The Particle Swarm Optimization Algorithm

Jaco F. Schutte

EGM 6365 - Structural Optimization
Fall 2005

Overview

- Introduction and background
- Applications
- Particle swarm optimization algorithm
- Algorithm variants
- Synchronous and asynchronous PSO
- Parallel PSO
- Structural optimization test set
- Concluding remarks
- References

Particle Swarm Optimizer

- Introduced by Kennedy & Eberhart 1995
- Inspired by social behavior and movement dynamics of insects, birds and fish
- Global gradient-less stochastic search method
- Suited to continuous variable problems
- Performance comparable to Genetic algorithms
- Has successfully been applied to a wide variety of problems (Neural Networks, Structural opt., Shape topology opt.)

Particle Swarm Optimizer

- Advantages
  - Insensitive to scaling of design variables
  - Simple implementation
  - Easily parallelized for concurrent processing
  - Derivative free
  - Very few algorithm parameters
  - Very efficient global search algorithm
- Disadvantages
  - Slow convergence in refined search stage (weak local search ability)

PSO applications

- Training of neural networks
  - Identification of Parkinson’s disease
  - Extraction of rules from fuzzy networks
  - Image recognition
- Optimization of electric power distribution networks
- Structural optimization
  - Optimal shape and sizing design
  - Topology optimization
- Process biochemistry
- System identification in biomechanics

Particle swarm optimization algorithm

Basic algorithm as proposed by Kennedy and Eberhart (1995)

$\boldsymbol{x}_k^i$ - Particle position

$\boldsymbol{v}_k^i$ - Particle velocity

$\boldsymbol{p}_k^i$ - Best "remembered" individual particle position

$\boldsymbol{p}_k^g$ - Best "remembered" swarm position

c_1, c_2 - Cognitive and social parameters

r_1, r_2 - Random numbers between 0 and 1

Position of individual particles updated as follows:

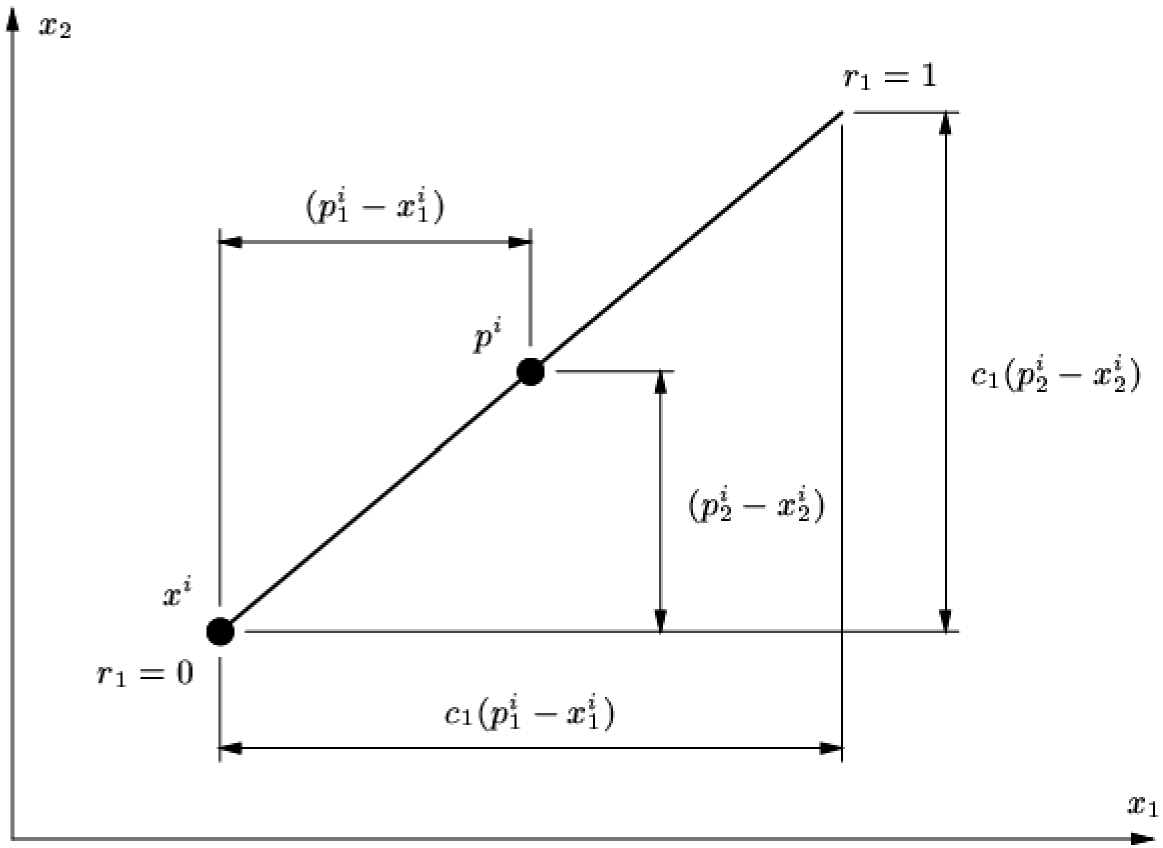
$$\boldsymbol{x}_{k+1}^i = \boldsymbol{x}_k^i + \boldsymbol{v}_{k+1}^i,$$

with the velocity calculated as follows:

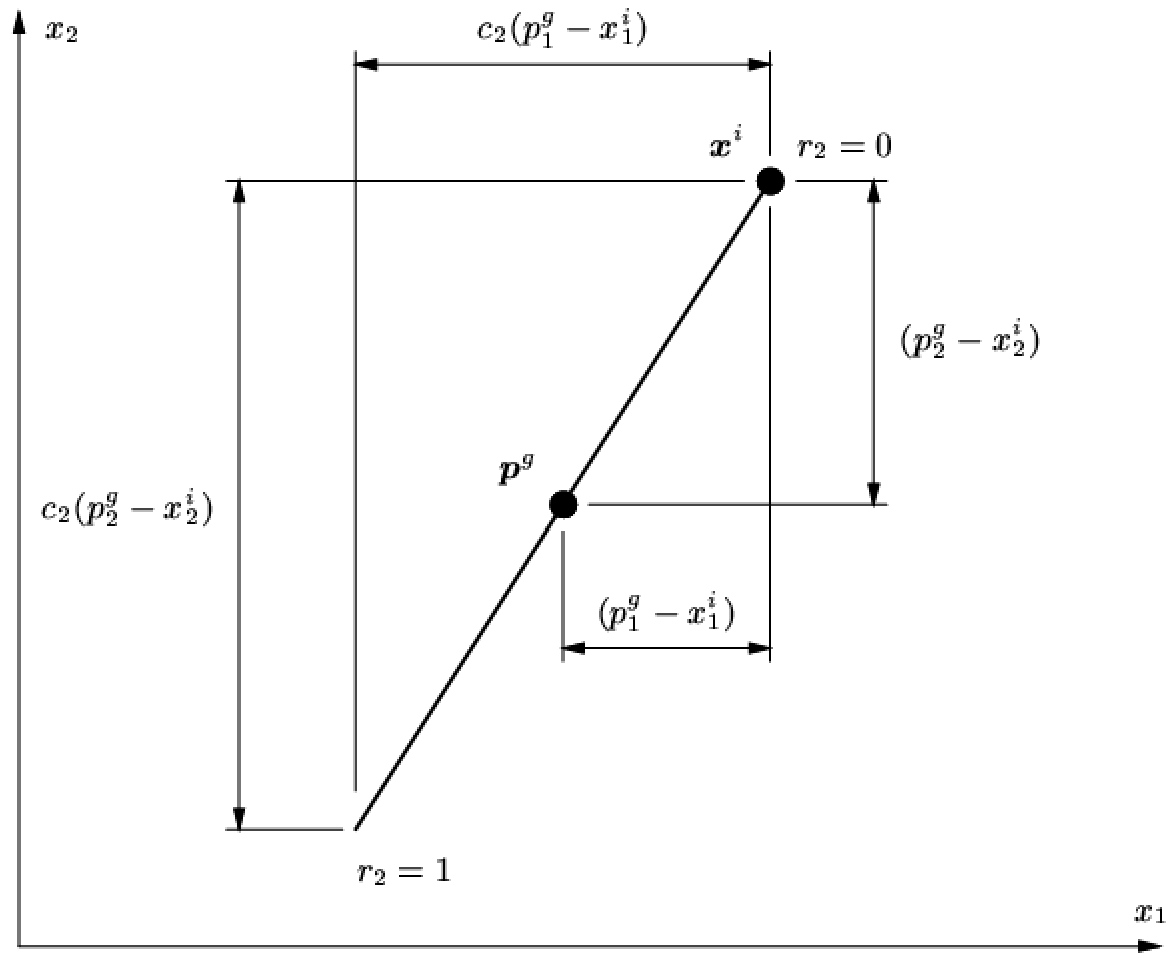
$$\boldsymbol{v}_{k+1}^i = \boldsymbol{v}_k^i + c_1 r_1 (\boldsymbol{p}_k^i - \boldsymbol{x}_k^i) + c_2 r_2 (\boldsymbol{p}_k^g - \boldsymbol{x}_k^i).$$

