OTIMIZAÇÃO DE CONTROLADOR NEBULOSO USANDO UMA ABORDAGEM DE ENXAME DE PARTÍCULAS: FUNDAMENTOS E ESTUDO DE CASO DE UM SISTEMA NÃO-LINEAR DO TIPO HAMMERSTEIN

LEANDRO DOS SANTOS COELHO[1], HELON VICENTE HULTMANN AYALA[2] E ANTONIO AUGUSTO RODRIGUES COELHO[3]

[1] *Programa de Pós-Graduação em Engenharia de Produção e Sistemas, PPGEPS*
[2] *Graduação em Engenharia Mecatrônica (Controle e Automação)*
Pontifícia Universidade Católica do Paraná, Rua Imaculada Conceição 1155, 80215-901 Curitiba, PR, Brasil
E-mails: leandro.coelho@pupr.br, helon.ayala@pucpr.br
[2] *Departamento de Automação e Sistemas, Grupo de Pesquisa em Tecnologias de Controle Aplicado*
Universidade Federal de Santa Catarina, CEP 88040-900, Florianópolis, SC, Brasil
E-mail: aarc@das.ufsc.br

Abstract— This paper presents the optimization of a fuzzy controller based on Particle Swarm Optimization (PSO). In this context, the analysis of performance of a controller using PI (Proportional-Integral) fuzzy part combined with a derivative conventional control action is investigated in this paper. Simulation results indicate the efficiency of fuzzy controller and PSO methodologies for a case study of nonlinear system of Hammerstein type.

Keywords— Fuzzy control, Optimization, Particle swarm optimization.

Resumo— Este artigo apresenta a otimização de um controlador nebuloso baseado em enxame de partículas (PSO, *Particle Swarm Optimization*). Neste contexto, a análise de desempenho do controlador usando parte nebulosa do tipo PI (Proporcional-Integral) combinado com uma ação de controle derivativa convencional é investigada neste artigo. Os resultados de simulação mostraram a eficiência do controlador nebuloso e sua otimização baseada em PSO para um estudo de caso de sistema não-linear do tipo Hammerstein.

Palavras-chave— Controle nebuloso, Otimização, Otimização por enxame de partículas.

1 Introdução

Muitas vezes, as técnicas clássicas de controle clássico, tal como os controladores PI (Proporcional-Integral) e PID (Proporcional-Integral-Derivativo), apresentam dificuldades em lidar com processos não-lineares. Neste contexto, é que as técnicas de controle nebuloso se mostram interessantes, pois geralmente não precisam do modelo matemático do processo para controlá-lo e podem também fazer uso do conhecimento de operador, este implementado em uma base de regras heurísticas.

Deve-se enfatizar que os controladores nebulosos têm chamado atenção pelos resultados bem sucedidos de suas aplicações tanto na academia quanto na indústria em controladores industriais. Para implementar um controlador nebuloso, o projetista necessita não somente de heurística, mas também de alguns procedimentos teóricos na síntese e análise do projeto. Geralmente, a concepção de um controle nebuloso é um projeto de controle não-linear. Assim, é uma tarefa difícil examinar-se a influência de cada parâmetro no desempenho e robustez do seu projeto. O procedimento de projeto adotado para o controlador nebuloso baseado em regras (controlador nebuloso do tipo Mamdani) pode ser resumido pelas seguintes etapas: (i) definir as funções de pertinência das entradas e das saídas; (ii) selecionar a(s) entrada(s) de controle; (iii) especificar as regras associadas às funções de pertinência; (iv) selecionar o método de inferência associado as regras de produção; (v) selecionar o método de nebulização e desnebulização; e (vi) avaliação do controlador.

A contribuição deste trabalho é apresentar o projeto e a otimização de um controlador nebuloso do tipo PI mais parte D (Derivativo) convencional. O controlador nebuloso é validado no controle de um exemplo de modelo não-linear do tipo Hammerstein.

