Particle Swarm Optimization

- an introduction and its recent developments

A tutorial prepared for GECCO'07

Dr. Xiaodong Li, School of Computer Science and IT,
RMIT University, Melbourne, Australia

Professor Andries P. Engelbrecht, Department of Computer Science,
University of Pretoria, Pretoria, South Africa

ion

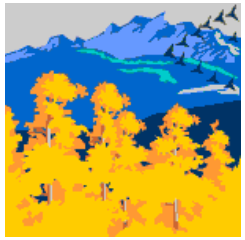
Outline

- Swarm Intelligence
  - Background; origin; links to EC.
- Introduction to PSO
  - Original PSO, Inertia weight, constriction coefficient
- Particle Trajectories
  - Simplified PSO; one or two particles
- Convergence aspects
- FIPS, Bare-bones, and other PSO variants
- Communication topologies
- Speciation and niching methods in PSO
- PSO for optimization in dynamic environments
- PSO for multiobjective optimization
- PSO for constrained optimization
- References

Swarm Intelligence

GECCO'07, July 7–11, 2007, London, England, United Kingdom.
ACM 978-1-59593-698-1/07/0007.

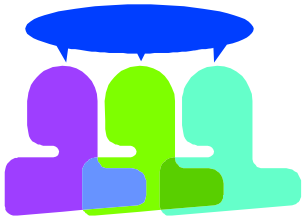
Swarm Intelligence



Swarm intelligence (SI) is an artificial intelligence technique based around the study of collective behavior in decentralized, self-organized systems.

SI systems are typically made up of a population of simple agents interacting locally with one another and with their environment. Although there is normally no centralized control structure dictating how individual agents should behave, local interactions between such agents often lead to the emergence of global behavior. Examples of systems like this can be found in nature, including ant colonies, bird flocking, animal herding, bacteria molding and fish schooling (from *Wikipedia*).

Swarm Intelligence



Mind is social…

Human intelligence results from social interaction:
Evaluating, comparing, and imitating one another, learning from experience and emulating the successful behaviours of others, people are able to adapt to complex environments through the discovery of relatively optimal patterns of attitudes, beliefs, and behaviours. (Kennedy & Eberhart, 2001).

Culture and cognition are inseparable consequences of human sociality:
Culture emerges as individuals become more similar through mutual social learning. The sweep of culture moves individuals toward more adaptive patterns of thought and behaviour.

ion

Swarm Intelligence

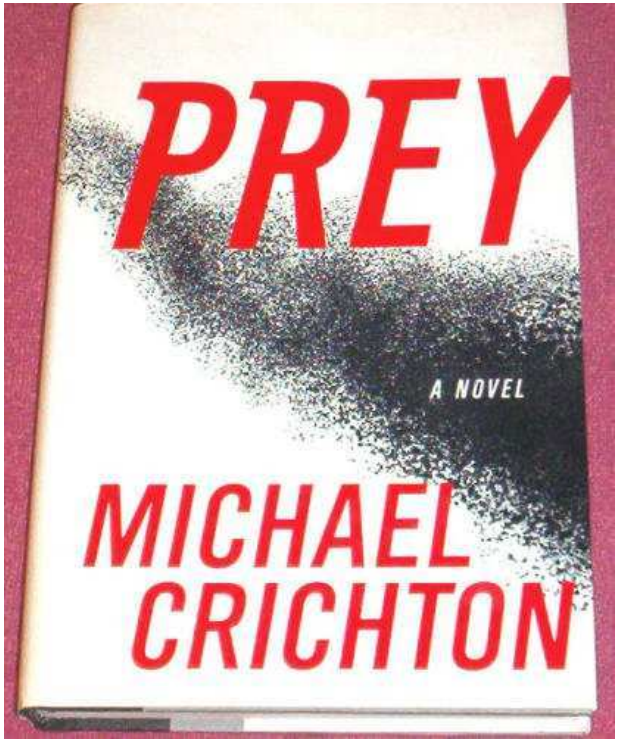
To model human intelligence, we should model individuals in a social context, interacting with one another.

Swarm Intelligence applications

- **Swarm-bots,** an EU project led by Marco Dorigo, aimed to study new approaches to the design and implementation of self-organizing and self-assembling artifacts (http://www.swarm-bots.org/).
- A 1992 paper by M. Anthony Lewis and George A. Bekey discusses the possibility of using swarm intelligence to control **nanobots** within the body for the purpose of killing cancer tumours.
- Artists are using swarm technology as a means of creating complex interactive environments.
  - Disney's ***The Lion King*** was the first movie to make use of swarm technology (the stampede of the bisons scene).
  - The movie "***Lord of the Rings***" has also made use of similar technology during battle scenes.

(Some examples from *Wikipedia*)

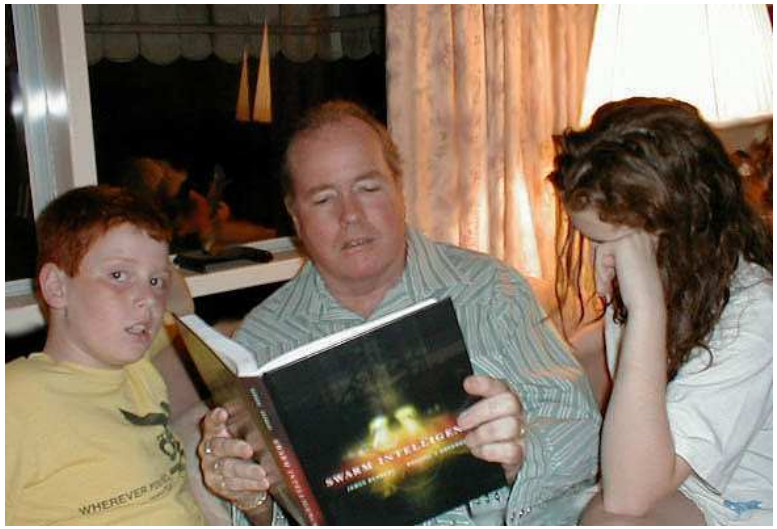
Novel about swarm



"… *Within hours of his arrival at the remote testing center, Jack discovers his wife's firm has created self-replicating nanotechnology--a literal swarm of microscopic machines. Originally meant to serve as a military eye in the sky, the swarm has now escaped into the environment and is seemingly intent on killing the scientists trapped in the facility.*" (Michael Crichton, 2002)

Particle Swarm Optimization

The inventors:



James Kennedy

Russell Eberhart

ion

Particle Swarm Optimization

PSO has its roots in Artificial Life and social psychology, as well as engineering and computer science.

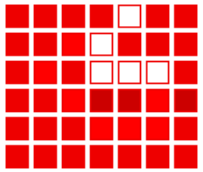
The particle swarms in some way are closely related to cellular automata (CA):

a) individual cell updates are done in parallel

b) each new cell value depends only on the old values of the cell and its neighbours, and

c) all cells are updated using the same rules (Rucker, 1999).

Blinker

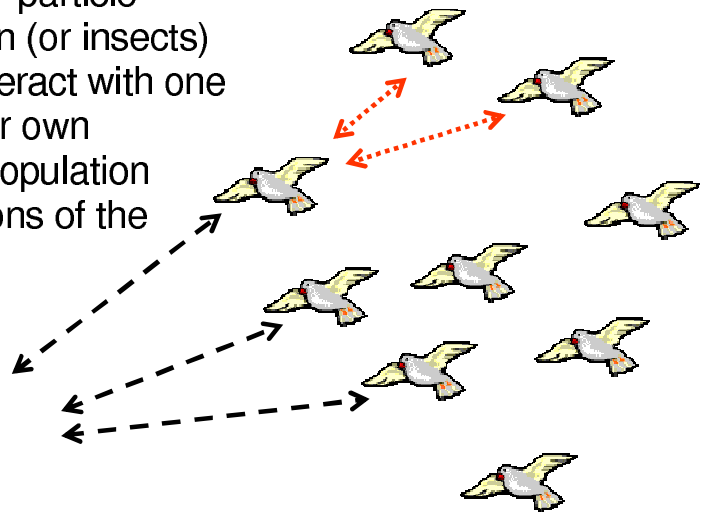


Glider

Individuals in a particle swarm can be conceptualized as cells in a CA, whose states change in many dimensions simultaneously.

Particle Swarm Optimization

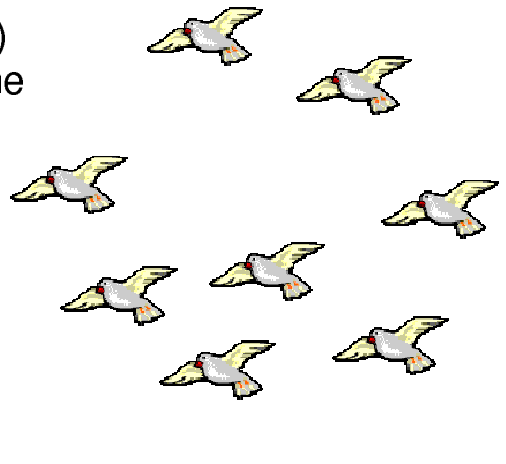
As described by the inventers James Kennedy and Russell Eberhart, “particle swarm algorithm imitates human (or insects) social behaviour. Individuals interact with one another while learning from their own experience, and gradually the population members move into better regions of the problem space”.



Why named as “particle”, not “points”? Both Kennedy and Eberhart felt that velocities and accelerations are more appropriately applied to particles.

Particle Swarm Optimization

As described by the inventers James Kennedy and Russell Eberhart, “particle swarm algorithm imitates human (or insects) social behaviour. Individuals interact with one another while learning from their own experience, and gradually the population members move into better regions of the problem space”.



Why named as “particle”, not “points”? Both Kennedy and Eberhart felt that velocities and accelerations are more appropriately applied to particles.

PSO Precursors

Reynolds (1987)'s simulation **Boids** – a simple flocking model consists of three simple local rules:

- **Collision avoidance**: pull away before they crash into one another;
- **Velocity matching**: try to go about the same speed as their neighbours in the flock;
- **Flock centering**: try to move toward the center of the flock as they perceive it.

A demo: http://www.red3d.com/cwr/boids/
With just the above 3 rules, **Boids** show very realistic flocking behaviour.

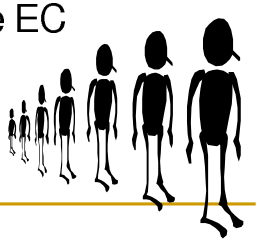
Heppner (1990) interests in rules that enabled large numbers of birds to flock synchronously.

ion

Its links to Evolutionary Computation

"*In theory at least, individual members of the school can profit from the discoveries and previous experience of all other members of the school during the search for food. This advantage can become decisive, outweighing the disadvantages of competition for food items, whenever the resource is unpredictably distributed in patches*" (by Sociobiologist E. O. Wilson)

- Both PSO and EC are population based.
- PSO also uses the fitness concept, but, less-fit particles do not die. No "survival of the fittest".
- No evolutionary operators such as crossover and mutation.
- Each particle (candidate solution) is varied according to its past experience and relationship with other particles in the population.
- Having said the above, there are hybrid PSOs, where some EC concepts are adopted, such as selection, mutation, etc.



PSO applications

Problems with continuous, discrete, or mixed search space, with multiple local minima; problems with constraints; multiobjective, dynamic optimization.

- Evolving neural networks:
  - Human tumor analysis;
  - Computer numerically controlled milling optimization;
  - Battery pack state-of-charge estimation;
  - Real-time training of neural networks (Diabetes among Pima Indians);
  - Servomechanism (time series prediction optimizing a neural network);
- Reactive power and voltage control;
- Ingredient mix optimization;
- Pressure vessel (design a container of compressed air, with many constraints);
- Compression spring (cylindrical compression spring with certain mechanical characteristics);
- Moving Peaks (multiple peaks dynamic environment); and more

PSO can be tailor-designed to deal with specific real-world problems.

Original PSO

$$\vec{v}_i \leftarrow \vec{v}_i + \vec{\varphi}_1 \otimes (\vec{p}_i - \vec{x}_i) + \vec{\varphi}_2 \otimes (\vec{p}_g - \vec{x}_i)$$

$$\vec{x}_i \leftarrow \vec{x}_i + \vec{v}_i$$

$\vec{x}_i$ denotes the current position of the *i*–th particle in the swarm;

$\vec{v}_i$ denotes the velocity of the *i*-th particle;

$\vec{p}_i$ the best position found by the *i*-th particle so far, i.e., personal best;

$\vec{p}_g$ the best position found from the particle's neighbourhood, i.e., global best;

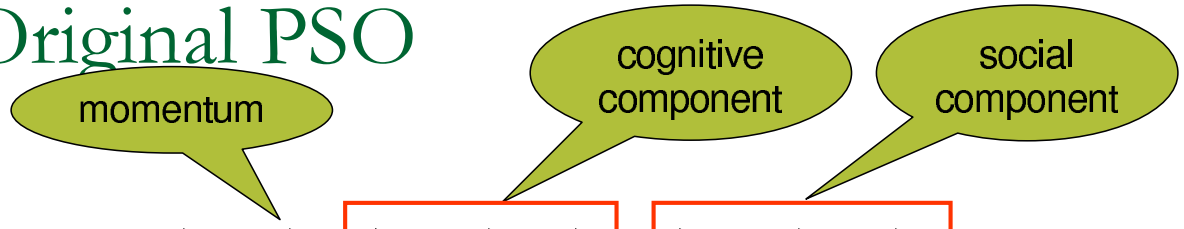
The symbol $\otimes$ denotes a point-wise vector multiplication;

$\vec{\varphi}_1 = c_1\vec{r}_1$ and $\vec{\varphi}_2 = c_2\vec{r}_2$;

$\vec{r}_1$ and $\vec{r}_2$ are two vectors of random numbers uniformly chosen from [0, 1];

c_1 and c_2 are acceleration coefficients.