PSO algorithm flow diagram

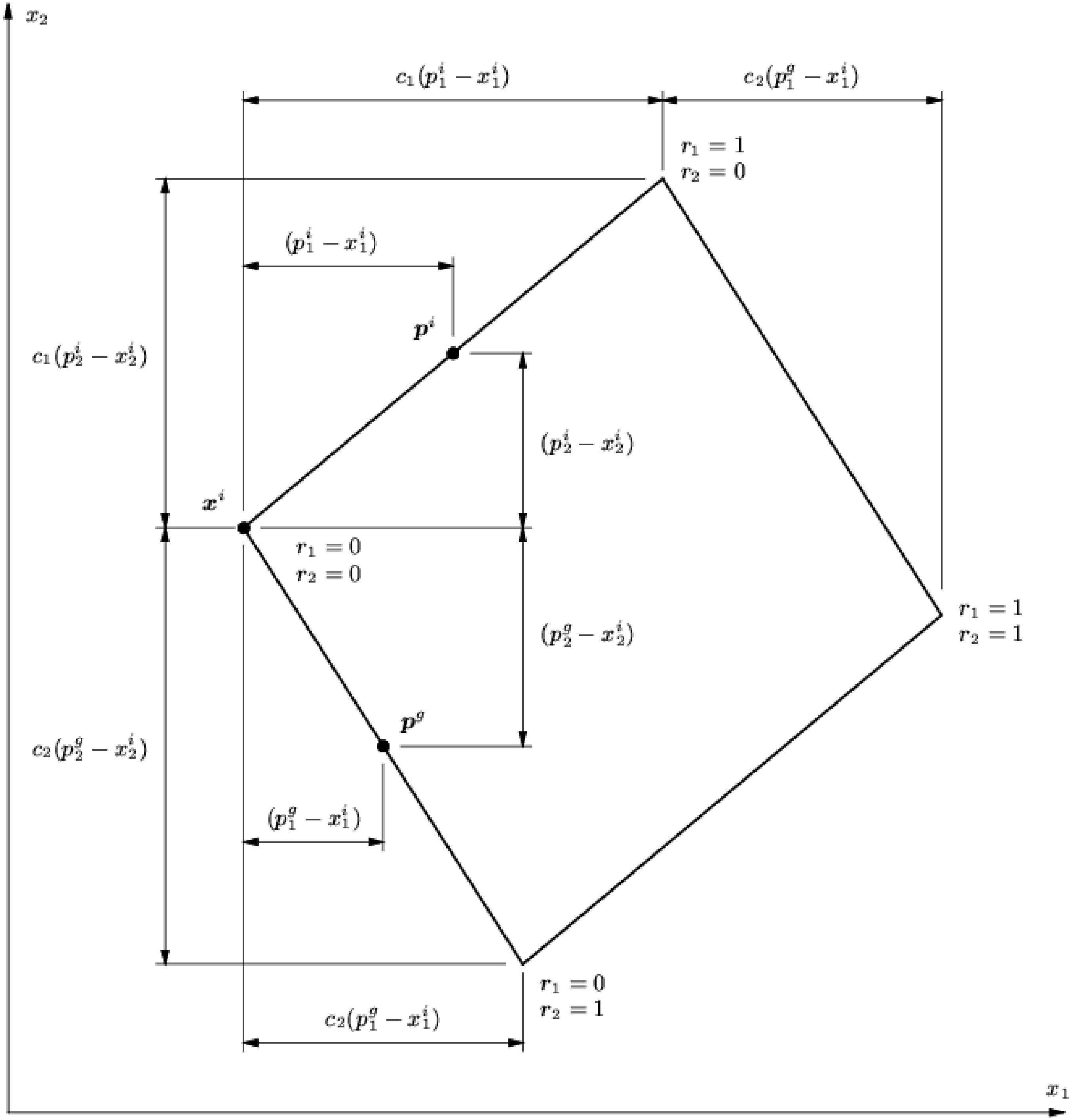
1. Initialize
   (a) Set constants k_{max}, c_1, c_2.
   (b) Randomly initialize particle positions $\boldsymbol{x}_0^i \in \boldsymbol{D}$ in $\mathbb{R}^n$ for $i = 1, ..., p$.
   (c) Randomly initialize particle velocities $0 \leq \boldsymbol{v}_0^i \leq \boldsymbol{v}_0^{max}$ for $i = 1, ..., p$.
   (d) Set $k = 1$
2. Optimize
   (a) Evaluate function value f_k^i using design space coordinates $\boldsymbol{x}_k^i$.
   (b) If $f_k^i \leq f_{best}^i$ then $f_{best}^i = f_k^i$, $\boldsymbol{p}_k^i = \boldsymbol{x}_k^i$.
   (c) If $f_k^i \leq f_{best}^g$ then $f_{best}^g = f_k^i$, $\boldsymbol{p}_k^g = \boldsymbol{x}_k^i$.
   (d) If stopping condition is satisfied then goto 3.
   (e) Update all particle velocities $\boldsymbol{v}_k^i$ for $i = 1, ..., p$
   (f) Update all particle positions $\boldsymbol{x}_k^i$ for $i = 1, ..., p$
   (g) Increment k.
   (h) Goto 2(a).
3. Terminate



Cognitive component search space contribution for 2-D problem



Social component search space contribution for 2-D problem



x_2
$c_1(p_1^i - x_1^i)$
$c_2(p_1^g - x_1^i)$
$r_1 = 1$
$r_2 = 0$
$(p_1^i - x_1^i)$
$\boldsymbol{p}^i$
$c_1(p_2^i - x_2^i)$
$(p_2^i - x_2^i)$
$\boldsymbol{x}^i$
$r_1 = 0$
$r_2 = 0$
$r_1 = 1$
$r_2 = 1$
$(p_2^g - x_2^i)$
$\boldsymbol{p}^g$
$c_2(p_2^g - x_2^i)$
$(p_1^g - x_1^i)$
$r_1 = 0$
$r_2 = 1$
$c_2(p_1^g - x_1^i)$
x_1

Particle Swarm Algorithm variants

- Good exploration abilities, but weak exploitation of local optima
- Can accelerate “collapse” of swarm for better local search – at the cost of higher possibility of premature convergence
- Accelerated localized search achieved by algorithm modifications

Particle Swarm Algorithm variants

- Constant inertia weight
- Linear reduction of inertia weight
- Constriction factor
- Dynamic inertia and maximum velocity reduction
- Tracking of time dependent minima
- Discrete optimization

Constant inertia weight

Inertia term introduced in velocity rule:

$$\boldsymbol{v}_{k+1}^{i} = w_k \boldsymbol{v}_k^{i} + c_1 r_1 \left(\boldsymbol{p}_k^{i} - \boldsymbol{x}_k^{i} \right) + c_2 r_2 \left(\boldsymbol{p}_k^{g} - \boldsymbol{x}_k^{i} \right)$$

Position update rule remains unchanged:

$$\boldsymbol{x}_{k+1}^{i} = \boldsymbol{x}_k^{i} + \boldsymbol{v}_{k+1}^{i}$$

Linear inertia reduction

Addition of inertia term to velocity rule:

$$\boldsymbol{v}_{k+1}^{i} = w_k \boldsymbol{v}_k^i + c_1 r_1 \left(\boldsymbol{p}_k^i - \boldsymbol{x}_k^i\right) + c_2 r_2 \left(\boldsymbol{p}_k^g - \boldsymbol{x}_k^i\right)$$

$$w_{k+1} = \alpha w_k \,, \quad 0 < \alpha < 1$$

Position rule unchanged:

$$\boldsymbol{x}_{k+1}^{i} = \boldsymbol{x}_k^i + \boldsymbol{v}_{k+1}^i$$

Constriction factor

Velocity rule modified to:

$$\boldsymbol{v}_{k+1}^{i} = K * \left[\boldsymbol{v}_{k}^{i} + c_1 r_1 \left(\boldsymbol{p}_{k}^{i} - \boldsymbol{x}_{k}^{i}\right) + c_2 r_2 \left(\boldsymbol{p}_{k}^{g} - \boldsymbol{x}_{k}^{i}\right)\right],$$

$$K = \frac{2}{\left|2 - \varphi - \sqrt{\varphi^2 - 4\varphi}\right|} \qquad \text{where} \qquad \varphi = c_1 + c_2, \qquad \varphi > 4.$$