O procedimento de otimização utilizado no controlador é baseado em uma meta-heurística denominada enxame (ou nuvem) de partículas (PSO, *Particle Swarm Optimization*). O algoritmo PSO foi proposto inicialmente por Kennedy e Eberhart (1995) baseada nos estudos do sócio-biologista Edward Osborne Wilson. O algoritmo PSO constitui uma técnica baseada em uma população de soluções e transições aleatórias. O algoritmo PSO apresenta características similares a técnicas da computação evolutiva, que são baseadas em uma população de soluções. Entretanto, no algoritmo PSO é motivada pela simulação de comportamento social e cooperação entre agentes em vez da sobrevivência do indivíduo mais apto. No algoritmo PSO, cada solução candidata (denominada partícula) possui associada uma velocidade. A velocidade é ajustada através de uma equação de atualização que considera a experiência da partícula correspondente e a experiência das outras partículas da população.

O restante do artigo está organizado da seguinte forma. A descrição do projeto de controle nebuloso

com otimização baseado no algoritmo PSO é detalhada na seção 2. Uma breve descrição dos dois estudos de caso e a análise dos resultados obtidos nas simulações são mencionadas, respectivamente, nas seções 3 e 4. Finalizando, a conclusão e comentários sobre futura pesquisa são apresentados na seção 5.

2 Controlador nebuloso

A estrutura básica de um modelo nebuloso pode ser vista com constituindo-se de: (i) *base de regras*, que contém um conjunto de regras nebulosas; (ii) *base de dados*, que define as funções de pertinência das regras nebulosas; e (iii) *mecanismo de inferência*, que executa os procedimentos de inferência sobre as regras e condição para obtenção de uma saída, conforme o fluxo de dados representado na figura 1.

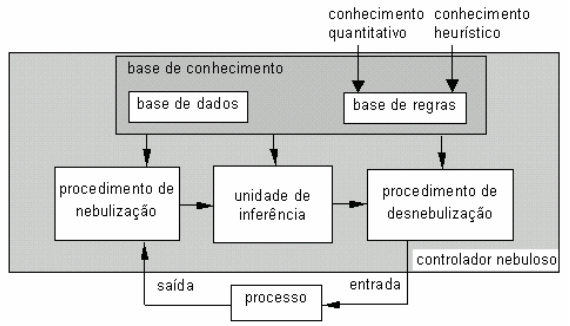

Figura 1. Estrutura de um controlador nebuloso.

A base de conhecimento abrange a base de regras e a base de dados do controlador nebuloso. A base de regras é projetada pela seleção das variáveis de entrada, saída e regras de controle. A base de dados está vinculada à definição do universo de discurso de cada variável de estado, dependendo da escolha das funções de pertinência e dos parâmetros e das funções, que as regem. A fase de nebulização converte os valores numéricos (números reais) em conjuntos nebulosos, ou seja, transforma os dados *crisp* em um conjunto nebuloso correspondente. Existem diversas formas para o projeto de funções de pertinência para as variáveis lingüísticas de erro, variação do erro, controle e variação do controle. Entre as mais utilizadas têm-se as funções trapezoidais, triangulares, Gaussianas ou em forma de sino.

A fase de análise e execução de regras é responsável pela avaliação das regras de produção. Quando uma regra é ativada efetua-se um procedimento de cálculo baseado nos valores dos antecedentes e, então é obtida a saída da regra. A fase de desnebulização objetiva a transformação das variáveis lingüísticas da saída do controlador nebuloso em saídas *crisp*. A desnebulização descreve o mapeamento de um espaço de ações de controle nebuloso em ações de controle não-nebuloso. O procedimento de desnebulização pode ser realizado de várias formas. Entre os métodos de desnebulização utilizados existem o centro de área, a média do máximo, o centro do máximo, o mínimo do máximo e a média ponderada. Entretanto, o centro de área ou centróide é o mais empregado em sistemas de controle nebulosos.

Os modelos lingüísticos, foco deste trabalho, são baseados em regras *se-então*, apresentando predicados vagos e utilizando raciocínio nebuloso. Nestes modelos, as quantidades nebulosas são associadas aos termos lingüísticos, e o modelo nebuloso é essencialmente uma expressão qualitativa do sistema. Os modelos, deste tipo, formam a base de modelagem qualitativa, que descreve o comportamento do sistema através da utilização da linguagem natural. O representante mais significativo desta classe é o modelo do tipo Mamdani.

O modelo de Mamdani (Mamdani e Assilian, 1975) caracteriza-se por utilizar os conjuntos nebulosos como conseqüentes das regras de produção, ou seja,

$$R_i: \quad \text{SE } x_1 \text{ é } A_{i,1} \text{ e } \ldots \text{ e } x_m \text{ é } A_{i,m}$$
$$\text{ENTÃO } y \text{ é } C_i \qquad (1)$$

onde x_i são as variáveis de entrada, y é a variável de saída, A_i e C_i são conjuntos nebulosos.