Original PSO



$$\vec{v}_i \leftarrow \vec{v}_i + \vec{\varphi}_1 \otimes (\vec{p}_i - \vec{x}_i) + \vec{\varphi}_2 \otimes (\vec{p}_g - \vec{x}_i)$$

$$\vec{x}_i \leftarrow \vec{x}_i + \vec{v}_i$$

Velocity $\vec{v}_i$ (which denotes the amount of change) of the *i*-th particle is determined by three components:

- **momentum** – previous velocity term to carry the particle in the direction it has travelled so far;
- **cognitive** component – tendency to return to the best position visited so far;
- **social** component – tendency to be attracted towards the best position found in its neighbourhood.

Neighbourhood topologies can be used to control information propagation between particles, e.g., ring, star, or von Neumann. ***lbest*** and ***gbest*** PSOs.

ion

Pseudo-code of a basic PSO

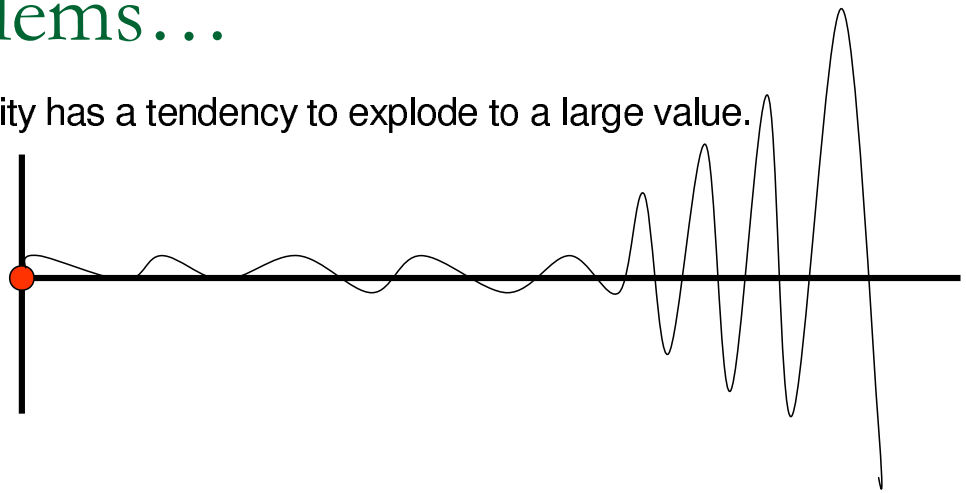
```
Randomly generate an initial population
repeat
    for i = 1 to population_size do
        if f(x_i) < f(p_i) then p_i = x_i;
        p_g = min(p_neighbours);
        for d =1 to dimensions do
            velocity_update();
            position_update();
        end
    end
until termination criterion is met.
```

Synchronous vs Asynchronous

- Synchronous updates
  - Personal best and neighborhood bests updated separately from position and velocity vectors
  - Slower feedback about best positions
  - Better for ***gbest*** PSO
- Asynchronous updates
  - New best positions updated after each particle position update
  - Immediate feedback about best regions of the search space
  - Better for ***lbest*** PSO

Problems…

The velocity has a tendency to explode to a large value.



To prevent it, a parameter **Vmax** can be used. Basically if the velocity value exceeds ±**Vmax**, it gets reset to ±**Vmax** accordingly.

This velocity clamping does not necessarily prevent particles from leaving the search space, nor to converge. However, it does limit the particle step size, therefore restricting particles from further divergence.

Inertia weight

The $\vec{p}_i$ and $\vec{p}_g$ can be collapsed into a single term $\vec{p}$ without losing any information:

$$\vec{v}_i \leftarrow \vec{v}_i + \vec{\varphi} \otimes (\vec{p} - \vec{x}_i)$$

$$\vec{x}_i \leftarrow \vec{x}_i + \vec{v}_i$$

where $\vec{\varphi} = \vec{\varphi}_1 + \vec{\varphi}_2$ and $\vec{p} = \dfrac{\vec{\varphi}_1 \otimes \vec{p}_i + \vec{\varphi}_2 \otimes \vec{p}_g}{\vec{\varphi}_1 + \vec{\varphi}_2}$

$\vec{p}$ represents the weighted average of $\vec{p}_i$ and $\vec{p}_g$. Note that the division operator is a point-wise vector division.

Since the velocity term tends to keep the particle moving in the same direction as of its previous flight, a coefficient inertia weight, ***w***, can be used to control this influence:

$$\vec{v}_i \leftarrow \boldsymbol{w}\, \vec{v}_i + \vec{\varphi}_1 \otimes (\vec{p}_i - \vec{x}_i) + \vec{\varphi}_2 \otimes (\vec{p}_g - \vec{x}_i)$$

The inertia weighted PSO can converge under certain conditions without using **Vmax**.

ion

Inertia weight

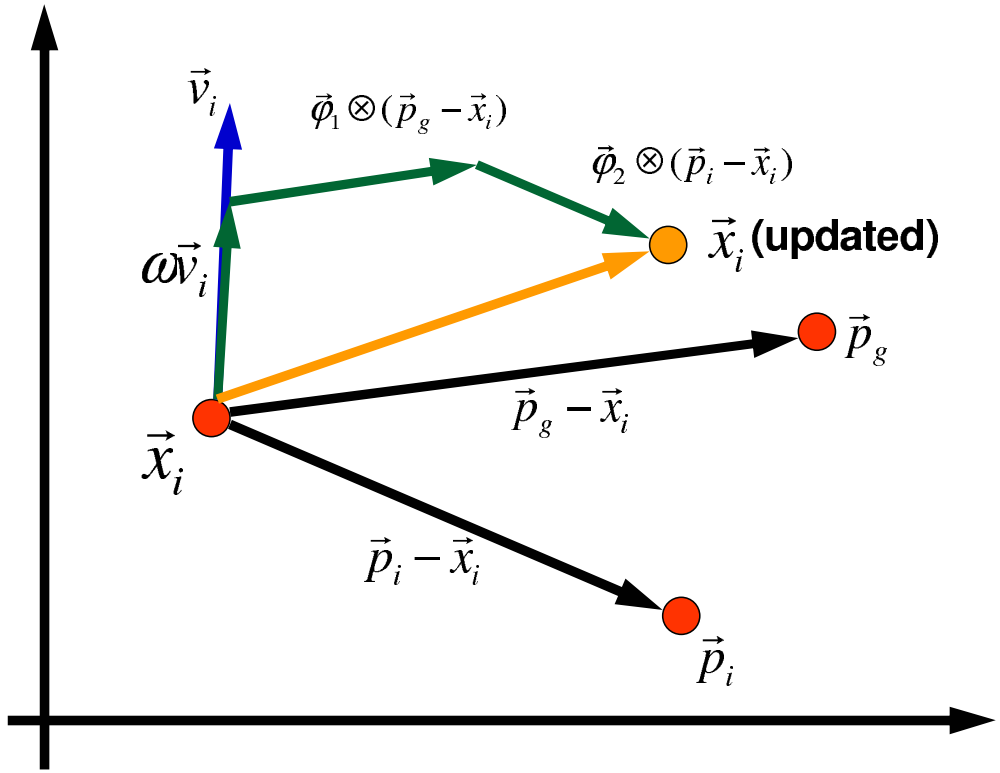
The inertia weight can be used to control exploration and exploitation:

For $w \geq 1$: velocities increase over time, swarm diverge;
For $0 < w < 1$: particles decelerate; convergence depends on value for c_1 and c_2;
For $w < 0$: velocities decrease over time, eventually reaching 0; convergence behaviour.

Empirical results suggest that a constant inertia weight $w = 0.7298$ and $c_1=c_2=1.49618$ provide good convergence behaviour.

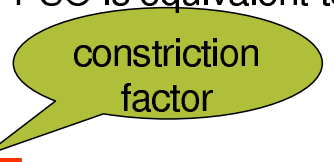
Eberhart and Shi also suggested to use the inertia weight which decreasing over time, typically from 0.9 to 0.4. It has the effect of narrowing the search, gradually changing from an exploratory to an exploitative mode.

Visualizing PSO



Constriction PSO

Clerc and Kennedy (2000) suggested a general PSO, where a constriction coefficient χ is applied to both terms of the velocity formula. The Constriction Type 1'' PSO is equivalent to the inertia weighted PSO:



$$\vec{v}_i \leftarrow \chi\,(\vec{v}_i + \vec{\varphi}_1 \otimes (\vec{p}_i - \vec{x}_i) + \vec{\varphi}_2 \otimes (\vec{p}_g - \vec{x}_i))$$

$$\vec{x}_i \leftarrow \vec{x}_i + \vec{v}_i$$

where $\chi = \dfrac{2k}{|\,2 - \varphi - \sqrt{\varphi^2 - 4\varphi}\,|}$ and $\varphi = \varphi_1 + \varphi_2$ with $\varphi_1 = c_1 r_1$, $\varphi_2 = c_2 r_2$.

If $\varphi \geq 4$ and k is in [0,1], then the swarm is guaranteed to converge. k controls the balance between exploration and exploitation.

Typically, k is set to 1, and $\boldsymbol{c_1}$=$\boldsymbol{c_2}$=2.05; and the constriction coefficient χ is 0.7298 (Clerc and Kennedy 2002).

Acceleration coefficients

$c_1>0$, $c_2=0$: independent hill-climbers; local search by each particle.

$c_1=0$, $c_2>0$: swarm is one stochastic hill-climber.

$c_1=c_2>0$: particles are attracted towards the average of p_i and p_g.

$c_2>c_1$: more beneficial for unimodal problems.

$c_1>c_2$: more beneficial for multimodal problems.

low c_1 and c_2: smooth particle trajectories.

high c_1 and c_2: more acceleration, abrupt movements.

Adaptive acceleration coefficients have also been proposed. For example to have c_1 and c_2 decreased over time.

ion

Particle Trajectory

Question: How important are the interactions between particles in a PSO?

To answer this question, we can study a simplified PSO, and look at scenarios where the swarm is reduced to only one or two particles. This simplified PSO assumes:

- No stochastic component;
- One dimension;
- Pre-specified initial position and velocity.

$$v \leftarrow \mathbf{w}\ v + c_1(p_i - x) + c_2(p_g - x)$$
$$x \leftarrow x + v$$

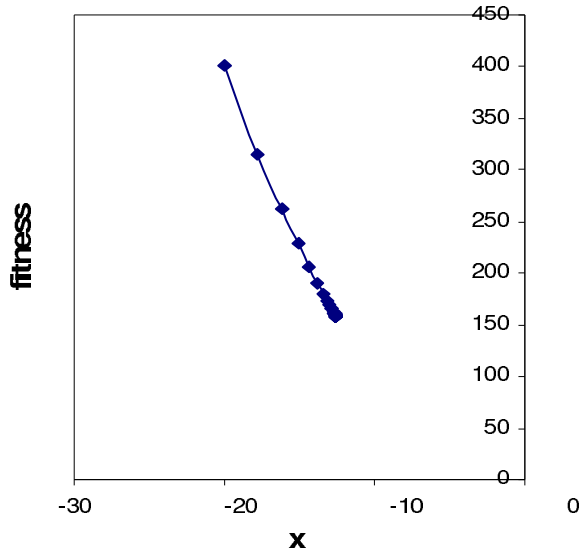
Acknowledgement: this example was taken from Clerc's recent book "Particle Swarm Optimization, with some modifications.

In the following examples, we assume ***w***=0.7, c_1=c_2=0.7. Note that even with just one particle, we actually know two positions, **x** and $\mathbf{p_i}$.

Consider the *Parabola 1D* function, $f(x) = x^2$, defined in [-20, 20]. We have two cases:

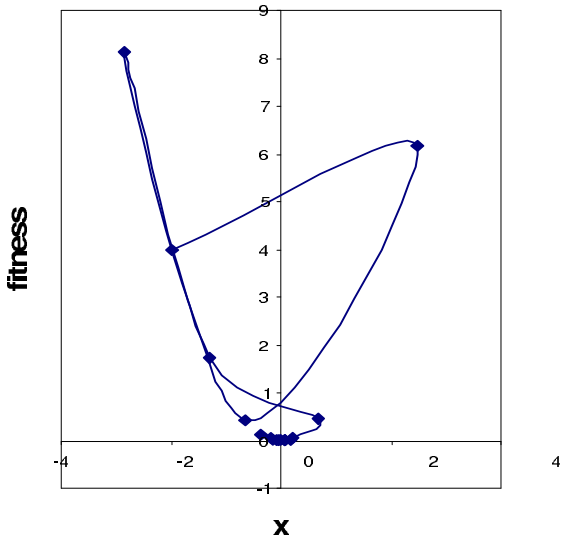
1) The first two positions are on the same side of the minimum (Initial position x= -20, v=3.2)
2) The first two positions frame the minimum (initial position x=-2, v=6.4).

Particle Trajectory (one particle)



Case 1: The first two positions are on the same side of the minimum.