Position rule unchanged:

$$\boldsymbol{x}_{k+1}^{i} = \boldsymbol{x}_{k}^{i} + \boldsymbol{v}_{k+1}^{i}$$

Dynamic inertia and maximum velocity reduction

Inertia weight velocity rule used:

$$\boldsymbol{v}_{k+1}^{i} = w_k \boldsymbol{v}_k^{i} + c_1 r_1 \left(\boldsymbol{p}_k^{i} - \boldsymbol{x}_k^{i}\right) + c_2 r_2 \left(\boldsymbol{p}_k^{g} - \boldsymbol{x}_k^{i}\right)$$

Maximum velocity limited:

$$\boldsymbol{v}^{max} = \gamma(\boldsymbol{x}_{UB} - \boldsymbol{x}_{LB})$$

$$\text{If} \quad f(\boldsymbol{p}_k^{g}) \geq f(\boldsymbol{p}_{k-h}^{g}), \quad \text{then} \quad w_{k+1} = \alpha w_k, \; \boldsymbol{v}_k^{max} = \beta \boldsymbol{v}_k^{max},$$

$$\text{with} \;\; 0 < \alpha, \beta < 1$$

Social pressure operator

1. Initialize

   (a) Set constants k_{max}, c_1, c_2.

   (b) Randomly initialize particle positions $\boldsymbol{x}_0^i \in \boldsymbol{D}$ in $\mathbb{R}^n$ for $i = 1, ..., p$.

   (c) Randomly initialize particle velocities $0 \leq \boldsymbol{v}_0^i \leq \boldsymbol{v}_0^{max}$ for $i = 1, ..., p$.

   (d) Set $k = 1$

2. Optimize

   (a) Evaluate function value f_k^i using design space coordinates $\boldsymbol{x}_k^i$.

   (b) If $f_k^i < p_{best}^i$ and $NI^i < NI_{allow}$ then $p_{best}^i = f_k^i$, $\boldsymbol{p}_k^i = \boldsymbol{x}_k^i$, else $c_1 = 0$

   (c) If $f_k^i < g_{best}$ and $NI^i < NI_{allow}$ then $g_{best} = f_k^i$, $\boldsymbol{p}_k^g = \boldsymbol{x}_k^i$

   (d) If stopping condition is satisfied then goto 3.

   (e) Update all particle velocities $\boldsymbol{v}_k^i$ for $i = 1, ..., p$ with rule (2.1).

   (f) Update all particle positions $\boldsymbol{x}_k^i$ for $i = 1, ..., p$ with rule (2.2).

   (g) Increment k.

   (h) Goto 2(a).

3. Terminate

Synchronous vs. Asynchronous PSO

- Original PSO implemented in a synchronous manner
- Improved convergence rate is achieved when p^i and p^g are updated after each fitness evaluation (asynchronous)

Synchronous Particle Swarm Algorithm (parallel processing)

1. Initialize population

2. Optimize

   (a) Evaluate **all** fitness values f_k^i (possibly using **parallel processes)**, at $\boldsymbol{x^i}$

   (b) Barrier synchronization of all processes

   (c) If $f_k^i < f_{best}^i$ then $f_{best}^i = f_k^i$, $\boldsymbol{p_k^i} = \boldsymbol{x_k^i}$

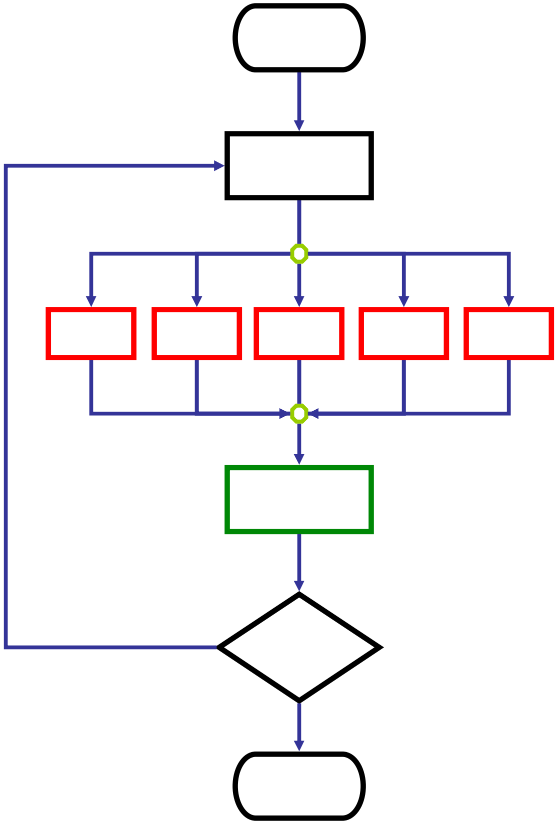
   (d) If $f_k^g < f_{best}^g$ then $f_{best}^g = f_k^g$, $\boldsymbol{p_k^g} = \boldsymbol{x_k^i}$

   (e) If stopping condition is satisfied go to 3

   (f) Update particle velocity $\boldsymbol{v_{k+1}^i}$ and position $\boldsymbol{x_{k+1}^i}$

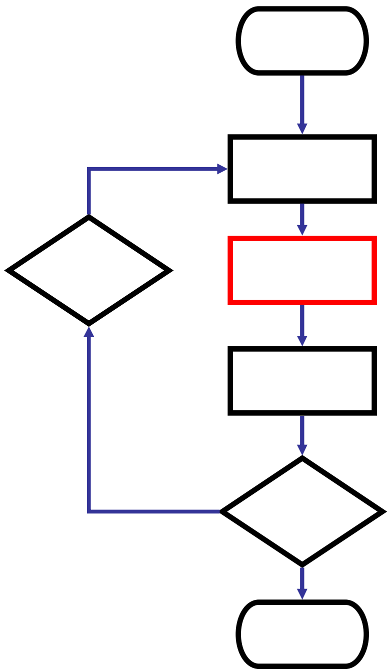
   (g) Increment k

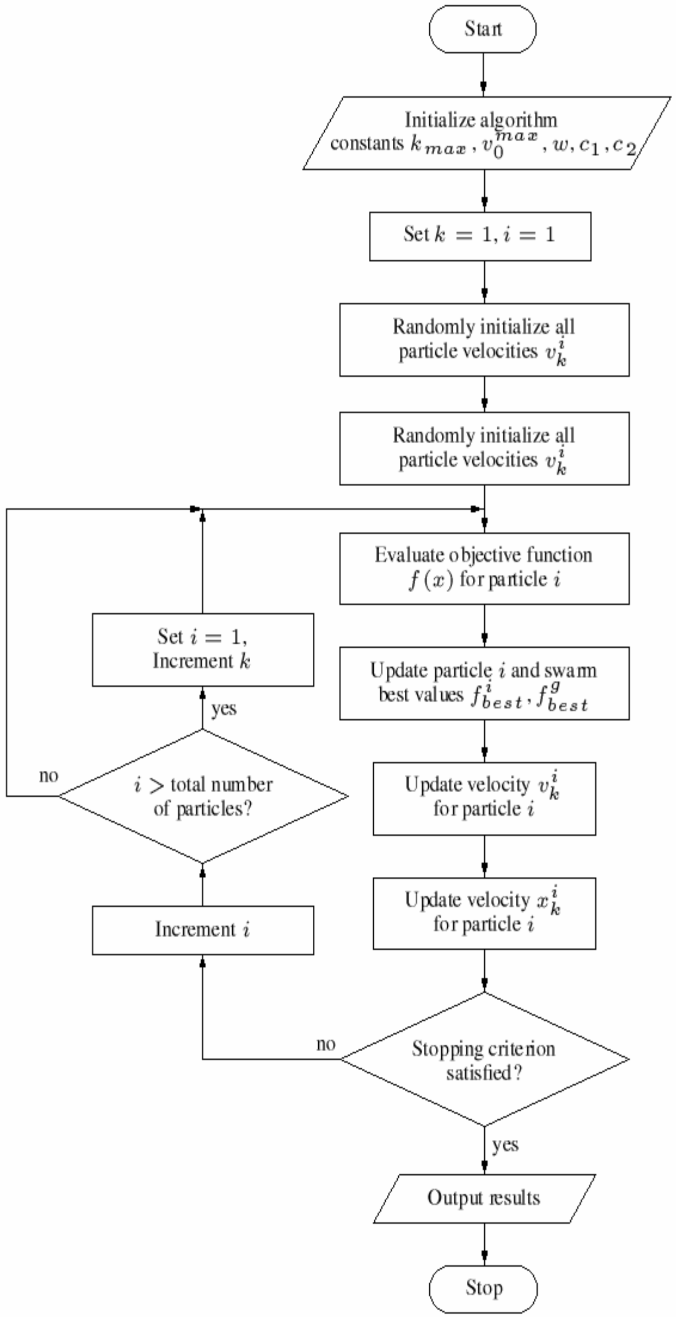
   (h) Go to 2(a)