O projeto de controlador PI nebuloso adicionado a uma ação D convencional, proposto por Qin (1994), apresentado na figura 2 foi adotado neste trabalho.

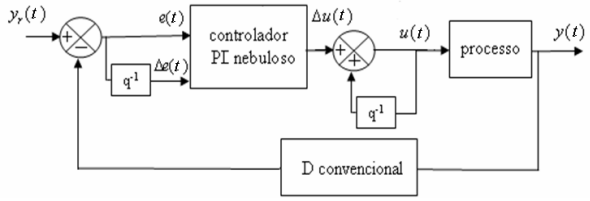

Figura 2. Controle PI nebuloso mais ação D convencional.

As equações que regem este controlador são:

$$u(t) = K_c\,[u_{PI}(t) + u_D(t)] \qquad (2)$$

$$u_{PI}(t) = K_e\,[e(t)] + K_{\Delta e}\,[\Delta e(t)] \qquad (3)$$

onde $u(t)$ é a ação de controle incremental, $u_{PI}(t)$ e $u_D(t)$ são as ações de controle PI e D; K_e , $K_{\Delta e}$ e K_c são os fatores de escala do erro $e(t)$, da variação do erro $\Delta e(t)$ e do sinal de controle, respectivamente.

A ação de controle derivativa leva em consideração um ganho em relação à derivada da saída do processo em relação a referência, ou seja,

$$u_D(t) = K_D\,y(t) - y_r(t)\,, \qquad (4)$$

onde K_D é ganho derivativo da saída do processo e $y_r(t)$ é a saída de referência (desejada). O princípio deste projeto é o algoritmo do tipo PI nebuloso, que tem a vantagem da implementação de controle derivativo na saída, evitando picos derivativos frente a mudanças de referência.

O controlador PI nebuloso mais D convencional apresenta uma vantagem em relação ao projeto do PID convencional, pois um controlador PID convencional tem um comportamento aceitável quando atua em torno de um determinado ponto de operação, mas, freqüentemente, apresenta um desempenho pobre em regime transiente quando mudanças entre pontos diferentes de operação ocorrem devido a mudanças na dinâmica do processo (comportamento regulatório deficiente) e o controlador não é sintonizado adequadamente.

2.1 Otimização do controlador usando PSO

O algoritmo PSO é uma abordagem da inteligência coletiva baseada em população de soluções. De forma similar aos algoritmos evolutivos, o PSO inicia com uma população de soluções gerada aleatoriamente com distribuição uniforme.

De forma diferente dos algoritmos evolutivos clássicos, cada solução potencial (indivíduo) no PSO é também atribuída uma velocidade aleatória. As soluções potenciais denominadas *partículas* são então "movimentadas" pelo espaço de busca do problema. Cada partícula conserva o conhecimento do seu melhor valor da função de aptidão (*fitness*) denotada por *pbest* (*p*ersonal *best*). Um outro melhor valor é "seguido" pela versão *global*, *gbest* (*g*lobal *best*), da PSO e sua localização obtida de alguma partícula que compõe a população.

O conceito do PSO consiste de, a cada passo iterativo, mudar a velocidade de cada partícula em direção as localizações do *pbest* e do *gbest*. A rapidez do procedimento de busca é ponderada através de um termo gerado de forma aleatória, sendo vinculado este, de forma separada, as localizações do *pbest* e do *gbest.* O procedimento para implementação da PSO é regido pelas seguintes etapas:

(i) iniciar uma população (matriz) de *TP* partículas, com posições e velocidades em um espaço de problema *n* dimensional, aleatoriamente com distribuição uniforme.