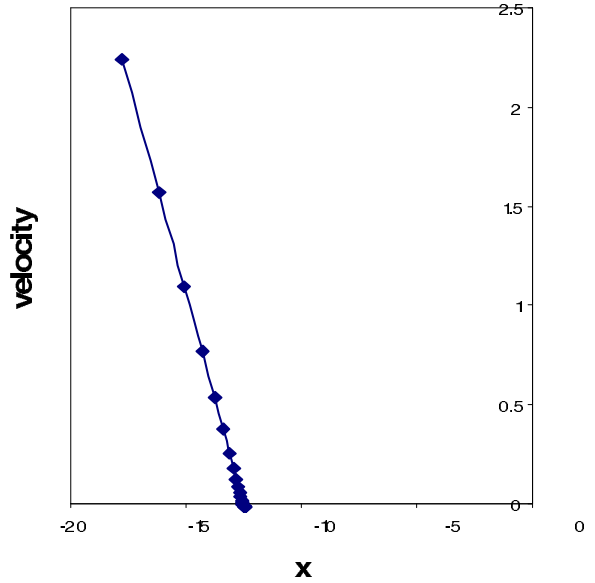
Since personal best is always equal to x, the particle is unable to reach the minimum (premature convergence).



Case 2: The first two positions frame the minimum.

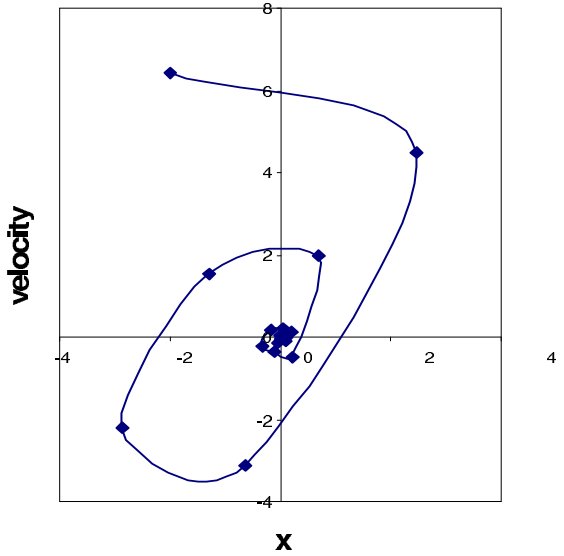
The particle oscillates around the minimum; the personal best is not always equal to x, resulting in a better convergence behaviour.

Particle Trajectory (one particle)



Case 1: The first two positions are on the same side of the minimum.

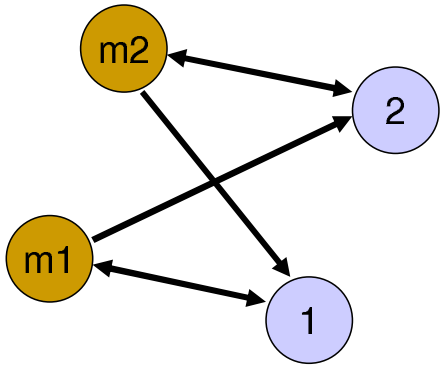
Phase space graph showing ***v*** reaches to 0 too early, resulting premature convergence



Case 2: The first two positions frame the minimum.

Phase space graph showing ***v*** in both positive and negative values (spiral converging behaviour)

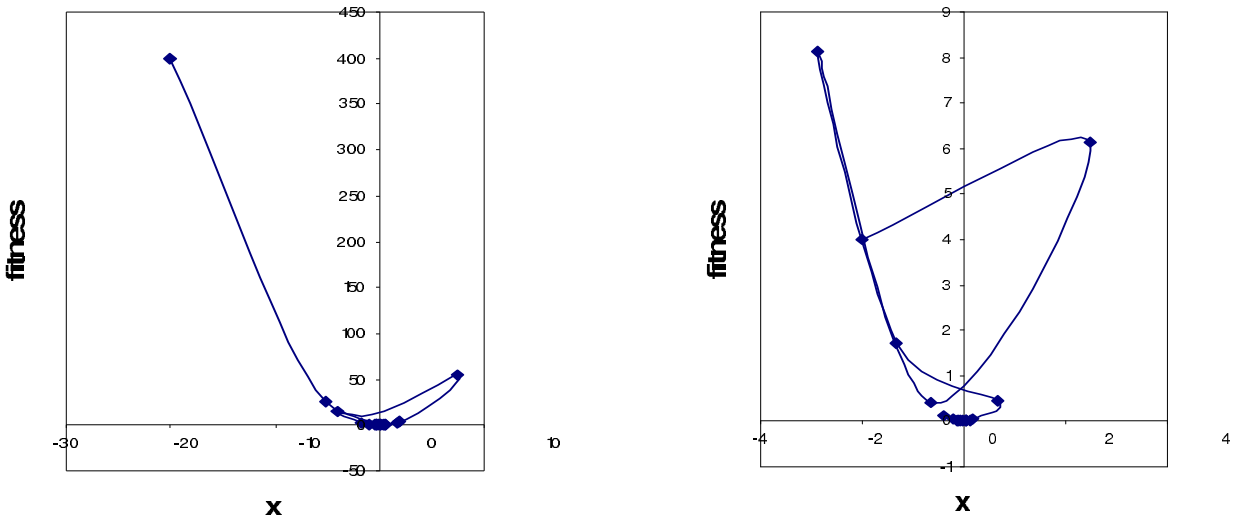
Particle Trajectory (two particles)



Graph of influence. In this case, we have two explorers and two memories. Each explorer receives information from the two memories, but informs only one (Clerc, 2006).

ion

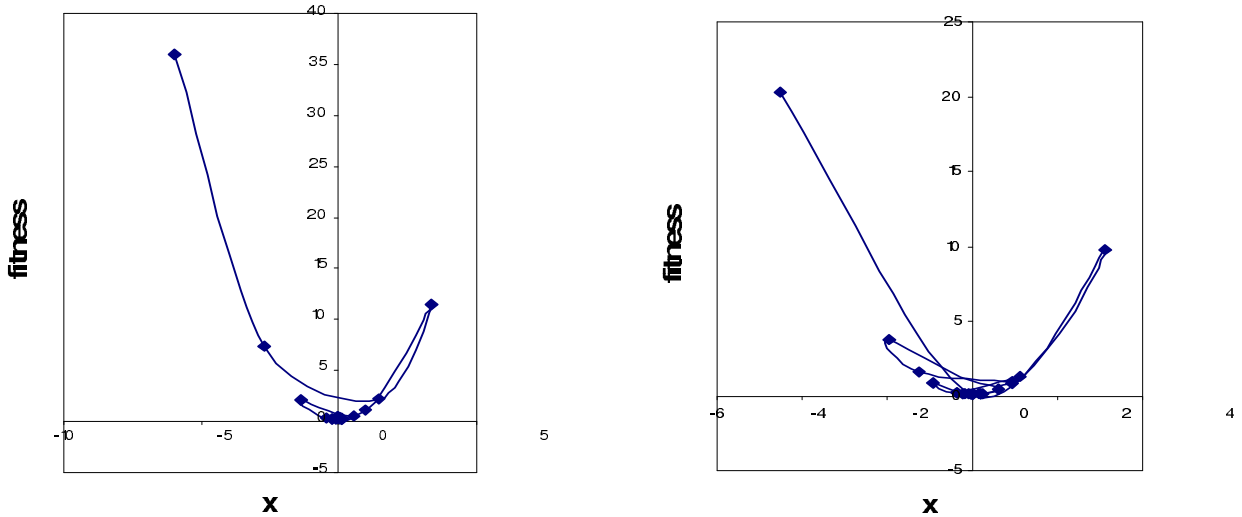
Particle Trajectory (two particles)



Now we have two particles (**two explorers and two memories**). The starting positions for the two particles are the same as in Case 1 and 2. But now the particles are working together (Clerc, 2006).

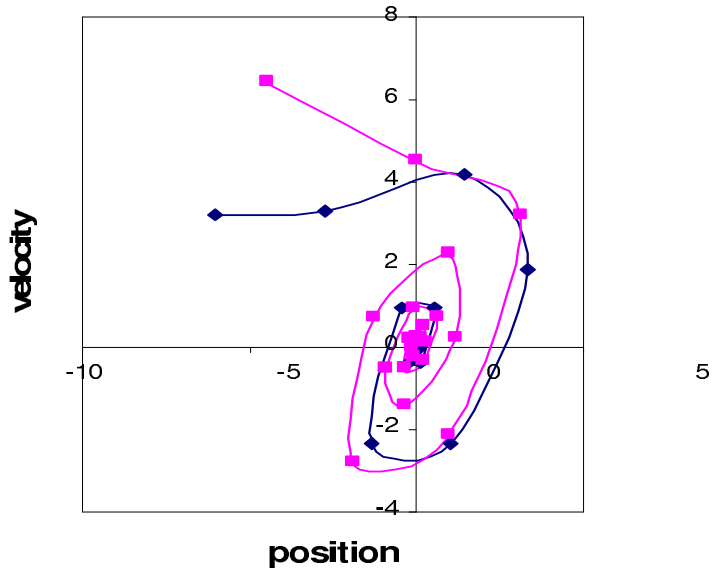
Note, however, here, memory 2 is always better than memory 1, hence the course of explorer 2 is exactly the same as seen in the previous Case 2 (Figure on the right-hand side). On the other hand, explorer 1 will benefit from the information provided by memory 2, ie., it will end up converging (Figure on the left) .

Particle Trajectory (two particles)



Two explorers and two memories. This is the more general case where each explorer is from time to time influenced by the memory of the other, when it is better than its own. Convergence is more probable, though may be slower.

Particle Trajectory (two particles)



Two explorers and two memories. Particle trajectories in the Phase space. The two particles help each other to enter and remain in the oscillatory process that allows convergence towards the optimum.

Convergence Aspects

- Formal proofs have been provided by Van den Bergh (2002), Trelea (2003), and Van den Bergh and Engelbrecht (2006) that particles converge to an equilibrium.
- In the limit, for the ***gbest*** PSO,

$$\lim_{t\to\infty} \vec{x}_i(t) = \frac{c_1\vec{p}_i(t) + c_2\vec{p}_g(t)}{c_1 + c_2}$$

- This shows that particles converge to a single point.

ion

Problem with PSO

- But, this does not mean that this weighted average of personal best and global best is a local minimum, as proven in Van den Bergh's PhD thesis (2002).
- In fact, particles may prematurely converge to a stable state.
- The original PSO is not a local optimizer, and there is no guarantee that the solution found is a local minimum.

Potential Dangerous Property

- What happens when

$$\vec{x}_i = \vec{p}_i = \vec{p}_g$$

- Then the velocity update depends only on

$$w\vec{v}_i$$

- If this condition persists for a number of iterations,

$$w\vec{v}_i \rightarrow 0$$

What is the solution?

- Prevent the condition from occurring
- How?
  - Let the global best particle perform a local search as is done in the GCPSO of Van den Bergh and Engelbrecht.
  - Use mutation to break the condition.

Fully Informed Particle Swarm (FIPS)

Previous velocity equation shows that that a particle tends to converge towards a point determined by $\vec{p}$, which is a weighted average of its previous best $\vec{p}_i$ and the neighbourhood's best $\vec{p}_g$. $\vec{p}$ can be further generalized to any number of terms:

$$p = \frac{\sum_{k \in \mathrm{N}} \vec{r}[0, \frac{c_{\max}}{|\mathrm{N}|}] \otimes \vec{p}_k}{\sum_{k \in \mathrm{N}} \vec{\varphi}_k}$$

N denotes the neighbourhood, and $\vec{p}_k$ the best previous position found by the *k*-th particle in **N**. If the size of **N** equals 2, $\vec{p}_1 = \vec{p}_i$ and $\vec{p}_2 = \vec{p}_g$ then the above is a generalization of the canonical PSO.

A significant implication of the above generalization is that it allows us to think more freely employing terms of influence other than just $\vec{p}_i$ and $\vec{p}_g$.

ion

Essential particle swarm(1)

Kennedy (2006) describes PSO in the following form:

New Position = Current Position +
Persistence +
Social Influence.

If we substitute $\vec{v}_{i,t} = \vec{x}_{i,t} - \vec{x}_{i,t-1}$ in FIPS, we have:

Persistence

Social influence

$$\vec{x}_{i,t+1} \leftarrow \vec{x}_{i,t} + \chi((\vec{x}_{i,t} - \vec{x}_{i,t-1}) + \sum_{k \in \mathrm{N}} \vec{r}[0, \frac{c_{\max}}{|\mathrm{N}|}] \otimes (\vec{p}_k - \vec{x}_{i,t}))$$

Persistence indicates the tendency of a particle to persist in moving in the same direction it was moving previously.

Essential particle swarm(2)

The social influence term can be further expanded:

New Position = Current Position +
Persistence +
Social Central Tendency +
Social Dispersion

Social central tendency can be estimated, for example by taking the mean of previous bests relative to the particle's current position (still open-ended questions)

Social dispersion may be estimated by taking the distance of a particle's previous best to any neighbor's previous best; or by averaging pair-wise distances between the particle and some neighbors.