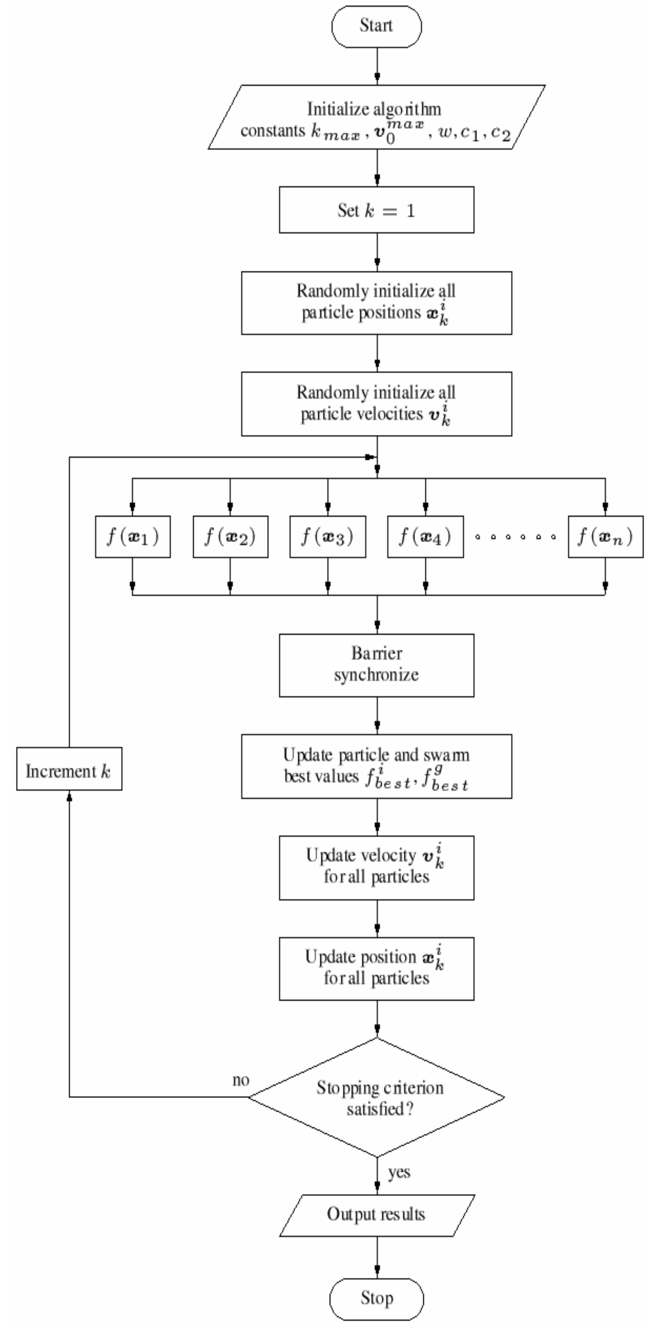
Asynchronous Particle Swarm Algorithm

1. Initialize population
2. Optimize
   (a) Evaluate fitness value f_k^i at $\boldsymbol{x}^i$
   (b) If $f_k^i < f_{best}^i$ then $f_{best}^i = f_k^i$, $\boldsymbol{p}_k^i = \boldsymbol{x}_k^i$
   (c) If $f_k^g < f_{best}^g$ then $f_{best}^g = f_k^g$, $\boldsymbol{p}_k^g = \boldsymbol{x}_k^i$
   (d) If stopping condition is satisfied go to 3
   (e) Update particle velocity $\boldsymbol{v}_{k+1}^i$ and position vector $\boldsymbol{x}_{k+1}^i$
   (f) Increment i. If i > p then increment k, i = 1
   (g) Go to 2(a)
3. Report results and terminate





Start
Initialize algorithm constants $k_{max}, v_0^{max}, w, c_1, c_2$
Set $k = 1, i = 1$
Randomly initialize all particle velocities v_k^i
Randomly initialize all particle velocities v_k^i
Evaluate objective function $f(x)$ for particle i
Update particle i and swarm best values f_{best}^i, f_{best}^g
Update velocity v_k^i for particle i
Update velocity x_k^i for particle i
Stopping criterion satisfied?
no
yes
Output results
Stop
Increment i
$i >$ total number of particles?
no
yes
Set $i = 1$, Increment k



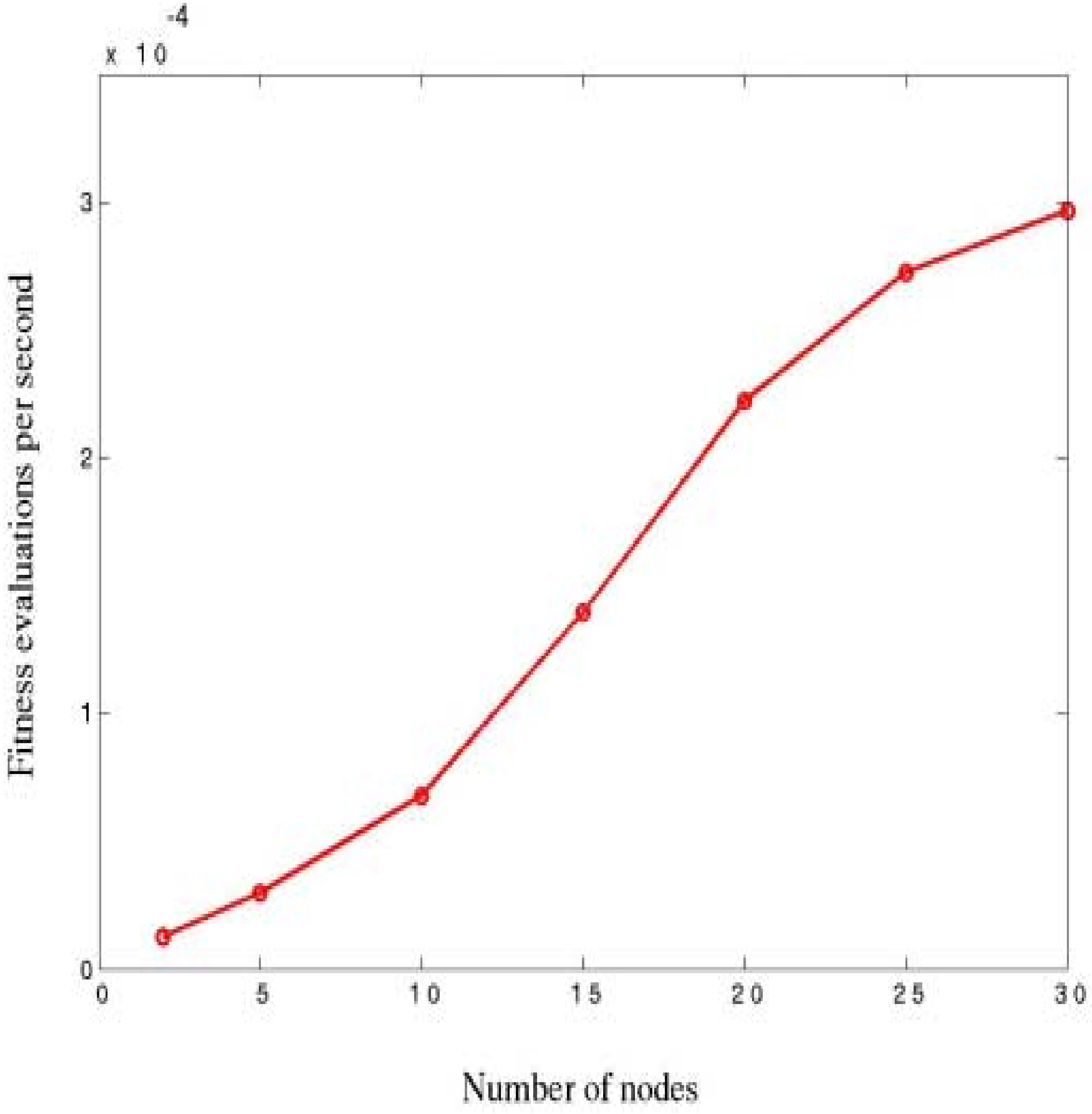
Start
Initialize algorithm constants $k_{max}, \boldsymbol{v}_0^{max}, w, c_1, c_2$
Set $k = 1$
Randomly initialize all particle positions $\boldsymbol{x}_k^i$
Randomly initialize all particle velocities $\boldsymbol{v}_k^i$
$f(\boldsymbol{x}_1)$
$f(\boldsymbol{x}_2)$
$f(\boldsymbol{x}_3)$
$f(\boldsymbol{x}_4)$
$f(\boldsymbol{x}_n)$
Barrier synchronize
Update particle and swarm best values f_{best}^i, f_{best}^g
Update velocity $\boldsymbol{v}_k^i$ for all particles
Update position $\boldsymbol{x}_k^i$ for all particles
Stopping criterion satisfied?
no
yes
Increment k
Output results
Stop