(ii) para cada partícula, avaliar a função de aptidão (função objetivo);

(iii) comparar a avaliação da função de aptidão da partícula com o *pbest* da partícula. Se o valor corrente é melhor que *pbest*, então o valor de *pbest* passa a ser igual ao valor da função de aptidão da partícula, e a localização do *pbest* passa a ser igual a localização atual no espaço *n* dimensional;

(iv) comparar a avaliação da função de aptidão com o prévio melhor valor de aptidão da população. Se o valor atual é melhor que o *gbest*, atualizar o valor de *gbest* para o índice e valor da partícula atual;

(v) modificar a velocidade e a posição da partícula de acordo com as equações:

$$v_i(k+1) = w \cdot v_i(k) + c_1 \cdot ud \cdot [p_i(k) - x_i(k)] + c_2 \cdot Ud \cdot [p_g(k) - x_i(k)] \quad (5)$$

$$x_i(k+1) = x_i(k) + \Delta k \cdot v_i(k+1) \quad (6)$$

onde Δk é igual a 1, k é a geração atual e i=1,...,*TP*, tal que o tamanho de população é *TP*.

(vi) ir para a etapa (ii) até que um critério de parada seja encontrado, usualmente uma função de aptidão suficientemente boa ou um número máximo de iterações (gerações), k_{max}.

As notações usadas são: $x_i = [x_{i1}, x_{i2}, ..., x_{in}]^T$ armazena a posição da i-ésima partícula, $v_i = [v_{i1}, v_{i2}, ..., v_{in}]^T$ armazena a velocidade da i-ésima partícula e $p_i = [p_{i1}, p_{i2}, ..., p_{in}]^T$ representa a posição do melhor valor de aptidão da i-ésima partícula. O índice g representa o índice da melhor partícula entre todas as partículas do grupo. A variável w é a ponderação de inércia, c_1 e c_2 são constantes positivas; *ud* e *Ud* são duas funções para geração de números aleatórios com distribuição uniforme no intervalo [0,1], respectivamente.

As velocidades das partículas em cada dimensão são limitadas a um valor máximo de velocidade, V_{max}. O V_{max} é importante, pois determina à resolução na qual a região próxima a solução atual é procurada. Se V_{max} é alto, o algoritmo PSO facilita a busca global, enquanto um valor V_{max} pequeno enfatiza as buscas locais.

A primeira parte na equação (5) é um termo de momento da partícula. A ponderação de inércia w representa o grau de momento da partícula. A segunda parte consiste da parte "cognitiva", que representa o "conhecimento" independente da partícula. A terceira parte é a "social", que representa a colaboração entre as partículas.

As constantes c_1 e c_2 representam ponderações às partes cognitivas e social que influenciam cada partícula em direção ao *pbest* e *gbest*, respectivamente. Estes parâmetros são usualmente ajustados por heurísticas de tentativa e erro. O tamanho da população também é selecionado dependendo do problema. Neste trabalho, uma modificação que varia durante as gerações t de c_1 e c_2 foi utilizada. Esta modificação pode ser representada por (Ratnaweera *et al.*, 2004):

$$c_1(k) = (c_{1f} - c_{1i}) \cdot \frac{k}{k_{\max}} + c_{1i} \quad (7)$$

$$c_2(k) = (c_{2f} - c_{2i}) \cdot \frac{k}{k_{\max}} + c_{2i} \quad (8)$$

onde c_{1i}, c_{1f}, c_{2i} e c_{2f} são constantes.

A otimização dos ganhos do controlador nebuloso pelo algoritmo PSO visa a minimização da função objetivo, *f*, dada por:

$$f = 0{,}25 \cdot \sum_{t,k=1}^{N} k \cdot |e(t)| \quad (9)$$

onde N é o número total de amostras avaliado, $e(t)$ é o erro dado pela diferença entre a saída, $y(t)$, do sistema e a referência desejada (*setpoint*), $y_r(t)$. Um detalhe importante, é que a variável k é igual a variável t, mas é reiniciada a cada mudança de referência (neste trabalho foram 3 mudanças de referência para cada um dos exemplos testados).

O algoritmo PSO foi utilizado para otimização dos fatores de escala do erro (K_e), da variação do erro ($K_{\Delta e}$), do sinal de controle (K_c) e do ganho derivativo (K_D) da parte convencional do controlador. A otimização do controlador nebuloso pelo algoritmo PSO foi realizada para melhorar o comportamento servo do sistema de controle em malha fechada de um sistema do tipo Hammerstein.