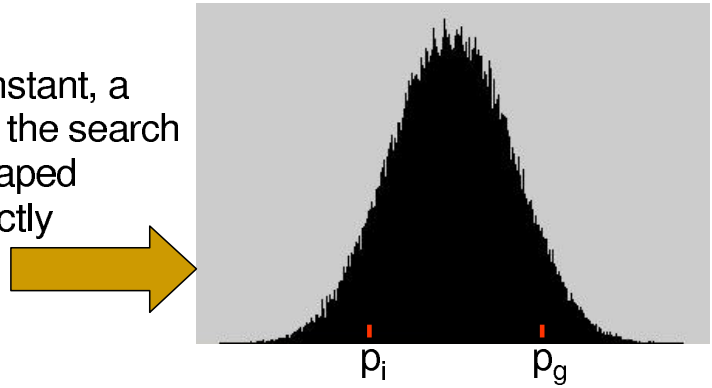
Some distributions such as Gaussian, double-exponential and Cauchy were used by Kennedy (2006).

Bare Bones PSO

What if we drop the velocity term? Is it necessary?

Kennedy (2003) carried out some experiments using a PSO variant, which drops the velocity term from the PSO equation.

If p_i and p_g were kept constant, a canonical PSO samples the search space following a bell shaped distribution centered exactly between the p_i and p_g.



This bare bones PSO produces normally distributed random numbers around the mean $(p_{id} + p_{gd})/2$ (for each dimension *d*), with the standard deviation of the Gaussian distribution being $| p_{id} - p_{gd} |$.

Binary PSO (1)

- PSO was originally developed to optimize continuous-valued parameters.
- Kennedy and Eberhart proposed a binary PSO to optimize binary-valued parameters.
- Here position vectors are binary vectors, and the velocity vectors are still floating-point vectors.
- However, velocities are used to determine the probability that an element of the position vector is bit 0 or bit 1.

ion

Binary PSO (2)

- Position update changes to:

$$x_{ij}(t+1) = \begin{cases} 1 & if\ U(0,1) < sig(v_{ij}(t+1)) \\ 0 & otherwise \end{cases}$$

where

$$sig(v) = \frac{1}{1+e^{-v}}$$

Angle Modulated PSO

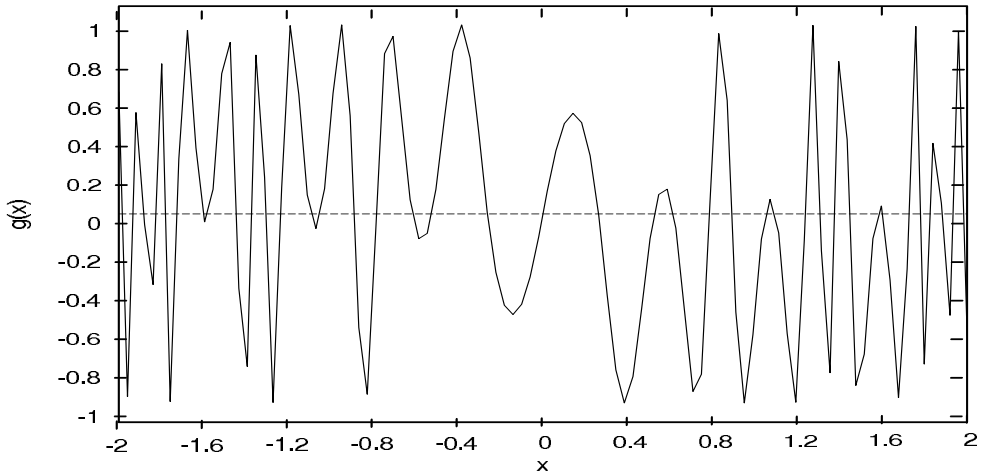
- Developed by Pampara, Engelbrecht and Franken to optimize binary-valued parameters by evolving a bitstring generating function,

$$g(x) = \sin(2\pi(x-a)\times b\times\cos(2\pi(x-a)\times c)) + d$$

- The task is then to find values for **a,b,c** and **d**, where these values are floating-points.
- A binary-valued problem is therefore solved by using the standard PSO to values for the 4 floating point variables, and then to use the generating function above to produce a bitstring. This bitstring is then evaluated using the fitness function

Producing the bitstring

- Sample the generating function at regular intervals. If the output is positive, record bit 1; otherwise record bit 0.

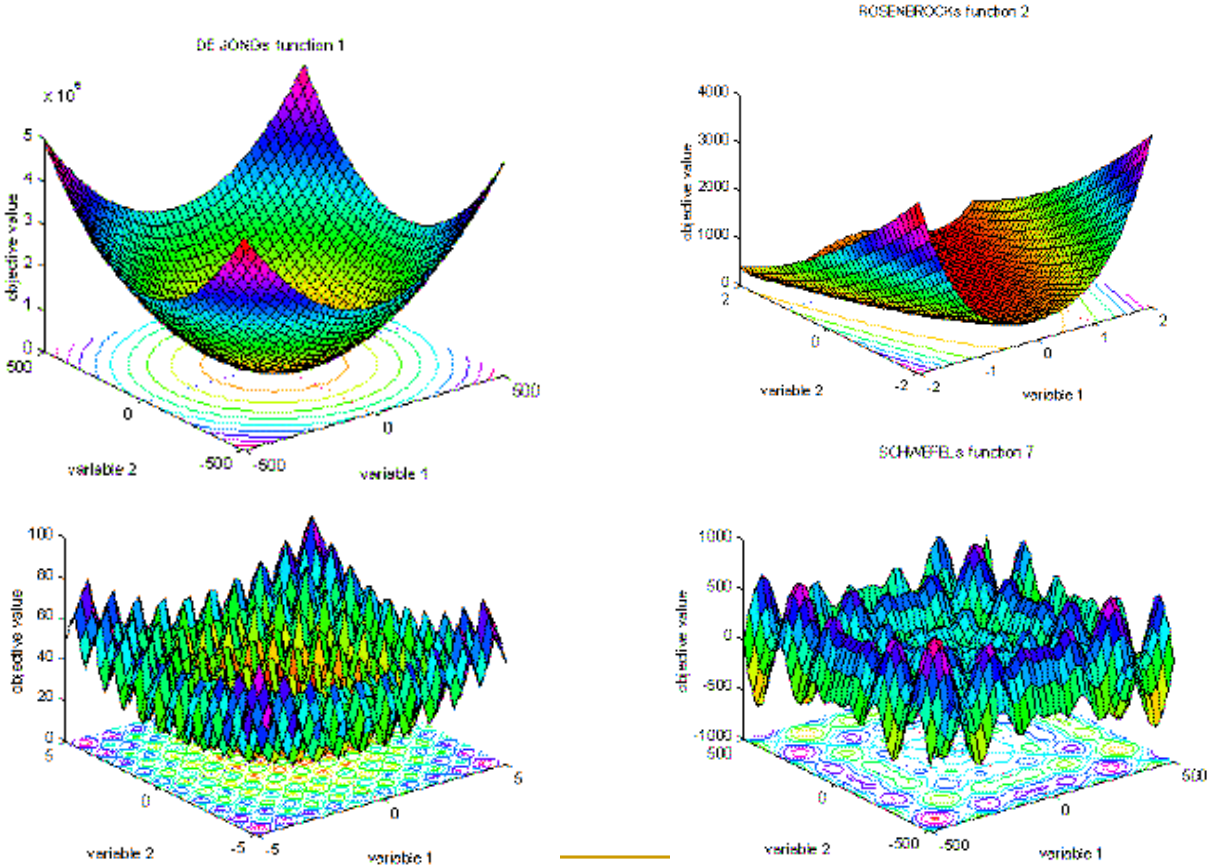


Some PSO variants

- Tribes (Clerc, 2006) – aims to adapt population size, so that it does not have to be set by the users; Tribes have also been used for discrete, or mixed (discrete/continuous) problems.
- ARPSO (Riget and Vesterstorm, 2002) – uses a diversity measure to alternate between 2 phases;
- Dissipative PSO (Xie, et al., 2002) – increasing randomness;
- PSO with self-organized criticality (Lovbjerg and Krink, 2002) – aims to improve diversity;
- Self-organizing HierachicI PSO (Ratnaweera, et al. 2004);
- FDR-PSO (Veeramachaneni, et al., 2003) – using nearest neighbour interactions;
- PSO with mutation (Higashi and Iba, 2003; Stacey, et al., 2004)
- Cooperative PSO (van den Bergh and Engelbrecht, 2005) – a cooperative approach
- DEPSO (Zhang and Xie, 2003) – aims to combine DE with PSO;
- CLPSO (Liang, et al., 2006) – incorporate learning from more previous best particles.

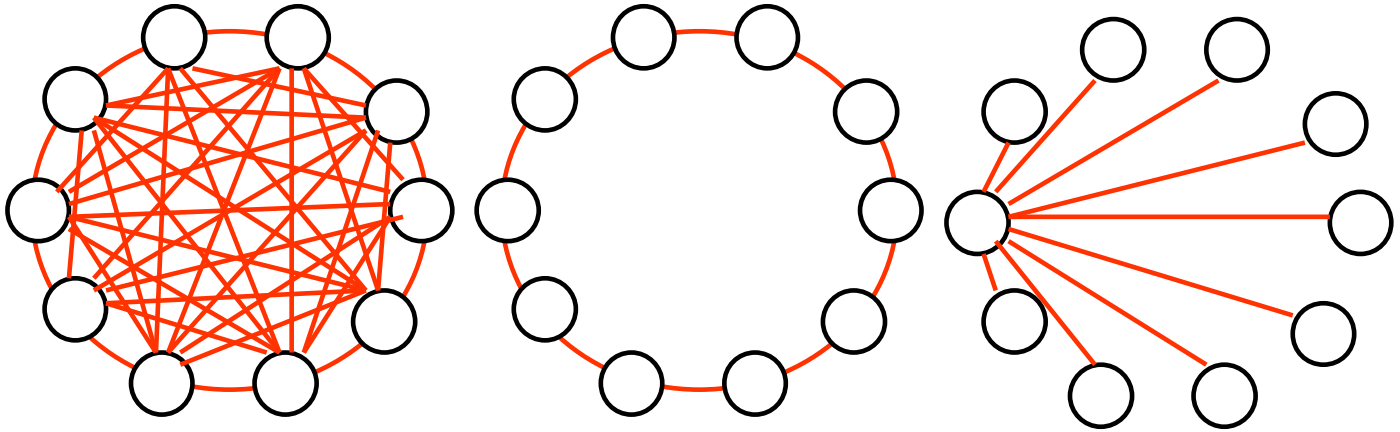
ion

Test functions



Note: Demos on some test functions using a PSO.

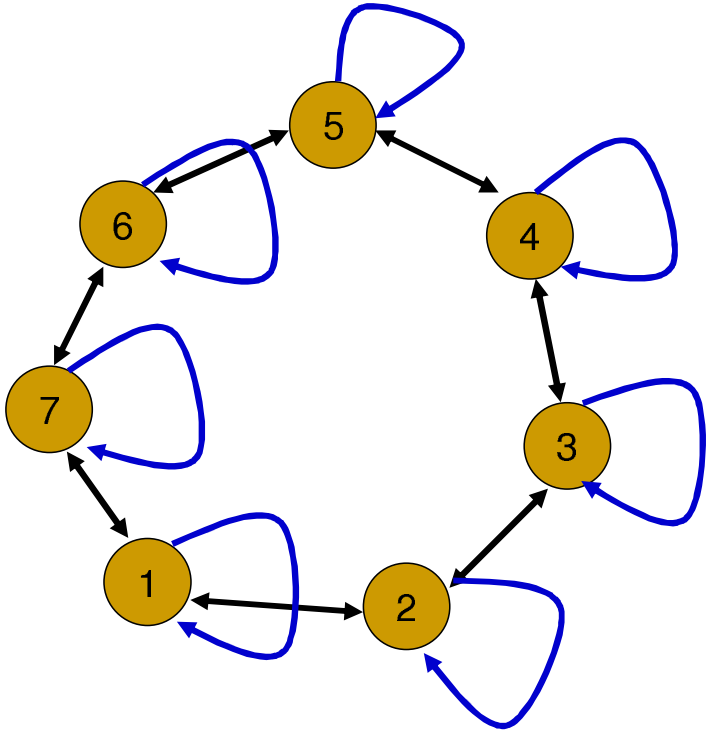
Communication topologies (1)



Two most common models:

- **gbest**: each particle is influenced by the best found from the entire swarm.
- **lbest**: each particle is influenced only by particles in local neighbourhood.