Parallel PSO

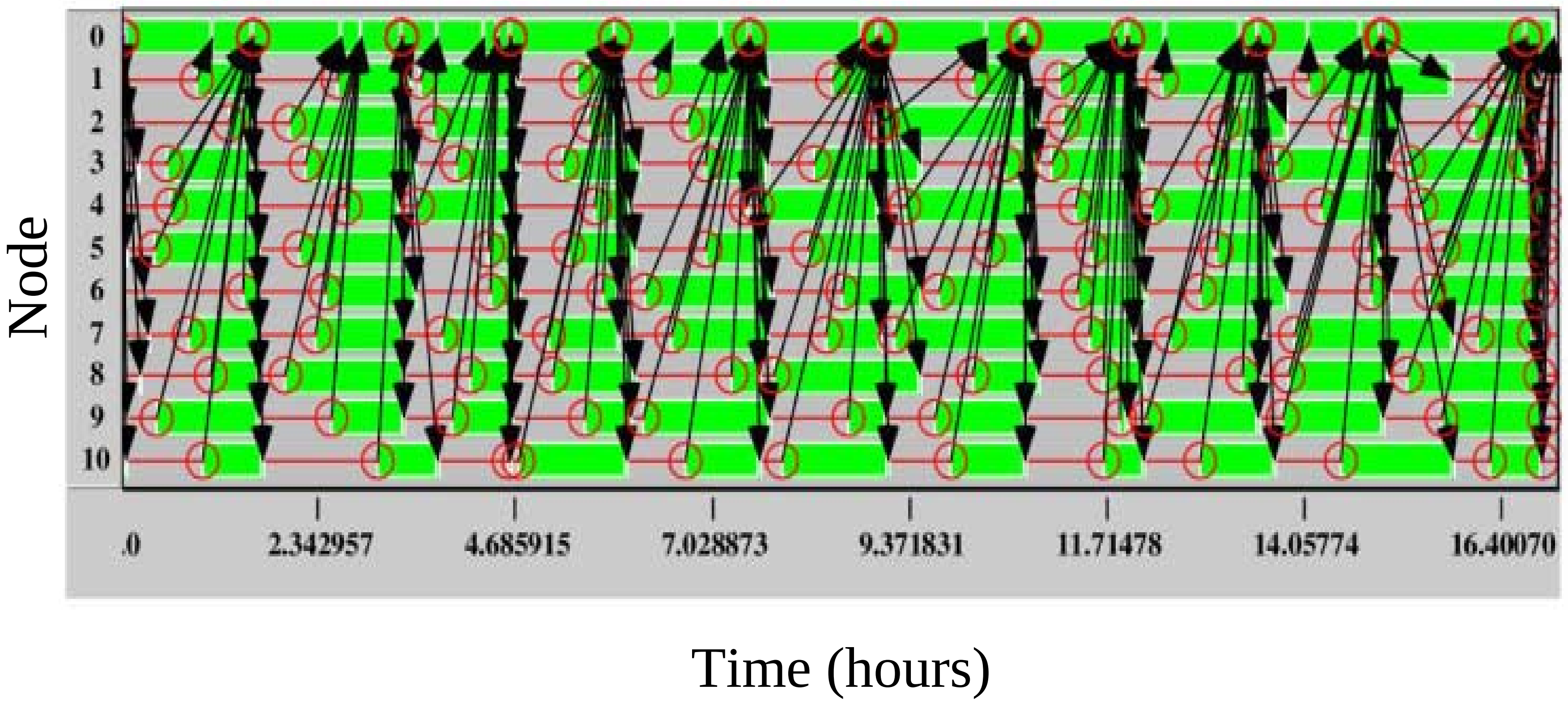
FEM problem solving efficiency:

- Parallel optimization algorithms allows:
  - Higher throughput:
    - Solving more complex problems in the same timespan.
    - Ability to solve previously intractable problems.
  - More sophisticated finite element formulations
  - Higher accuracy (mesh densities)

