, o FA apresentou um resultado melhor que o ótimo em função das tolerâncias escolhidas para a solução do problema.

Tabela 4. Comparação de resultados com outros algoritmos evolutivos

	Ótimo		***Melhor***	***Média***	***Pior***	***Desvio Padrão***	***Avaliações***
G01	15.00	PSO+	**-15.000**	**-15.0**	**-11**	**0**	**330**
		RGen.III	-14.890	-14.89	-14.89	-	350000
		Gen.V	-15.000	-9.68	-2.15	-	350000
		RGen.V	-15.000	-12.88	-9.50	-	350000
		eABC	-15.000	-15.000	-15.000	0	240000
		SSO-C	-15.000	-15.000	-15.000	0	25000
G02	0.8036	PSO+	0.7967	0.7967	0.7743	1.59e-6	9510
		eABC	0.8036	0.8021	0.7990	1.26e-3	240000
		SSO-C	**0.8036**	0.8015	0.7925	**3.5e-5**	**25000**
G03	1	PSO+	**1.0000**	**0.6739**	**0.0939**	**1.16e-15**	**3900**
		eABC	**-1.0041**	-1.0031	-1.0010	1.33e-3	240000
G04	-30665.5	PSO+	**-30665.5**	**-30661.09**	**-30665.5**	**5.09e-11**	**2190**
		eABC	-30665.5	-30665.5	-30665.5	0	240000
		FA	-30665.0	-30664.67	-30663.9	-	12000
		SSO-C	-30665.5	-30665.53	-30665.14	1.1e-4	25000
G05	5126.49	PSO+	**5126.49**	**5126.49**	**5905.7**	**1.66e-10**	**2760**
		eABC	**5126.39**	**5299.67**	**5968.58**	**248.42**	**240000**
G06	-6961.8	PSO+	**-6961.8**	**-6961.7**	**-6783.1**	**3.18e-1**	**2490**
		RGen.III	-6961.8	-6961.8	-6961.8	-	350000
		Gen.V	-6873.6	-6787.49	-6614.13	-	350000
		RGen.V	-6961.8	-6961.79	-6961.77	-	350000
		eABC	-6961.8	-6961.81	-6961.81	0	240000
		FA	-6956.6	-6956.63	-6953.47	-	12000
		SSO-C	-6961.8	-6961.01	-6961.91	1.1e-3	25000
G07	24.3062	PSO+	24.310	26.46	42.25	8.52e-14	1860
		RGen.III	25.57	28.23	35.25	-	350000
		Gen.V	24.62	27.14	35.79	-	350000
		RGen.V	24.78	25.31	25.61	-	350000
		eABC	24.55	24.80	25.10	1.2e-1	240000
		FA	24.38	24.47	24.60	-	50000
		SSO-C	**24.306**	**24.306**	**24.306**	**4.9e-5**	**25000**
G08	0.09582	PSO+	**0.09582**	**0.09582**	**0.08544**	**1.89e-4**	**1200**
		eABC	0.09582	0.09582	0.09582	0	240000
		FA	**0.09582**	**0.09582**	**0.09582**	**-**	**12000**
G09	680.6300	PSO+	**680.6300**	**681.6134**	**681.47**	**9.45e-1**	**30000**
		RGen.III	680.6300	680.63	680.63	-	350000
		Gen.V	680.6500	680.67	680.69	-	350000
		RGen.V	680.6300	680.85	680.98	-	350000
		eABC	680.6387	680.6512	680.6776	885.15	240000
		FA	680.8463	681.0415	681.2603	-	12000
G10	7049.25	PSO+	**7050.2600**	**7050.26**	**7222.77**	**8.13e-3**	**28980**
		RGen.III	7129.0000	7508.32	7796.21	-	350000
		Gen.V	7255.54	8559.57	11597.24	-	350000
		RGen.V	7068.44	7182.63	7273.53	-	350000
		eABC	7304.817	7445.860	1647.175	87.75	**240000**

G11	0.75	PSO+	**0.7500**	**0.75**	**1.00**	**1.33e-15**	**20790**
		eABC	0.7490*	0.7490	0.7490	5.0e-7	240000
		FA	**0.7490***	**0.7490**	**0.7490**	**-**	**12000**
		SSO-C	0.7499*	0.7499	0.7499	4.1e-9	25000
G12	1	PSO+	**1.0000**	**1.0000**	**0.52**	**1.27e-10**	**60**
		eABC	1.0000	1.0000	1.0000	0	240000
		FA	0.9999	0.9999	0.9999	-	12000
G13	0.0539	PSO+	**0.0539**	**0.0539**	**0.0622**	**4.13e-17**	**2000**
		eABC	0.4910	0.9439	1.3252	1.71e-1	240000
		SSO-C	0.0539	0.0539	0.0539	6.3e-9	25000

5 Conclusões

Este artigo apresentou um novo algoritmo de otimização de enxame de partículas para otimização com restrições. Demonstrou-se que o PSO é um método eficiente e geral para resolver problemas de otimização com restrições. Ao utilizar o operador de crossover para ser usado com PSO, permite-se que as partículas inválidas facilmente se redirecionem para uma região válida com um baixo custo computacional. Esta abordagem também permite que o algoritmo mantenha uma população inteira de partículas válidas em qualquer momento da otimização. O conceito de pontos de apoio funciona bem quando as restrições de otimização são ativas na solução ótima, permitindo que o enxame, além de manter alguma diversidade nas soluções e melhorar a cobertura do espaço de busca válido, possa alcançar mais facilmente a fronteira do espaço de busca válido. A heurística de iniciação de partículas permite que o estágio inicial da otimização não seja tão custoso do ponto de vista computacional quando o problema apresenta restrições que dificultem a procura por soluções válidas. Os resultados obtidos permitem avaliar positivamente o PSO+. O mesmo apresentou resultados de otimização e velocidade de convergência, em sua maioria, melhores que três modelos de algoritmos genéticos, quatro diferentes implementações, além de três outros algoritmos baseados em inteligência de enxames.