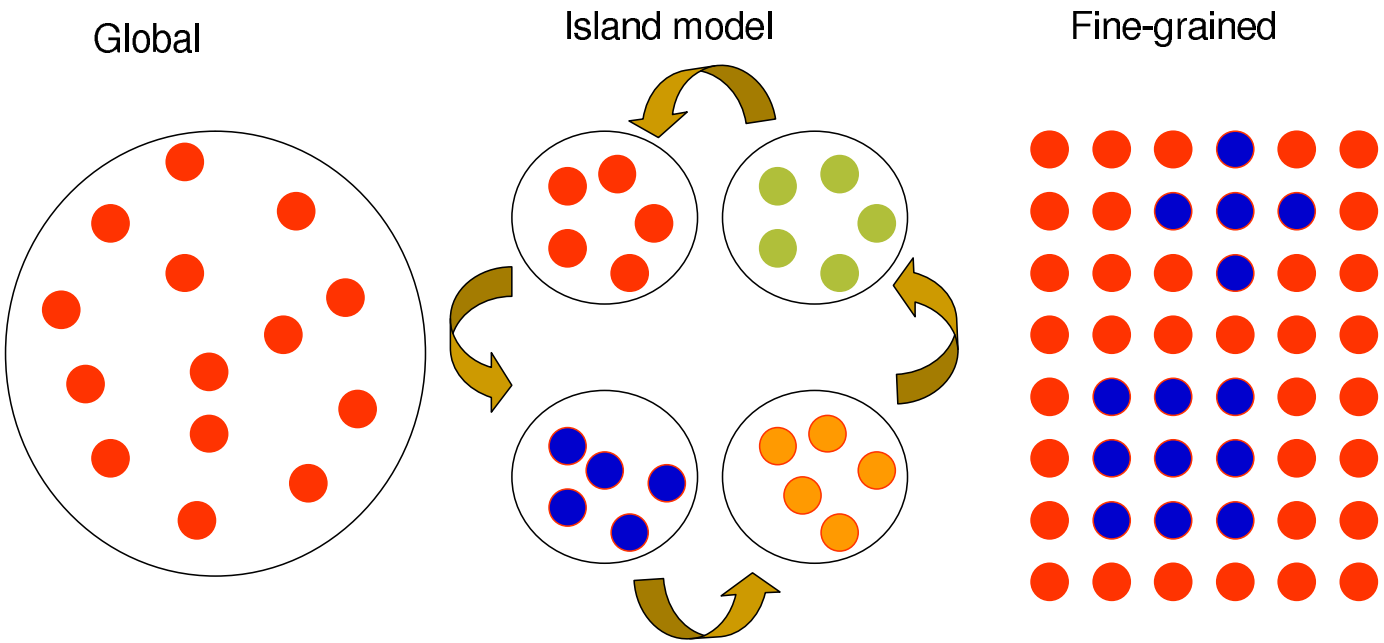
Communication topologies (2)



Graph of influence of a swarm of 7 particles. For each arc, the particle origin influence (informs) the end particle (Clerc, 2006)

This graph of influence can be also expanded to include previous best positions (i.e., memories).

Communication topologies (3)



ion

Communication topologies (4)

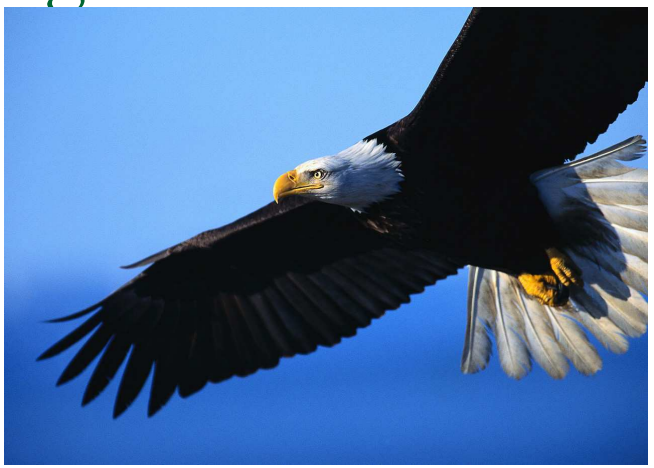
Which one to use?

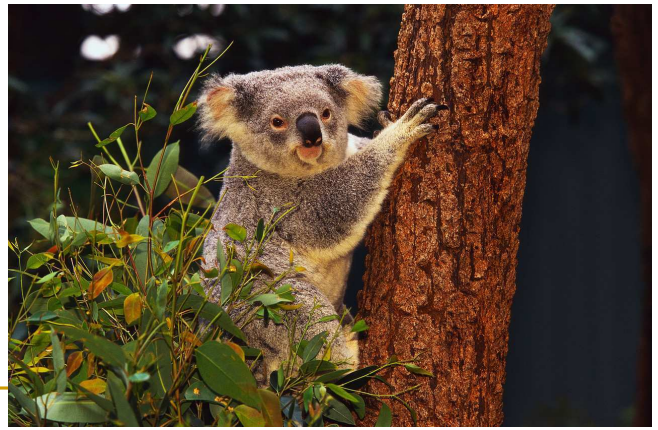
Balance between exploration and exploitation…

gbest model propagate information the fastest in the population; while the **lbest** model using a ring structure the slowest. For complex multimodal functions, propagating information the fastest might not be desirable. However, if this is too slow, then it might incur higher computational cost.

Mendes and Kennedy (2002) found that von Neumann topology (north, south, east and west, of each particle placed on a 2 dimensional lattice) seems to be an overall winner among many different communication topologies.

Speciation and niching



Speciation and niching

Biological species concept: *a species is a group of actually or potentially interbreeding individuals who are reproductively isolated from other such groups*.

The definition of a species is still debatable.

Most researchers believe either the **morphological** species concept (ie., members of a species look alike and can be distinguished from other species by their appearance), or the **biological** species concept (a species is a group of actually or potentially interbreeding individuals who are reproductively isolated from other such groups). Both definitions have their weaknesses.

Speciation and niching

- Kennedy (2000) proposed a *k*-means clustering technique;
- Parsopoulos and Vrahitis (2001) used a stretching function;
- Brits et al. (2002) proposed a NichePSO;
- Petrowski (1996) introduced a clearing procedure, and subsequently, Li, et al. (2002) introduced a species conserving genetic algorithm (SCGA) for multimodal optimization.
- Li (2004) developed SPSO based on Petrowski's clearing procedure.
- Many other niching methods developed for Evolutionary Algorithms, such as Crowding method, fitness-sharing, etc.

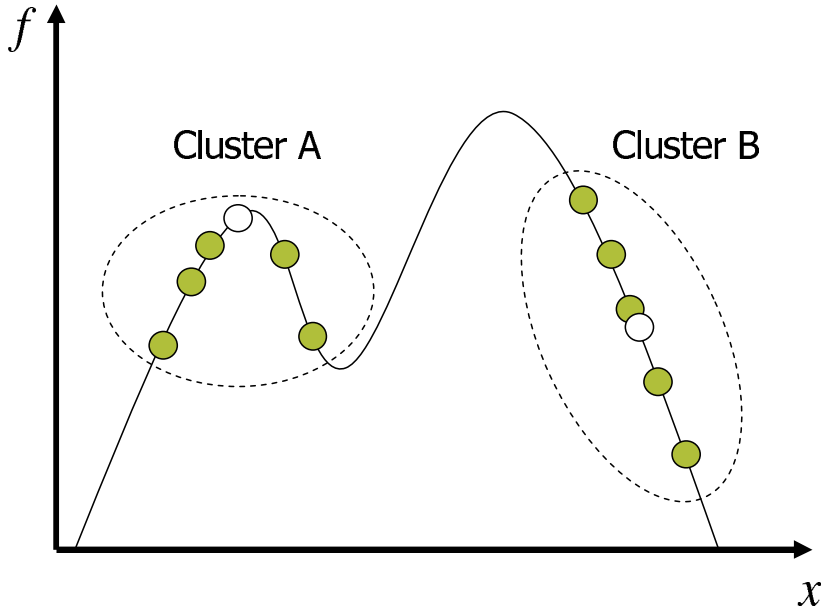
The notion of species:

- A population is classified into groups according to their similarity measured by Euclidean distance.

- The definition of a species also depends on another parameter r_s ,which denotes the radius measured in Euclidean distance from the center of the a species to its boundary.

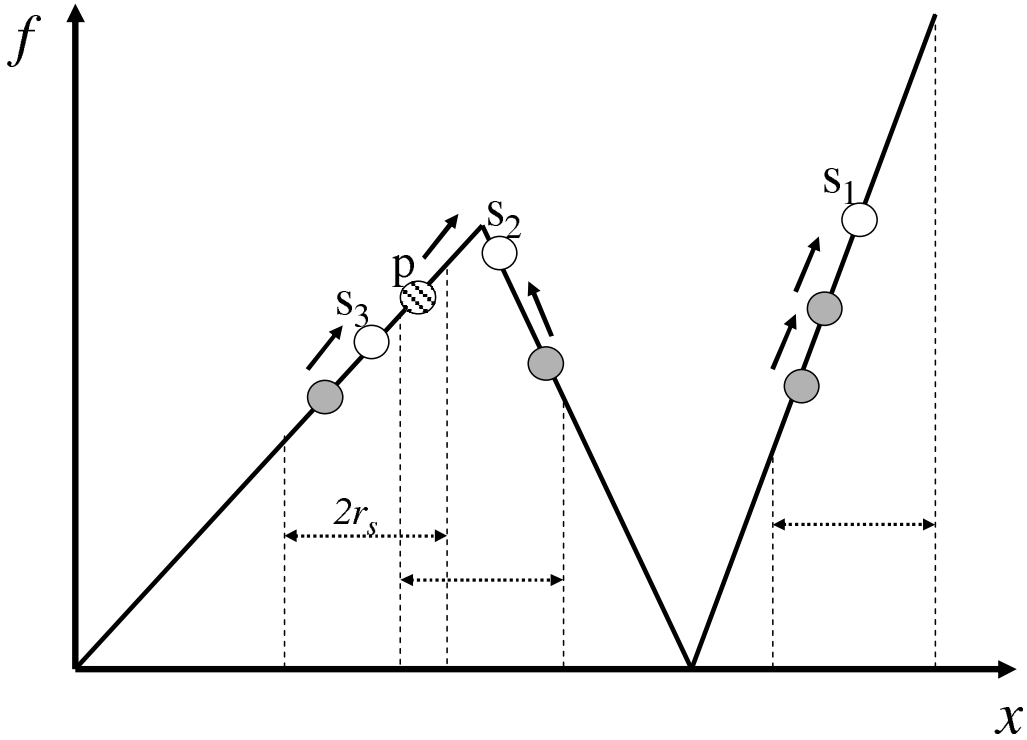
ion

Clustering-based PSO



Cluster A's center performs better than all members of cluster A, whereas cluster B's center performs better than some and worse than others.

Speciation-based PSO



An example of how to determine the species seeds from the population at each iteration. s_1, s_2, and s_3 are chosen as the species seeds. Note that p follows s_2.

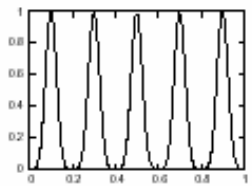
Speciation-based PSO

Step 1: Generate an initial population with randomly generated particles;
Step 2: Evaluate all particle individuals in the population;
Step 3: Sort all particles in descending order of their fitness values (i.e., from the best-fit to least-fit ones);
Step 4: Determine the species seeds for the current population;
Step 5: Assign each species seed identified as the $\vec{p}_g$ to all individuals identified in the same species;
Step 6: Adjusting particle positions according to the PSO velocity and position update equation (1) and (2);
Step 7: Go back to step 2), unless termination condition is met.