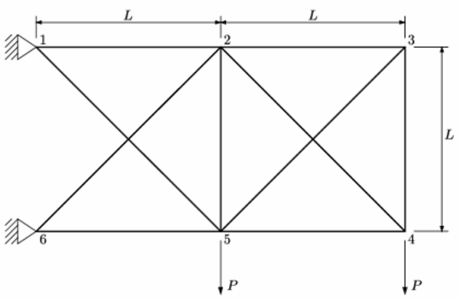
Parallelization Speedup

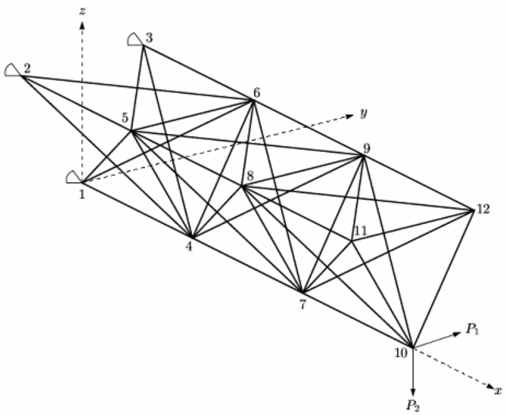


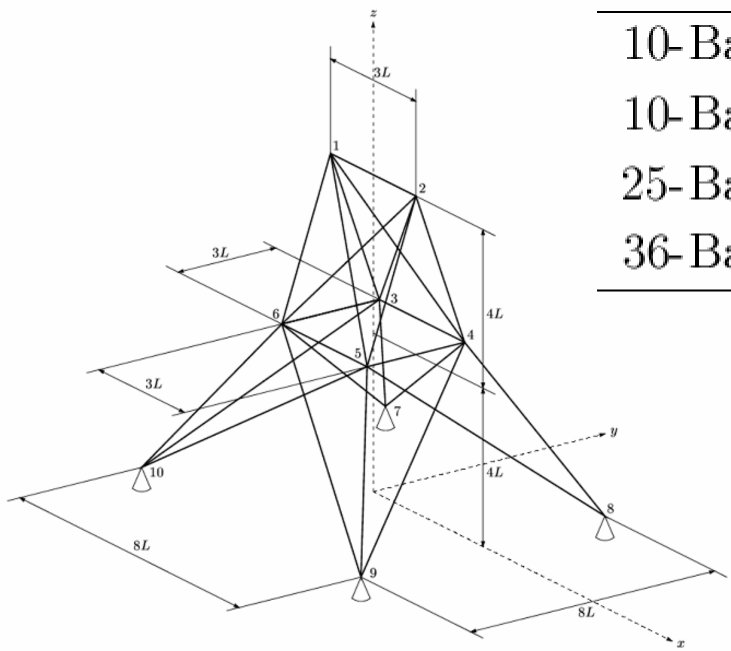
Parallel PSO Network Communication



PSO on structural sizing problems







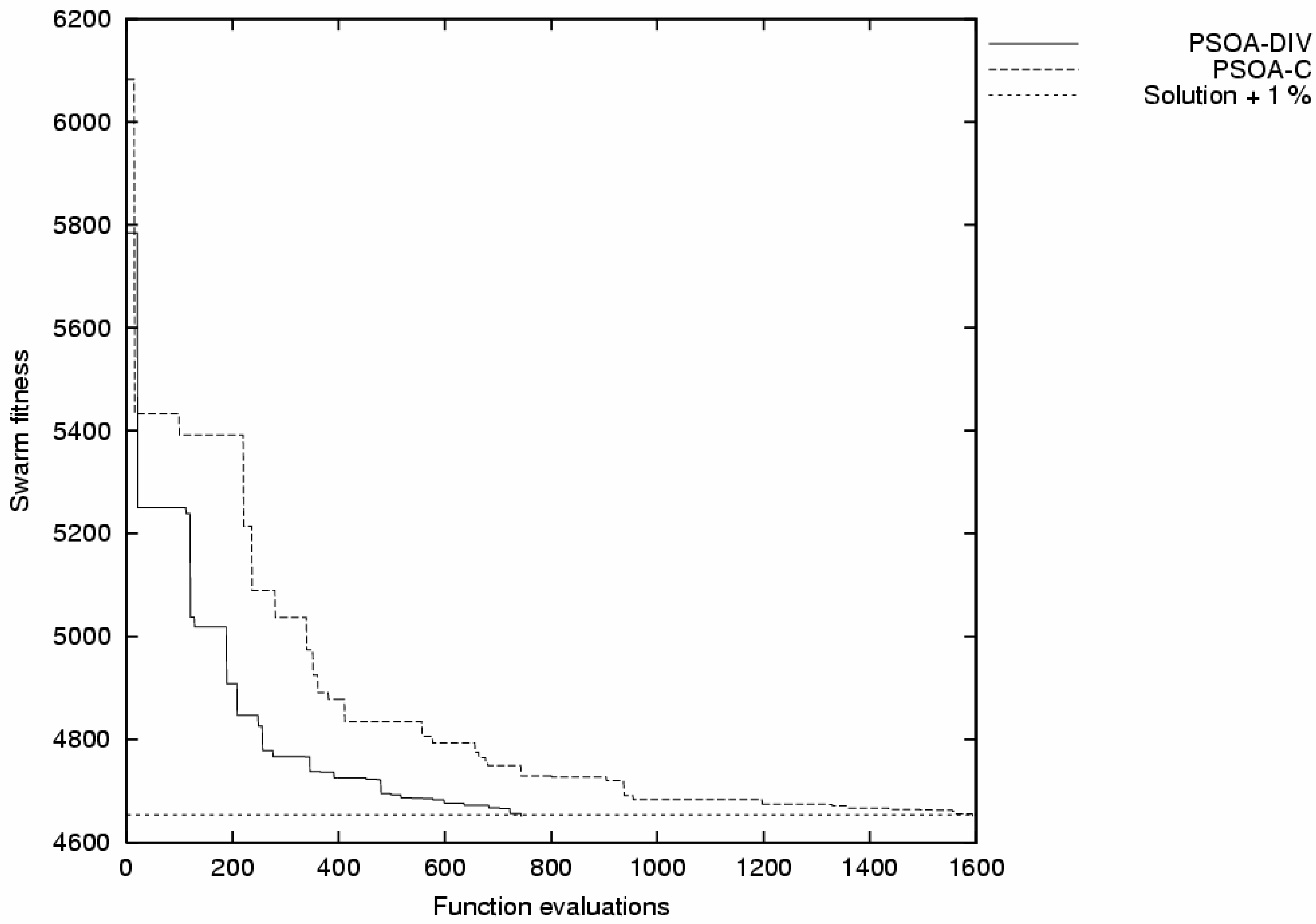
Problem Name	Problem Nature	n	m
10-Bar	Convex	10	32
10-Bar	Non-Convex	10	34
25-Bar	Non-Convex	8	84
36-Bar	Convex	21	95

Accommodation of constraints

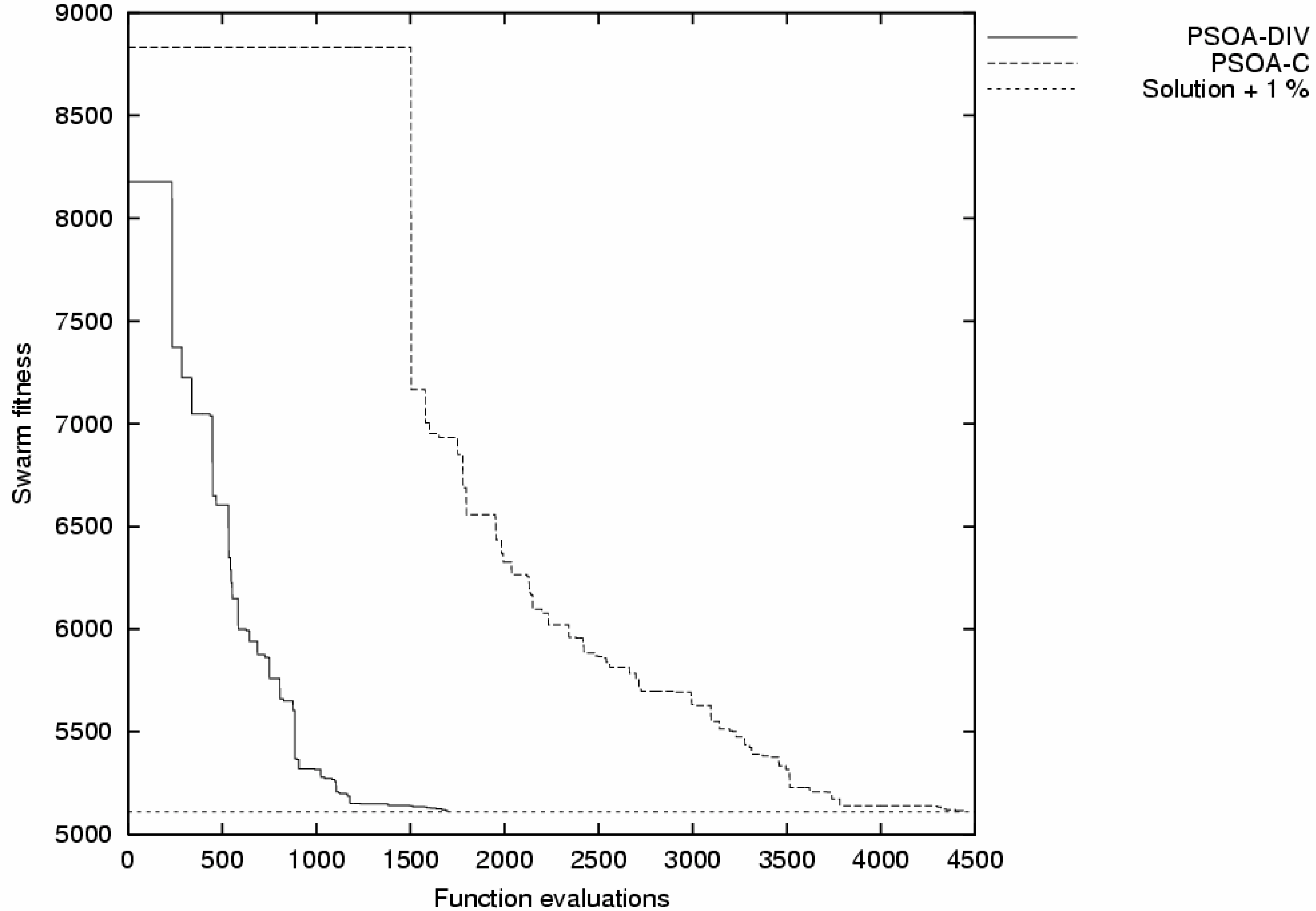
$$\tilde{f} = f(\boldsymbol{x}) + \sum_{j=1}^{m} \lambda_j [g_j(\boldsymbol{x})]^2 \mu_j(g_j) \ ,$$

$$\text{with} \quad \mu_j(g_j) = \begin{cases} 0 & \text{if} \quad g_j(\boldsymbol{x}) \le 0 \\ 1 & \text{if} \quad g_j(\boldsymbol{x}) > 0 \end{cases} \quad ,$$