3 Descrição do sistema do tipo Hammerstein

O sistema não-linear avaliado neste trabalho é do tipo Hammerstein e representa uma coluna de destilação, tal que:

$$y(t) = 0{,}796\,y(t-1) + 0{,}204\,x(t-1) \quad (10)$$

$$x(t) = 1{,}04u(t-1) - 14{,}11u^2(t-1) - 16{,}72u^3(t-1) + 562{,}75u^4(t-1) \quad (11)$$

Este modelo relaciona a composição do topo de uma coluna de destilação y (%) com a taxa de fluxo de refluxo u (mol/min). Ambas variáveis de entrada e saída no modelo são definidas como desvios dos valores nominais. Uma descrição detalhada do modelo pode ser encontrada em Eskinat *et al*. (1991).

4 Resultados de controle e otimização

A implementação do controlador nebuloso foi realizada em ambiente computacional Matlab, da MathWorks. Para calcular os ganhos e os fatores de escala das funções de pertinências dos controladores nebulosos, se utilizou o algoritmo PSO.

As funções de pertinência de erro, variação do erro e sinal de controle dos dois controladores nebulosos são do tipo triangular são projetadas neste trabalho conforme apresentado na figura 3. Na tabela 1 apresenta a base de regras do controlador nebuloso projetada para o controlador nebuloso para o sistema não-linear do tipo Hammerstein.

As convenções adotadas na figura 3 e tabela 1 são as seguintes: PP é o Positivo Pequeno, PG é o Positivo Grande, Z é o Zero, NP é o Negativo Pequeno, NG é o Negativo Grande, NS é o Negativo do Sinal de erro, Nd é o Negativo da Derivada do erro, PS consiste do Positivo do Sinal de erro, Pd são o Positivo da Derivada do erro, ZS o Zero do Sinal de erro e Zd é o Zero da Derivada do erro.

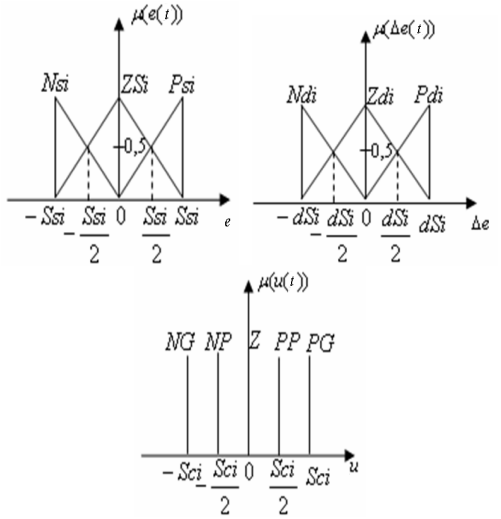

Figura 3. Funções de pertinência do erro, da variação do erro e do sinal de controle.

Tabela 1. Base de regras do controle PI nebuloso.

e \ Δe	Pd	Zd	Nd
PS	PP	PP	PG
ZS	Z	Z	Z
NS	NG	NP	NP

Nas simulações computacionais de otimização do controlador nebuloso pelo PSO foram utilizados os seguintes parâmetros de projeto:

- tamanho da população do algoritmo PSO: 20 partículas;
- fator de inércia com decaimento linear, iniciando com valor 0,9 e terminado com 0,4;
- velocidade máxima, V_{max}, é configurada para 20% do intervalo da variável de cada dimensão;
- PSO clássico (PSOC): adota os coeficientes $c_1 = c_2$=2,05;
- PSO modificado (PSOM): adota os coeficientes $[c_{1i}; c_{1f}]$=[2,05; 0,40] e $[c_{2i}; c_{2f}]$=[0,40; 2,05].
- número máximo de gerações (critério de parada) do algoritmo PSO: 20 gerações;
- espaço de busca dos quatro parâmetros de projeto dos dois exemplos: $K_e \in [0; 6]$; $K_{\Delta e} \in [-6;1]$; $K_c \in [0; 10]$ e $K_D \in [-1 ; 0{,}001]$.
- restrições impostas ao sinal de controle: $u \in [-0{,}2; 0{,}2]$.

Os melhores ganhos obtidos para o controlador nebuloso para 30 simulações com a otimização via PSOC e PSOM são apresentados na tabela 1. O resultado usando o PSOM foi superior ao obtido com o PSOC em termos do valor da função objetivo (problema de minimização).