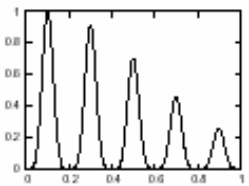
Multimodal problems

ion

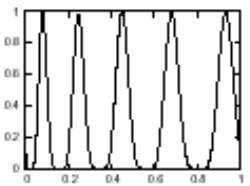
Multimodal functions



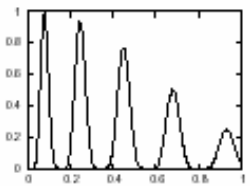
(a) F1



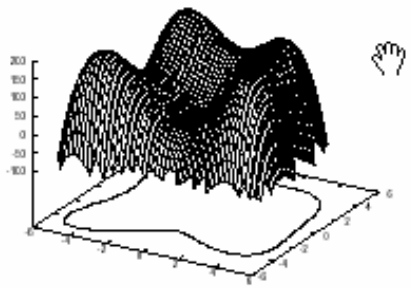
(b) F2



(c) F3



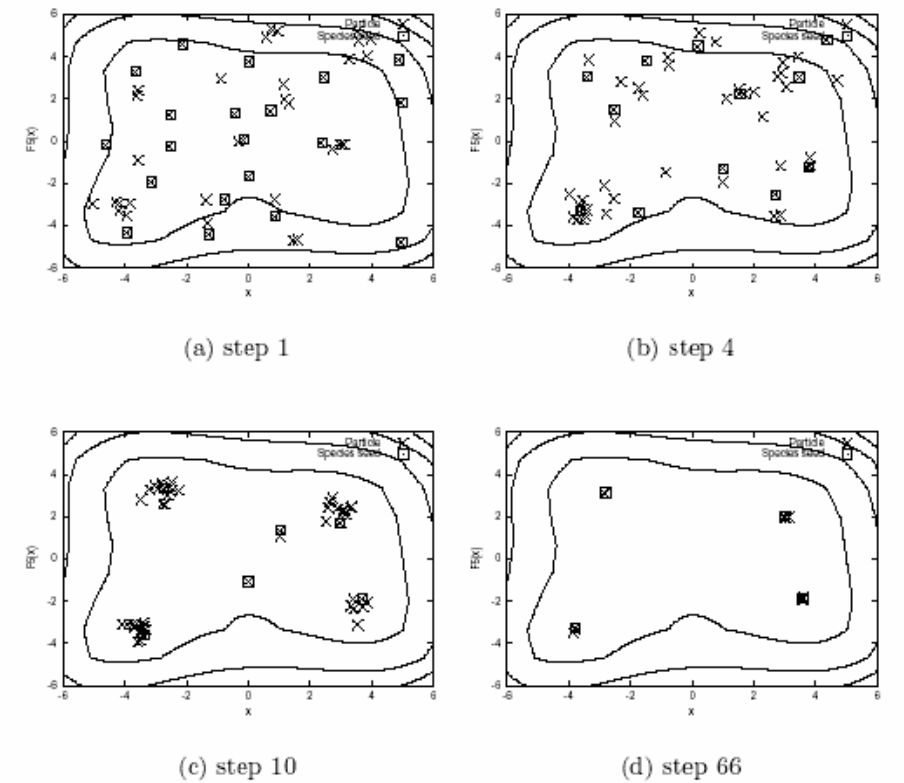
(d) F4



(e) F5:Himmelblau's function

(f) F6:Rastrigin function

Simulation runs



(a) step 1 (b) step 4

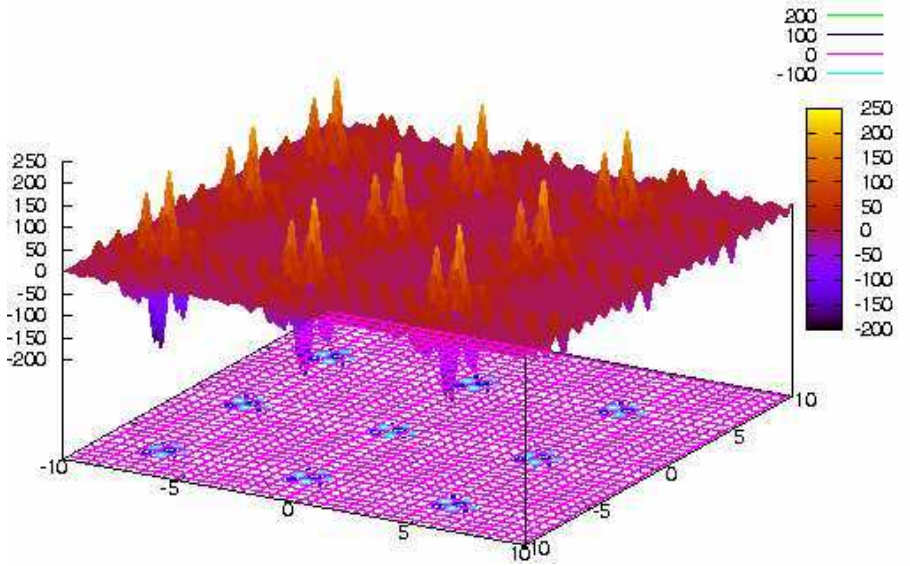
(c) step 10 (d) step 66

Fig. 6. A simulation run of SPSO on F5 - step 1, 4, 10 and 66.

Refer to Li (2004) for details.

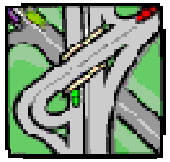
Niching parameters

Difficulty in choosing the niching parameters such as the species radius *r* . For example, for Shubert 2D, there is no value of *r* that can distinguish the global optima without individuals becoming overly trapped in local optima.



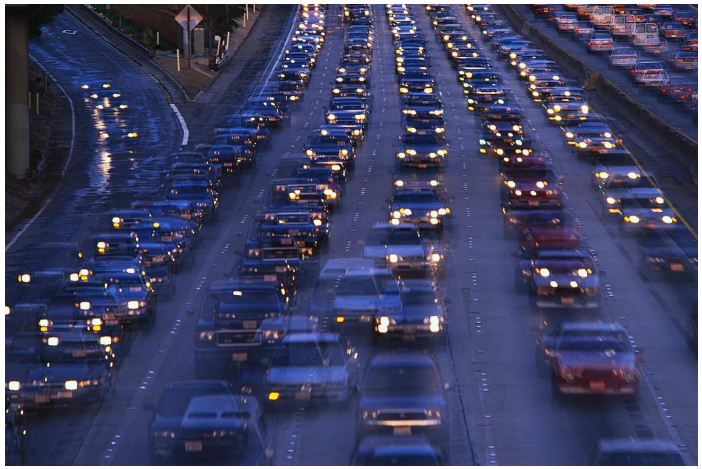
Some recent works in handling this problem (Bird & Li, 2006a; Bird & Li, 2006b).

Optimization in a dynamic environment



Many real-world optimization problems are dynamic and require optimization algorithms capable of adapting to the changing optima over time.

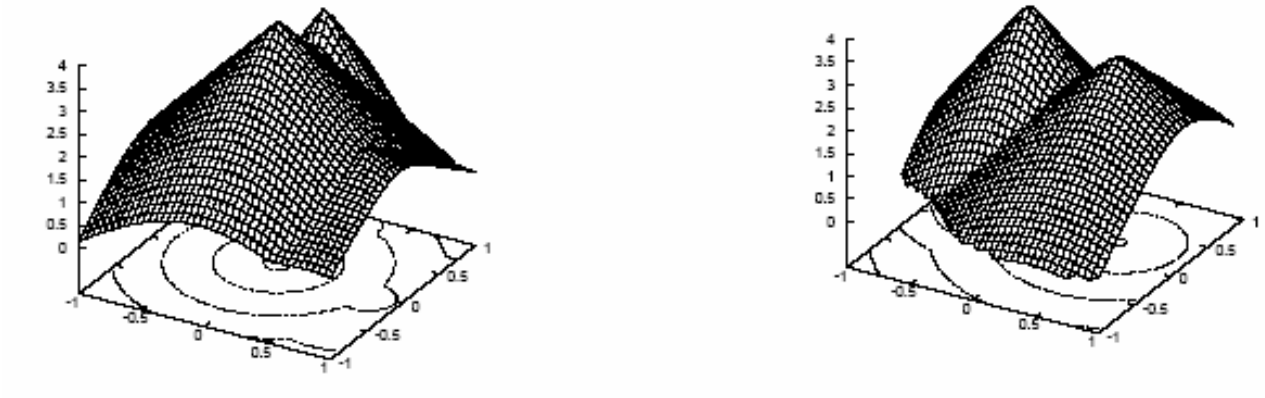
E.g., Traffic conditions in a city change dynamically and continuously. What might be regarded as an optimal route at one time might not be optimal in the next minute.



In contrast to optimization towards a static optimum, in a dynamic environment the goal is to track as closely as possible the dynamically changing optima.

ion

Optimization in a dynamic environment



Three peak multimodal environment, before (above left) and after (above right) movement of optima. Note that the small peak to the right of the figure becomes hidden and that the highest point switches optimum (Parrott and Li, 2006).

Why PSO?

- With a population of candidate solutions, a PSO algorithm can maintain useful information about characteristics of the environment.
- PSO, as characterized by its fast convergence behaviour, has an in-built ability to adapt to a changing environment.
- Some early works on PSO have shown that PSO is effective for locating and tracking optima in both static and dynamic environments.

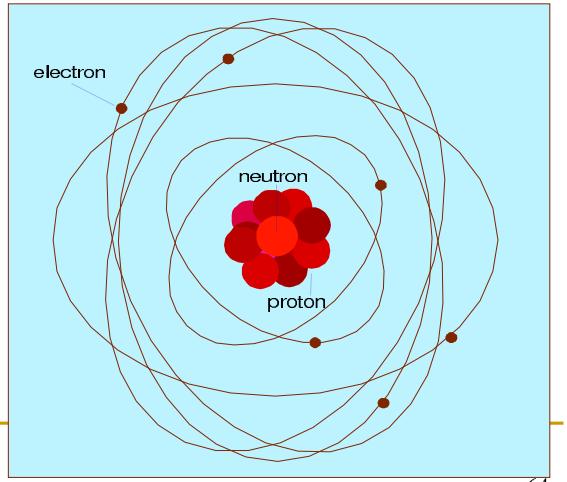
Following questions must be addressed:

1) How to detect a change that has actually occurred?
2) What response strategies are appropriate to use once a change is detected?
3) How to handle the issue of **'outdated memory'** issue as particles' personal bests become invalid once environment has changed?
4) How to handle the trade-off issue between convergence (in order to locate optima) and diversity (in order to relocate changed optima)?

Related work

- Tracking the changing optimum of a unimodal parabolic function (Eberhart and Shi, 2001).
- Carlisle and Dozier (2002) used a randomly chosen sentry particle to detect if a change has occurred.
- Hu and Eberhart (2002) proposed to re-evaluate the global best particle and a second best particle.
- Carlisle and Dozier (2002) proposed to re-evaluate all personal bests of all particles when a change has been detected.
- Hu and Eberhart (2002) studied the effects of re-randomizing various proportions of the swarm.

- Blackwell and Bentley (2002) introduced charged swarms.
- Blackwell and Branke (2004, 2006) proposed an interacting multi-swarm PSO (using quantum particles) as a further improvement to the charged swarms.



Set the scope

Many complex scenarios are possible:

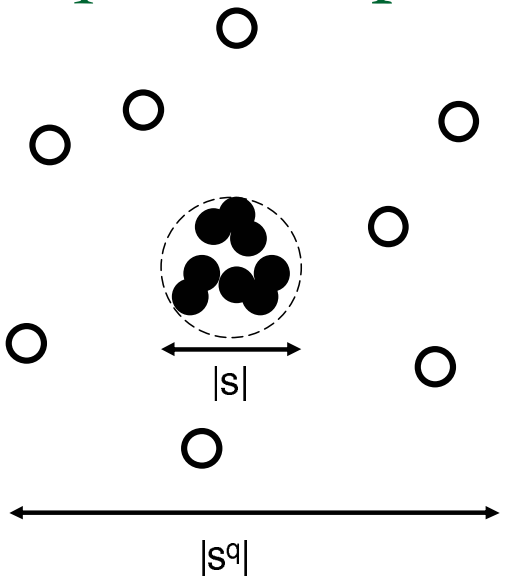
- Small and continuous changes;
- Large, random and infrequent changes;
- Large and frequent changes.

Assumption:
Here we assume that changes are only slight in a dynamic environment. It would be beneficial to use knowledge about the old environment to help search in the new environment.

- Speciation-based PSO is able to identify peaks and converge onto these peaks in parallel and adaptively.
- It can be further enhanced by other techniques (eg., quantum swarms) to better track changing optima.

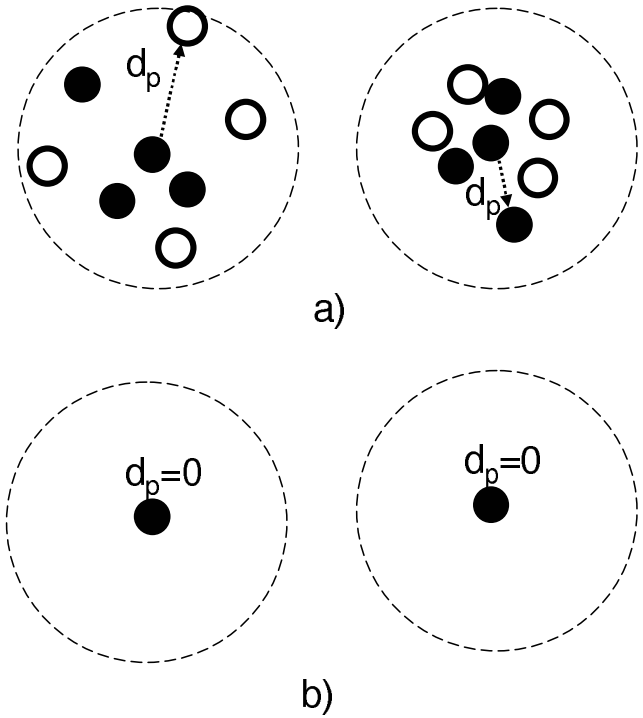
ion

SPSO with quantum particles



In this quantum swarm model, a swarm is made up of neutral (ie., conventional and quantum particles. Quantum particles are positioned as a *cloud* centered around the $\vec{p}_g$, providing a constant level of particle diversity within a species (Li *et al.*, 2006).

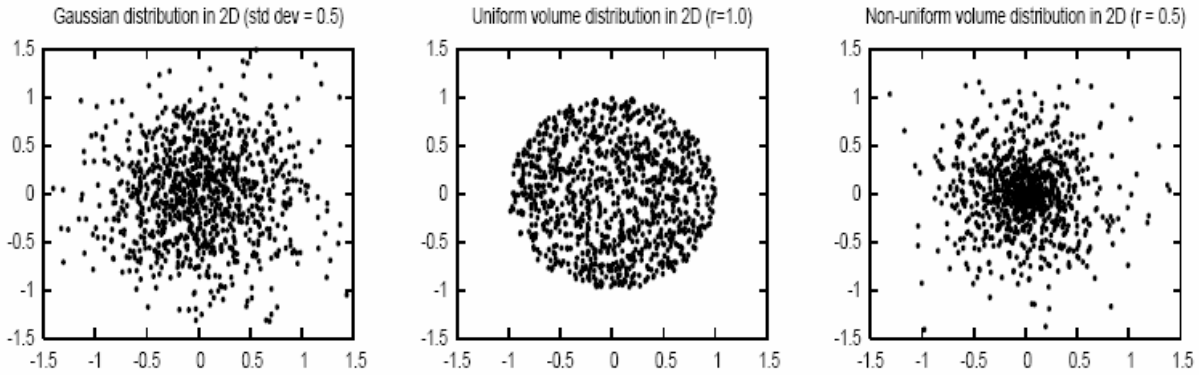
SPSO with quantum particles



To see if a species has converged, we check if the particle diversity, d_p, of a species is smaller than a threshold.

To regain diversity, all particles except the species seed in the converged species are replaced by the same quantity of particles, centered around the species seed, with 50% as neutral particles and the remaining 50% as quantum particles.

Local sampling



Different sampling distributions can be employed to produce the quantum `cloud'. Local sampling around the center of a species (or other points) can be carried out immediately after a change is detected in the environment.