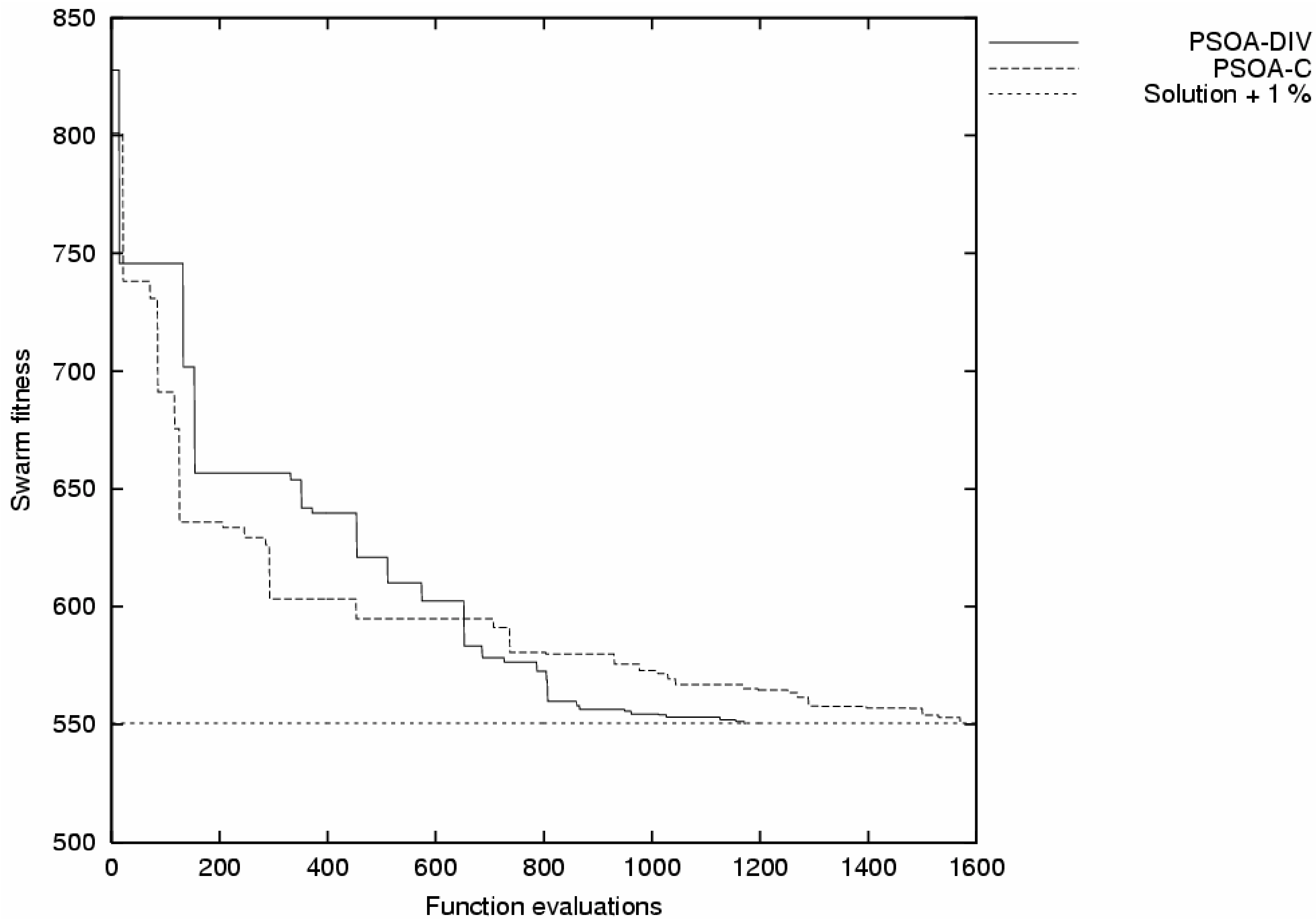
and penalty parameters $\lambda_j > 0$, prescribed.



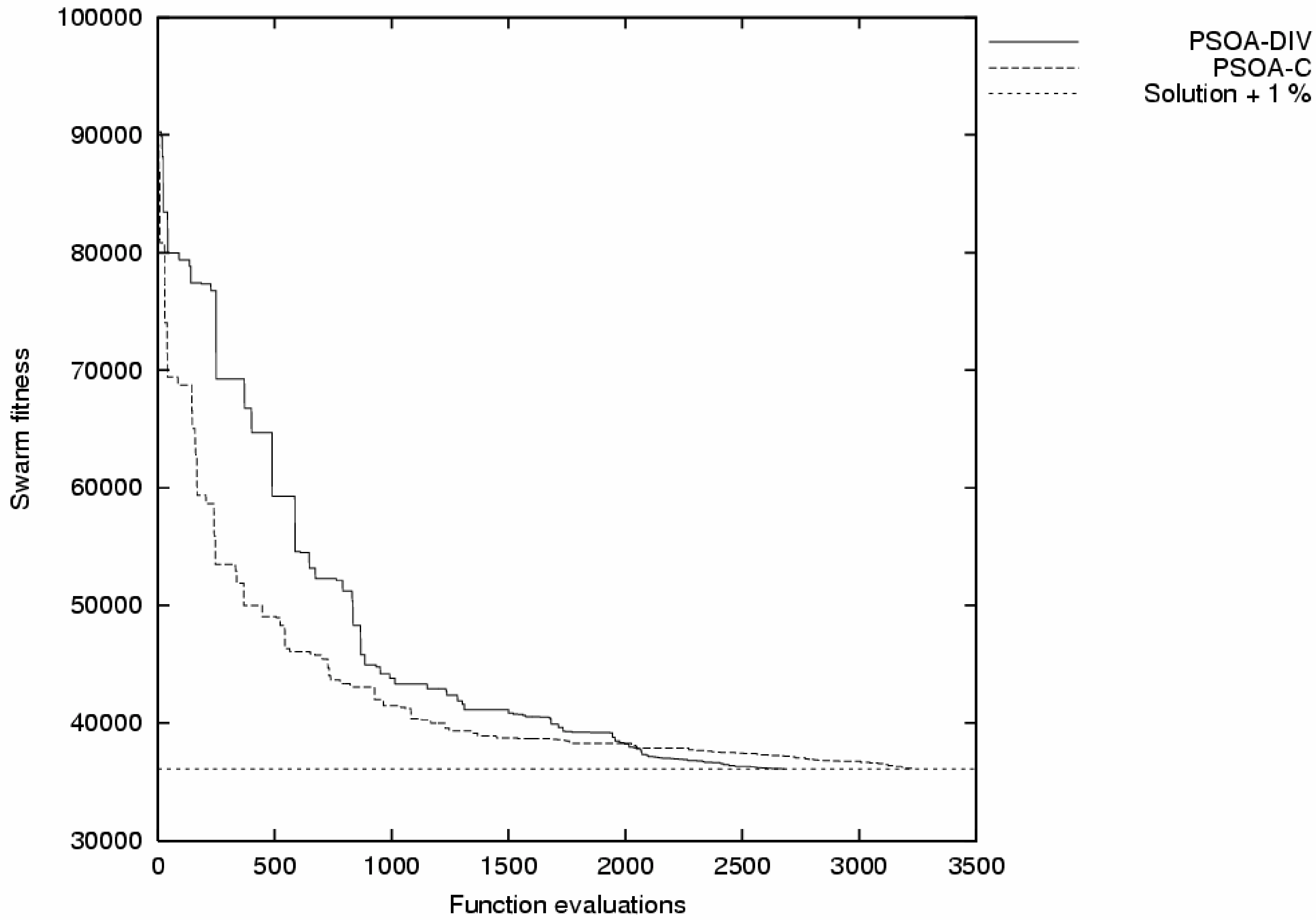
Convex 10-bar truss



Non-convex 10-bar truss



Non-convex 25-bar truss



Non-convex 36-bar truss

Concluding remarks

- The PSO is a is an efficient global optimizer for continuous variable problems (structural applications)
- Easily implemented, with very little parameters to fine-tune
- Algorithm modifications improve PSO local search ability
- Can accommodate constraints by using a penalty method