References

1. He Q, Wang L (2007) A hybrid particle swarm optimization with a feasibility-based rule for constrained optimization. Appl Math Comput 186:1407–1422. doi: 10.1016/j.amc.2006.07.134
2. Kennedy J, Eberhart R (1995) Particle swarm optimization. Neural Networks, 1995 Proceedings, IEEE Int Conf 4:1942–1948 vol.4. doi: 10.1109/ICNN.1995.488968
3. Kumar R, Anand S, Sydulu M (2011) A new multi agent based PSO approaches for optimal power flows with security constraints using different cost functions. 2011 2nd Int Conf Intell Agent Multi-Agent Syst 67–72. doi: 10.1109/IAMA.2011.6049008
4. Khamsawang S, Wannakarn P, Jiriwibhakorn S (2010) Hybrid PSO-DE for solving the economic dispatch problem with generator constraints. 2010 2nd Int Conf Comput Autom Eng ICCAE 2010 5:135–139. doi: 10.1109/ICCAE.2010.5451501
5. Fuentes Cabrera J, Coello Coello C (2007) Handling constraints in particle swarm optimization using a small population size. MICAI 2007 Adv Artif Intell 4827:41–51.

6. Robinson J, Rahmat-Samii Y (2004) Particle swarm optimization in electromagnetics. Antennas Propagation, IEEE Trans 52:397–407. doi: 10.1109/TAP.2004.823969
7. Parsopoulos KE, Vrahatis MN (2011) Particle Swarm Optimization Method for Constrained Optimization Problems. Optimization 181:1153–1163. doi: 10.1016/j.ins.2010.11.033
8. Chang W-D, Shih S-P (2010) PID controller design of nonlinear systems using an improved particle swarm optimization approach. Commun Nonlinear Sci Numer Simul 15:3632–3639. doi: 10.1016/j.cnsns.2010.01.005
9. Li X, Tian P, Kong M (2005) A Novel Particle Swarm Optimization for Constrained Optimization Problems. AI 2005 Adv Artif Intell 18th Aust Jt Conf Artif Intell 1305–1310.
10. Xu Y, Lei B, Sun S (2010) Three particle swarm algorithms to improve coverage of camera networks with mobile nodes. Bio-Inspired Comput Theor Appl (BIC-TA), 2010 IEEE Fifth Int Conf 816–820.
11. Hu X, Eberhart R (2002) Solving Constrained Nonlinear Optimization Problems with Particle Swarm Optimization. Optimization 2:1677–1681. doi: 10.1109/CEC.2002.1004494
12. Liu C-A (2008) New Multiobjective PSO Algorithm for Nonlinear Constrained Programming Problems. In: Adv. Cogn. Neurodynamics ICCN 2007. Springer Netherlands, Dordrecht, pp 955–962
13. Lu L, Luo Q, Liu J, Long C (2008) An improved particle swarm optimization algorithm. IEEE Int Conf Granul Comput 486–490. doi: 10.1109/GRC.2008.4664694
14. Michalewicz Z, Schoenauer M (1996) Evolutionary algorithms for constrained parameter optimization problems. Evol Comput vol:4no1pp1-32.
15. Michalewicz Z, Nazhiyath G (1995) Genocop III: a co-evolutionary algorithm for numerical optimization problems with nonlinear constraints. In: Proc. 1995 IEEE Int. Conf. Evol. Comput. IEEE, pp 647–651
16. Liang J, Runarsson TP, Mezura-Montes E, et al (2006) Problem definitions and evaluation criteria for the CEC 2006 special session on constrained real-parameter optimization. 41:
17. Aguirre AH, Zavala AEM, Villa Diharce E, et al (2009) COPSO: CONSTRAINED OPTIMIZATION VIA PSO ALGORITHM. Constraint-Handling Evol. Optim.
18. Kato K, Sakawa M, Katagiri H (2008) Revision of a Floating-Point Genetic Algorithm GENOCOP V for Nonlinear Programming Problems. Open Cybern Syst J 2:24–29. doi: 10.2174/1874110X00802010024
19. Ahmad R, Babaeizadeh S (2014) An Efficient Artificial Bee Colony Algorithm for Constrained Optimization Problems. J Eng Appl Sci 9:405–413.
20. Deshpande AM, Phatnani GM, Kulkarni AJ (2013) Constraint handling in Firefly Algorithm. In: 2013 IEEE Int. Conf. Cybern. IEEE, pp 186–190
21. Cuevas E, Cienfuegos M (2014) A new algorithm inspired in the behavior of the social-spider for constrained optimization. Expert Syst Appl 41:412–425. doi: 10.1016/j.eswa.2013.07.067