Test functions for dynamic optimization

Jürgen Branke's **Moving peak test functions** - The moving peak benchmark (MPB) is widely used in the EC community. A few recent PSO works also adopted it (Clerc, 2006; Blackwell and Branke, 2004; Li et al., 2006). For more information, refer to:

http://www.aifb.uni-karlsruhe.de/~jbr/MovPeaks/

Morrison and De Jong's **DF1** function generator – one of the early dynamic test function generator proposed (Morrison, 2005). A few authors have used it (Parrott and Li, 2006).

A few other dynamic test functions have also been proposed in recent years.

A demonstration run of SPSO tracking the global peak in a 10 peaks dynamic environment (Moving peaks Scienario2). Refer to (Li, et al. 2006) for details.

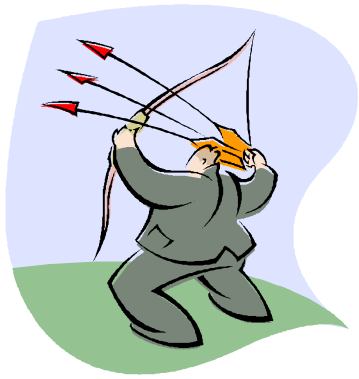
ion

Multiobjective optimization

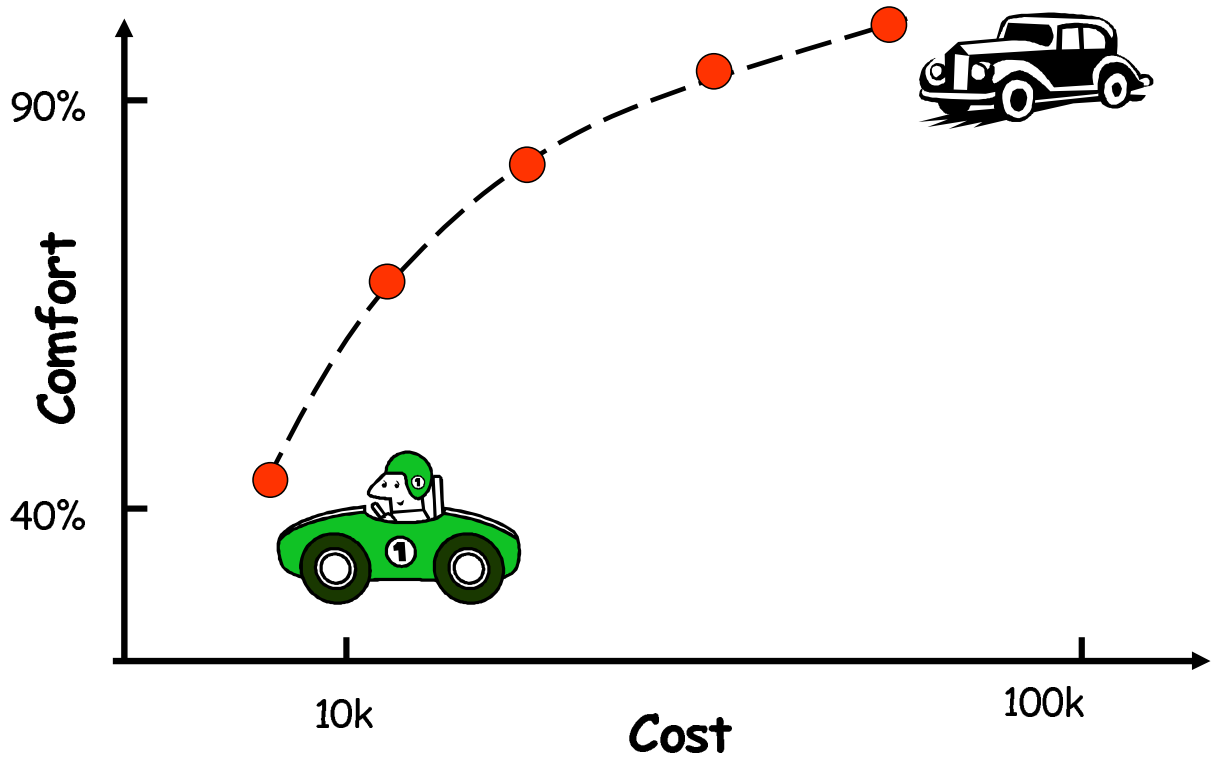
"The great decisions of human life have as a rule far more to do with the instincts and other mysterious unconscious factors than with conscious will and well-meaning reasonableness. The shoe that fits one person pinches another; there is no recipe for living that suits all cases. Each of us carries his own life-form - an indeterminable form which cannot be superseded by any other."

Carl Gustav Jung, Modern Man in Search of a Soul, 1933, p. 69

Many real-world problems involve multiple conflicting objectives, which need to be optimized simultaneously. The task is to find the best possible solutions which still satisfy all objectives and constraints. This type of problems is known as multiobjective optimization problems.



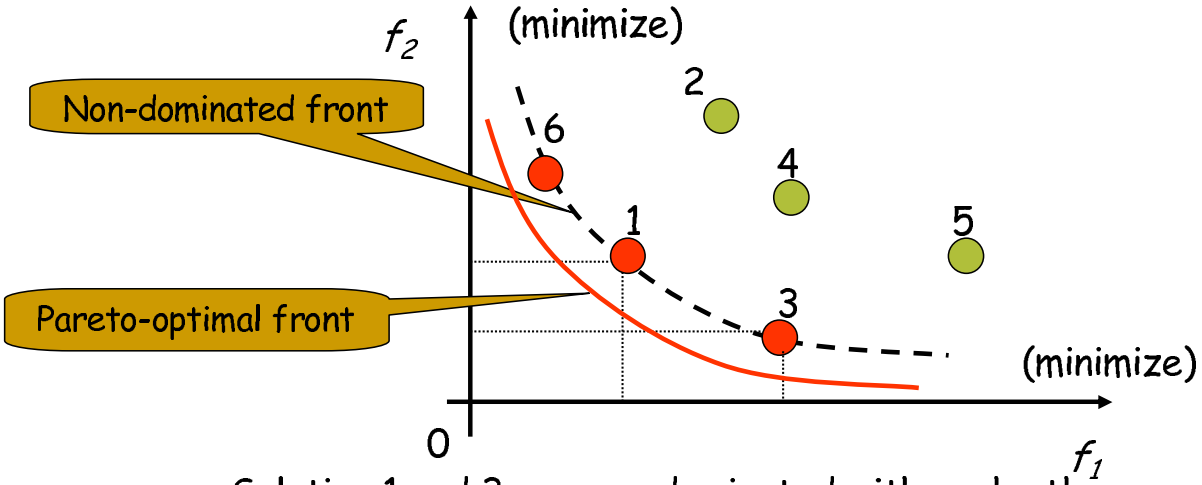
Multiobjective optimization



Concept of domination

A solution vector **x** is said to dominate the other solution vector **y** if the following 2 conditions are true:

- The solution **x** is no worse than **y** in all objectives;
- The solution **x** is strictly better than **y** in at least one objective.



Solution 1 and 3 are non-dominated with each other.

Solution 6 dominates 2, but not 4 or 5.

PSO for Multiobjective Optimization

Two major goals in multiobjective optimization:

- To obtain a set of non-dominated solutions as closely as possible to the true Pareto front;
- To main well-distributed solutions along the Pareto front.

Several issues have to be taken into consideration:

1) How to choose p_g (i.e., a leader) for each particle? The PSO needs to find diverse solutions along the Pareto front, not just a single point. This requires that particles are allocated with different leaders.

2) How to identify non-dominated particles with respect to all particles' current positions and personal best positions? And how to retain these solutions during the search process? One strategy is to combine all particles' personal bests and current positions, and then extract the non-dominated solutions from the combined population.

3) How to maintain particle diversity so that a set of well-distributed solutions can be found along the Pareto front? Some classic niching methods (e.g., crowding or sharing) can be adopted for this purpose.

ion

PSO algorithms for MO

Some earlier PSO models using different techniques:

MOPSO (Coello et al., 2002) – dominance comparison for each particle with its personal best; diversity is maintained using a grid-based approach.

Aggregation approaches (Parsopoulos and Vrahatis, 2002) – 3 different aggregation functions used.

Fieldsend and Sigh (2002) – use "dominated tree" to store non-dominated solutions.

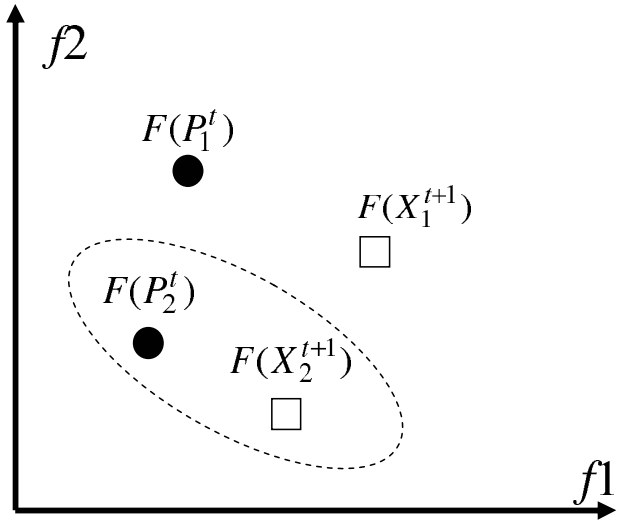
Dynamic neighbourhood (Hu and Eberhart, 2002, 2003) – One objective optimized at a time, later enhanced with an "extended memory".

Sigma method (Mostaghim & Teich, 2003) – a method to better choose local guides.

Non-dominated Sorting PSO (Li, 2003) – dominance comparison for all particles including personal bests; non-dominated sorting is used, similar to NSGA II.

Recently a survey by Sierra and Coello shows that there are currently 25 different PSO algorithms for solving MO problems (Sierra and Coello, 2006).

Better dominance comparison for PSO



Dominance relationships among 4 particles, including the personal bests of two particles, and their potential offspring, assuming minimization of *f1* and *f2*.

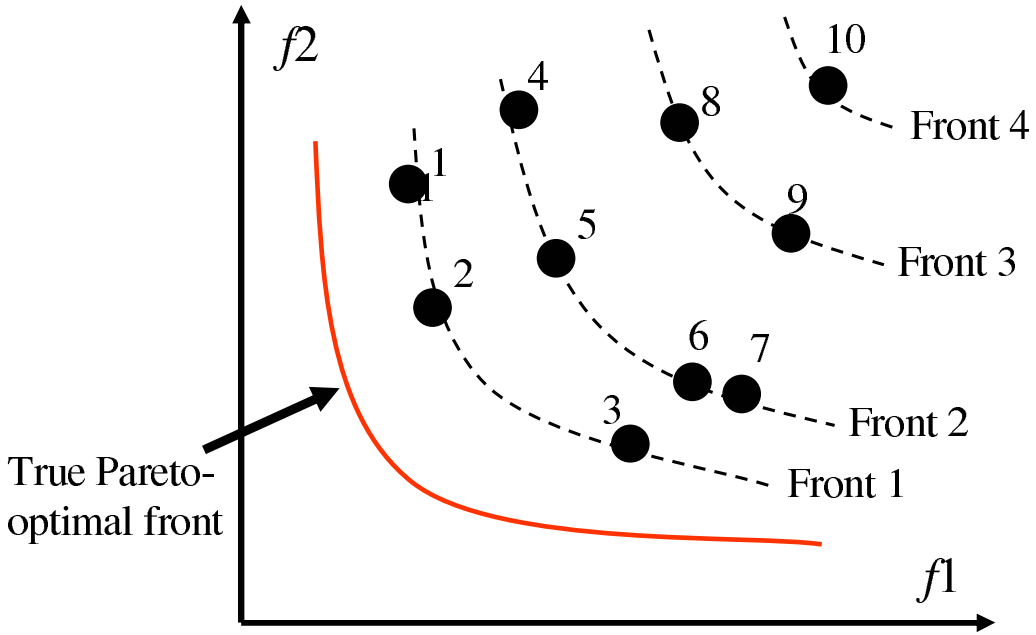
Extracting non-dominated solutions from combined current positions and their personal bests are more effective than just a single comparison between a particle and its personal best alone.

NSPSO Algorithm

The basic idea:

- Instead of comparing solely on a particle's personal best with its potential offspring, the entire population of **N** particles' personal bests and **N** of these particles' offspring are first combined to form a temporary population of 2**N** particles. After this, domination comparisons among all the 2**N** individuals in this temporary population are carried out.
- Sort the entire population in different non-domination levels (as in NSGA II). This type of sorting can then be used to introduce the selection bias to the individuals in the populations, in favour of individuals closer to the true Pareto front.
- At each iteration step, we choose only **N** individuals out of the 2**N** to the next iteration step, based on the non-domination levels, and two niching methods.