References

- Carlisle, A., and Dozier, G. (2001). An off-the-shelf PSO. *Proceedings of the Workshop on Particle Swarm Optimization*. Indianapolis, IN: Purdue School of Engineering and Technology, IUPUI (in press).
- Clerc, M. (1999). The swarm and the queen: towards a deterministic and adaptive particle swarm optimization. *Proc. 1999 Congress on Evolutionary Computation*, Washington, DC, pp 1951-1957. Piscataway, NJ: IEEE Service Center.
- Eberhart, R. C., and Hu, X. (1999). Human tremor analysis using particle swarm optimization. *Proc. Congress on Evolutionary Computation 1999*, Washington, DC, pp 1927–1930. Piscataway, NJ: IEEE Service Center.
- Eberhart, R. C., and Kennedy, J. (1995). A new optimizer using particle swarm theory. *Proceedings of the Sixth International Symposium on Micro Machine and Human Science*, Nagoya, Japan, 39-43. Piscataway, NJ: IEEE Service Center.
- Eberhart, R. C., Simpson, P. K., and Dobbins, R. W. (1996). *Computational Intelligence PC Tools*. Boston, MA: Academic Press Professional.
- Eberhart, R. C., and Shi, Y. (1998)(a). Evolving artificial neural networks. *Proc. 1998 Int'l. Conf. on Neural Networks and Brain*, Beijing, P.R.C., PL5-PL13.
- Eberhart, R. C. and Shi, Y. (1998)(b). Comparison between genetic algorithms and particle swarm optimization. In V. W. Porto, N. Saravanan, D. Waagen, and A. E. Eiben, Eds. *Evolutionary Programming VII: Proc. 7th Ann. Conf. on Evolutionary Programming Conf.*, San Diego, CA. Berlin: Springer-Verlag.
- Eberhart, R. C., and Shi, Y. (2000). Comparing inertia weights and constriction factors in particle swarm optimization. *Proc. Congress on Evolutionary Computation 2000*, San Diego, CA, pp 84-88.
- Eberhart, R. C., and Shi, Y. (2001)(a). Tracking and optimizing dynamic systems with particle swarms. *Proc. Congress on Evolutionary Computation 2001*, Seoul, Korea. Piscataway, NJ: IEEE Service Center. (in press)
- Eberhart, R. C., and Shi, Y. (2001)(b). Particle swarm optimization: developments, applications and resources. *Proc. Congress on Evolutionary Computation 2001*, Seoul, Korea. Piscataway, NJ: IEEE Service Center. (in press)
- Fan, H.-Y., and Shi, Y. (2001). Study of Vmax of the particle swarm optimization algorithm. *Proceedings of the Workshop on Particle Swarm Optimization*. Indianapolis, IN: Purdue School of Engineering and Technology, IUPUI (in press).
- Fukuyama Y., Yoshida, H. (2001). A Particle Swarm Optimization for Reactive Power and Voltage Control in Electric Power Systems, *Proc. Congress on Evolutionary Computation 2001*, Seoul, Korea. Piscataway, NJ: IEEE Service Center. (in press)
- He, Z.,Wei, C., Yang, L., Gao, X., Yao, S., Eberhart, R., and Shi, Y. (1998). Extracting rules from fuzzy neural network by particle swarm optimization, *Proc. IEEE International Conference on Evolutionary Computation,* Anchorage, Alaska, USA
- Kennedy, J. (1997). The particle swarm: social adaptation of knowledge. *Proc. Intl. Conf. on Evolutionary Computation*, Indianapolis, IN, 303-308. Piscataway, NJ: IEEE Service Center.
- Kennedy, J. (1998). Methods of agreement: inference among the eleMentals. *Proc. 1998 Intl. Symp. on Intelligent Control*. Piscataway, NJ: IEEE Service Center.
- Kennedy, J. (1998). The behavior of particles. In V. W. Porto, N. Saravanan, D. Waagen, and A. E. Eiben, Eds. *Evolutionary Programming VII: Proc. 7th Ann. Conf. on Evolutionary Programming Conf.*, San Diego, CA, 581–589. Berlin: Springer-Verlag.
- Kennedy, J. (1998). Thinking is social: experiments with the adaptive culture model. *Journal of Conflict Resolution.* 42(1), 56–76.
- Kennedy, J. (1999). Small worlds and mega-minds: effects of neighborhood topology on particle swarm performance. *Proc. Congress on Evolutionary Computation 1999*, 1931–1938. Piscataway, NJ: IEEE Service Center.
- Kennedy, J. (2000). Stereotyping: improving particle swarm performance with cluster analysis. *Proc. of the 2000 Congress on Evolutionary Computation*, San Diego, CA. Piscataway, NJ: IEEE Press.

References (continued)

- Kennedy, J. (2001). Out of the computer, into the world: externalizing the particle swarm. *Proceedings of the Workshop on Particle Swarm Optimization*. Indianapolis, IN: Purdue School of Engineering and Technology, IUPUI (in press).
- Kennedy, J. and Eberhart, R. C. (1995). Particle swarm optimization. *Proc. IEEE Int'l. Conf. on Neural Networks*, *IV*, 1942–1948. Piscataway, NJ: IEEE Service Center.
- Kennedy, J. and Eberhart, R. C. (1997). A discrete binary version of the particle swarm algorithm. *Proc. 1997 Conf. on Systems, Man, and Cybernetics*, 4104–4109. Piscataway, NJ: IEEE Service Center.
- Kennedy, J., and Eberhart, R. C. (1999). The particle swarm: social adaptation in information processing systems. In Corne, D., Dorigo, M., and Glover, F., Eds., *New Ideas in Optimization*. London: McGraw-Hill.
- Kennedy, J., Eberhart, R. C., and Shi, Y. (2001). *Swarm Intelligence*, San Francisco: Morgan Kaufmann Publishers.
- Kennedy, J. and Spears, W. M. (1998). Matching algorithms to problems: an experimental test of the particle swarm and some genetic algorithms on the multimodal problem generator. *Proc. Intl. Conf. on Evolutionary Computation*, 78–83. Piscataway, NJ: IEEE Service Center.
- Mohan, C. K., and Al-kazemi, B. (2001). Discrete particle swarm optimization. *Proceedings of the Workshop on Particle Swarm Optimization*. Indianapolis, IN: Purdue School of Engineering and Technology, IUPUI (in press).
- Naka, S., Grenji, T., Yura, T., Fukuyama, Y. (2001).
- Practical Distribution State Estimation Using Hybrid Particle Swarm Optimization*, Proc. of IEEE PES Winter Meeting*, Columbus, Ohio, USA.
- Ozcan, E., and Mohan, C. (1999). Particle swarm optimization: surfing the waves. *Proc. 1999 Congress on Evolutionary Computation*, 1939–1944. Piscataway, NJ: IEEE Service Center.
- Parsopoulos, K. E., Plagianakos, V. P., Magoulas, G. D. and Vrahatis, M. N. (2001). Stretching technique for obtaining global minimizers through particle swarm optimization. *Proceedings of the Workshop on Particle Swarm Optimization*. Indianapolis, IN: Purdue School of Engineering and Technology, IUPUI (in press).
- Secrest, B. R., and Lamont, G. B. (2001). Communication in particle swarm optimization illustrated by the traveling salesman problem. *Proceedings of the Workshop on Particle Swarm Optimization*. Indianapolis, IN: Purdue School of Engineering and Technology, IUPUI (in press).
- Shi, Y. and Eberhart, R. C. (1998a). Parameter selection in particle swarm optimization. In *Evolutionary Programming VII: Proc. EP98*, New York: Springer-Verlag, pp. 591-600.
- Shi, Y. and Eberhart, R. C. (1998b). A modified particle swarm optimizer. *Proceedings of the IEEE International Conference on Evolutionary Computation*, 69-73. Piscataway, NJ: IEEE Press.
- Shi, Y. and Eberhart, R. C. (1999). Empirical study of particle swarm optimization. *Proceedings of the 1999 Congress on Evolutionary Computation*, 1945----–1950. Piscataway, NJ: IEEE Service Center.
- Shi, Y. and Eberhart, R., (2000). Experimental study of particle swarm optimization. *Proc. SCI2000 Conference*, Orlando, FL.
- Shi, Y. and Eberhart, R., (2001a). Fuzzy Adaptive Particle Swarm Optimization, *Proc. Congress on Evolutionary Computation 2001*, Seoul, Korea. Piscataway, NJ: IEEE Service Center. (in press)
- Shi, Y. and Eberhart, R., (2001b). Particle Swarm Optimization with Fuzzy Adaptive Inertia Weight*, Proceedings of the Workshop on Particle Swarm Optimization.* Indianapolis, IN: Purdue School of Engineering and Technology, IUPUI (in press).
- Suganthan, P. N. (1999). Particle swarm optimiser with neighbourhood operator. *Proceedings of the 1999 Congress on Evolutionary Computation*, 1958----–1962. Piscataway, NJ: IEEE Service Center.
- Tandon, V. (2000). Closing the gap between CAD/CAM and optimized CNC end milling. Master's thesis, Purdue School of Engineering and Technology, Indiana University Purdue University Indianapolis.
- Yoshida, H., Kawata, K., Fukuyama, Y., and Nakanishi, Y. (1999). A particle swarm optimization for reactive power and voltage control considering voltage stability. In G. L. Torres and A. P. Alves da Silva, Eds., *Proc. Intl. Conf. on Intelligent System Application to Power Systems*, Rio de Janeiro, Brazil, 117–121.