A figura 4 mostra o resultado de controle nebuloso em malha fechada quando usados os parâmetros de projeto apresentados na tabela 1 para o

PSOM. Nota-se pelo resultado da figura 4 que o controlador foi bem sucedido no controle do sistema não-linear. Entretanto, nota-se uma maior dificuldade do controlador em lidar com a dinâmica não-linear do processo, pois o sistema de controle apresentou dinâmica oscilatória antes de estabilizar na referência estipulada pelo projetista.

Tabela 1. Resultado usando PSOC e PSOM (30 simulações) para otimização do controlador nebuloso PI + D convencional.

PSOC				
f	K_e	$K_{\Delta e}$	K_c	K_D
0,8842	0,2674	-4,9998	0,9992	-0,3280
PSOM				
f	K_e	$K_{\Delta e}$	K_c	K_D
0,8033	1,4736	-4,9569	4,9398	0,0004

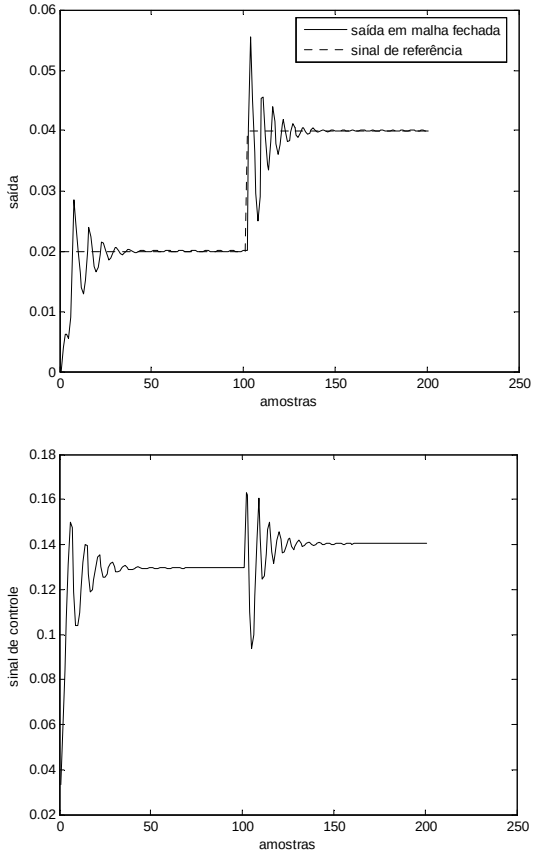

Figura 4. Resultado de controle nebuloso otimizado por PSOM para um sistema não-linear.

5 Conclusão

A análise e o projeto de um controlador nebuloso do tipo Mamdani requer a sintonia de parâmetros muitas vezes de forma heurística. Neste artigo o projeto de base de regras do controlador nebuloso foi realizado *a priori* e a otimização dos fatores de escala e ganho derivativo de uma concepção de controlador PI nebuloso combinado à ação D convencional foi realizada usando o algoritmo PSOM.

Neste trabalho, a proposta de avaliar um sistema não-linear do tipo Hammerstein foi apresentada e validada com sucesso pelo controlador nebuloso otimizado pelo PSOM.

Em futura pesquisa vislumbra-se à concepção de técnicas de controle nebuloso em sistemas não-lineares multivariáveis que apresentem forte acoplamento entre as variáveis de entrada e saída.

Agradecimentos

Os autores agradecem o apoio financeiro do Conselho Nacional de Desenvolvimento Científico e Tecnológico – CNPq (processo: 309646/2006-5/PQ).

Referências Bibliográficas

Eskinat, E., Johnson, S. H. e Luyben, W. L. (1991). Use of Hammerstein models in identification of nonlinear systems, AIChE Journal, **37**(1): 255-268.

Kennedy, J. e Eberhart, R. C. (1995). Particle swarm optimization, *Proceedings of IEEE International Conference on Neural Networks*, Perth, Australia, pp. 1942-1948.

Mamdani, E. H. e Assilian, S. (1975). An experiment in linguistic synthesis with a fuzzy logic controller, *International Journal of Man-Machine Studies*, **7**(1): 1-13.

Qin, S. J. (1994). Auto-tuned fuzzy logic control, *Proceedings of American Control Conference*, Baltimore, Maryland, USA, pp. 2465-2469.

Ratnaweera, A., Halgamuge, S. K. e Watson, H. C., 2004, Self-Organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients, *IEEE Transactions on Evolutionary Computation*, **8**(3): 240-255.