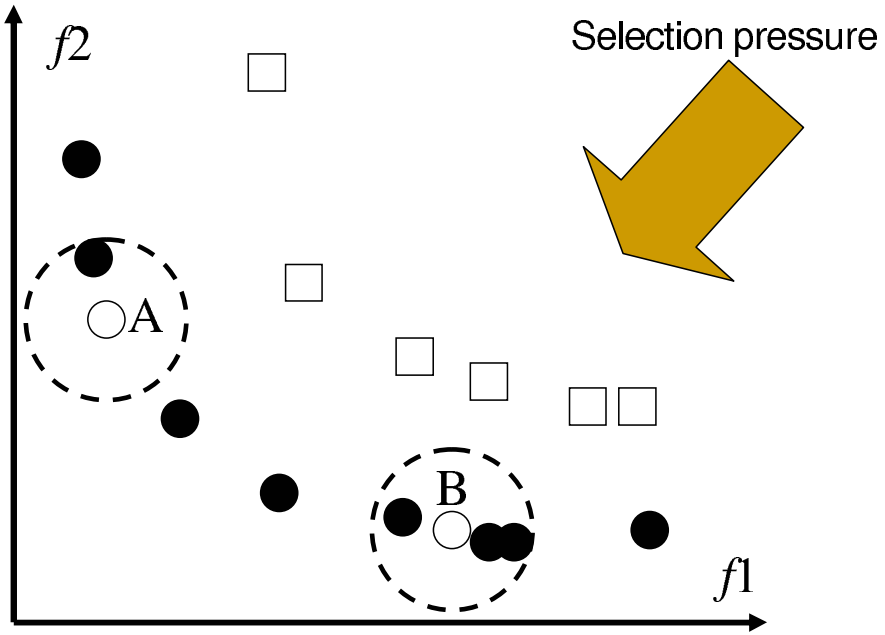
Non-dominated Sorting PSO



Selection pressure towards the true Pareto-optimal front.

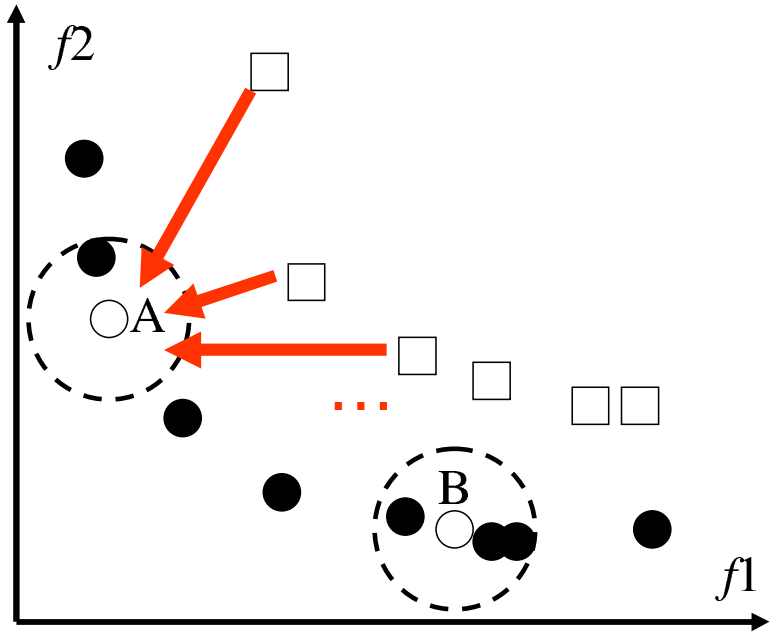
ion

Niching techniques



A will be preferred over B, since A has a smaller niche count than B.

Selecting better guides



Particles in the "less-crowded' area of the non-dominated front is more likely to be chosen as $\vec{p}_g$ (ie., leader) for particles in the population, eg., **A** is more likely than **B**.

Performance metrics

- Diversity of the solutions along the Pareto front in the final population:

$$\Delta = \frac{\sum_{m=1}^{M} d_m^e + \sum_{i=1}^{|Q|} | d_i - \bar{d} |}{\sum_{m=1}^{M} d_m^e + | Q | \bar{d}},$$

- Closeness to the true Pareto-optimal front:

$$GD = \frac{(\sum_{i=1}^{|Q|} d_i^p)^{1/p}}{| Q |}.$$

Test functions (ZDT series)

Two objectives are to be minimized:

$$\begin{aligned} &Minimize \quad f_1(x) \\ &Minimize \quad f_2(x) = g(x)h(f_1(x), g(x)). \end{aligned}$$

In all problems except ZDT5, the Pareto-optimal front is formed with g(x) = 1

Note that more scalable test functions, such as the DTLZ functions (with more than 2 objectives) were also proposed.

ion

ZDT series

ZDT1

$$f_1(x) = x_1,$$

$$g(x) = 1 + 9\left(\sum_{i=2}^{n} x_i\right)/(n-1)$$

$$h(f_1, g) = 1 - \sqrt{f_1 / g}.$$

ZDT2

$$f_1(x) = x_1,$$

$$g(x) = 1 + 9\left(\sum_{i=2}^{n} x_i\right)/(n-1)$$

$$h(f_1, g) = 1 - (f_1 / g)^2.$$

ZDT3

$$f_1(x) = x_1,$$

$$g(x) = 1 + 9\left(\sum_{i=2}^{n} x_i\right)/(n-1)$$

$$h(f_1, g) = 1 - \sqrt{f_1 / g} - (f_1 / g)\sin(10\pi f_1).$$

ZDT4

$$f_1(x) = x_1,$$

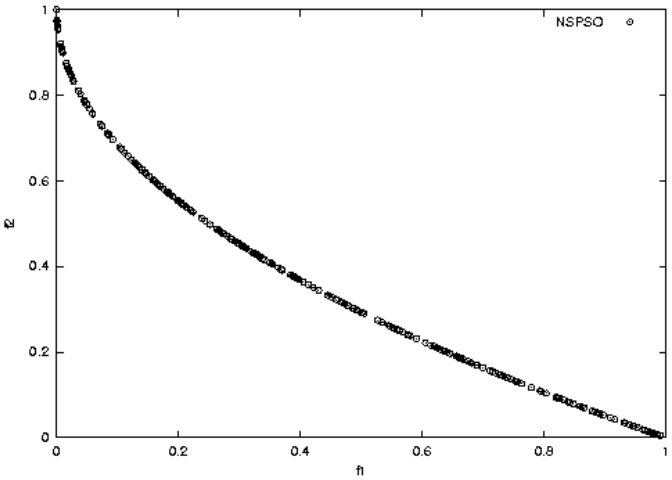
$$g(x) = 1 + 10(n-1) + \left(\sum_{i=2}^{n} (x_i^2 - 10\cos(4\pi x_i)\right),$$

$$h(f_1, g) = 1 - \sqrt{f_1 / g}.$$

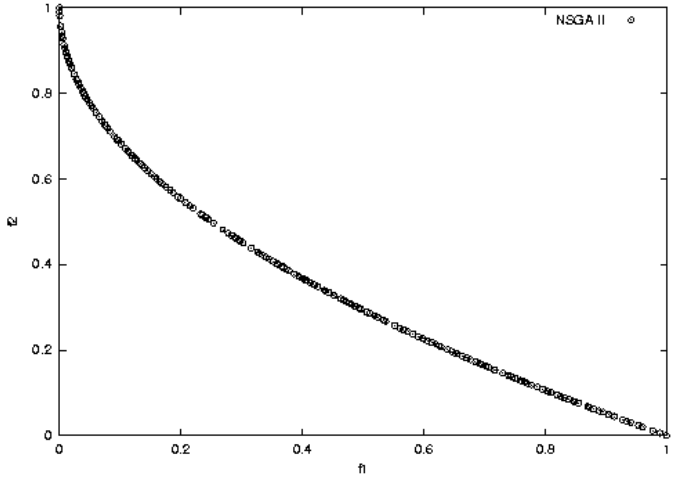
Note: n= 30 (30 variables); x_i in the range [0,1], except for ZDT4, where x_2- x_{30} lie in the range [-5, 5].

Experimental results

NSPSO



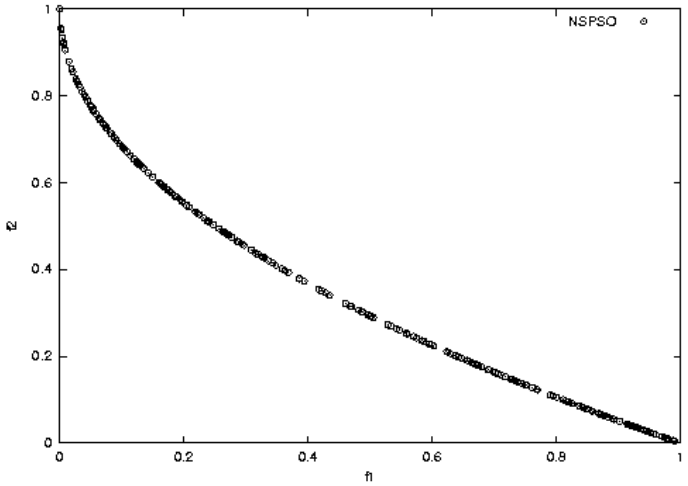
NSGA II



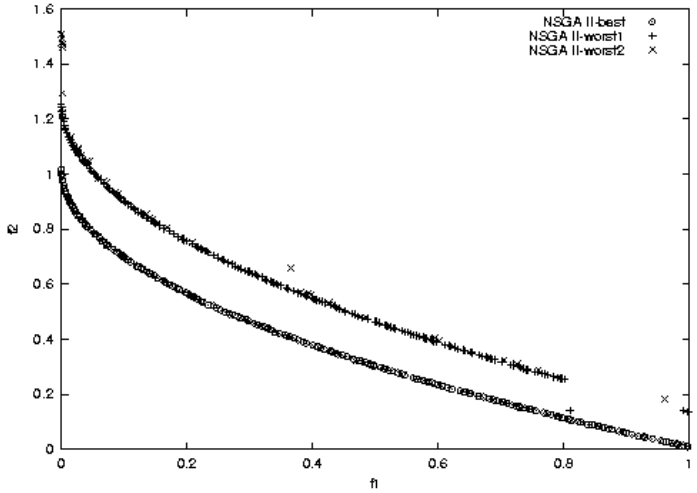
Non-dominated solutions found for ZDT1.

Experimental results

NSPSO

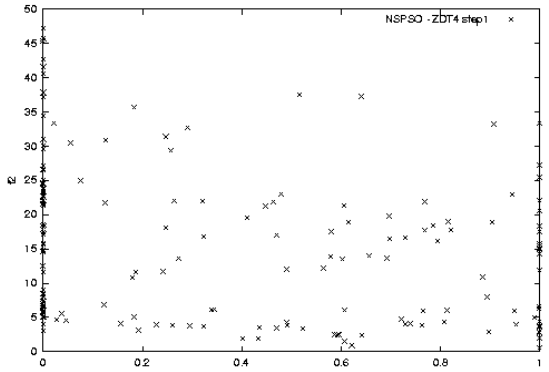


NSGA II

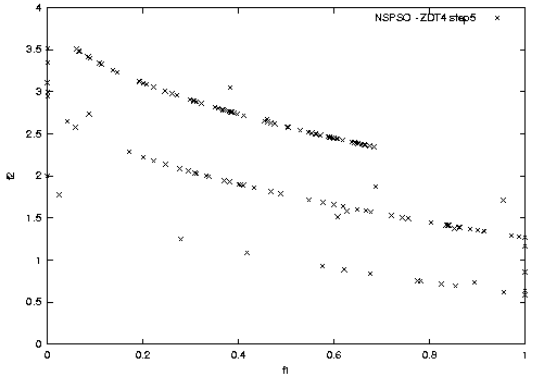


Non-dominated solutions found for ZDT4.

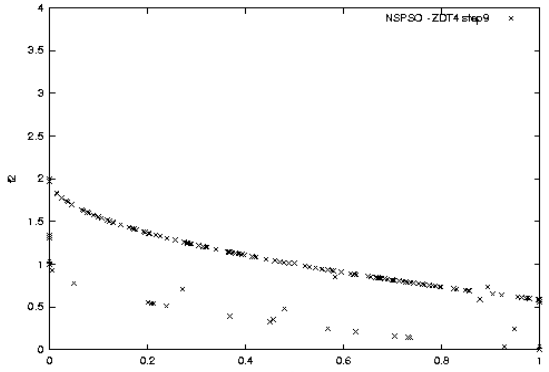
Snapshots of a NSPSO run on ZDT4



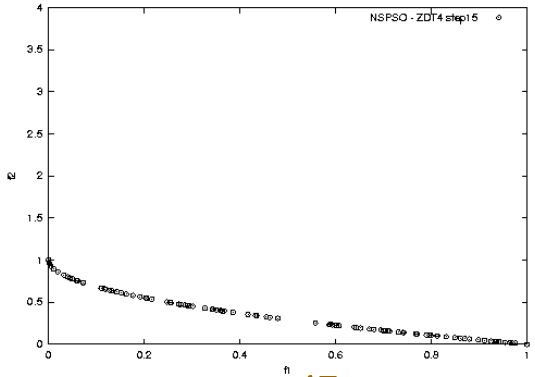
Step 1



Step 3



Step 9



Step 15

ion

Constraint handling

The most common approach for solving constrained problems is the use of a penalty function. The constrained problem is transformed into an unconstrained one, by penalizing the constraints and creating a single objective function.

Non-stationary penalty functions (Parsopoulos and Vrahatis, 2002):

A penalty function is used, and the penalty value is dynamically modified during a run. This method is problem dependent, however, its results are generally superior to those obtained through stationary functions.

Preservation of feasible solutions (Hu and Eberhart, 2002):

During initialization, all particles are repeatedly initialized until they satisfy all constraints; when calculating personal best and global best, only those positions in feasible space are counted.

Based on closeness to the feasible region (Toscano and Coello, 2004):

If both particles compared are infeasible, then the particle that has the lowest value in its total violation of constraints wins.

Constraint handling

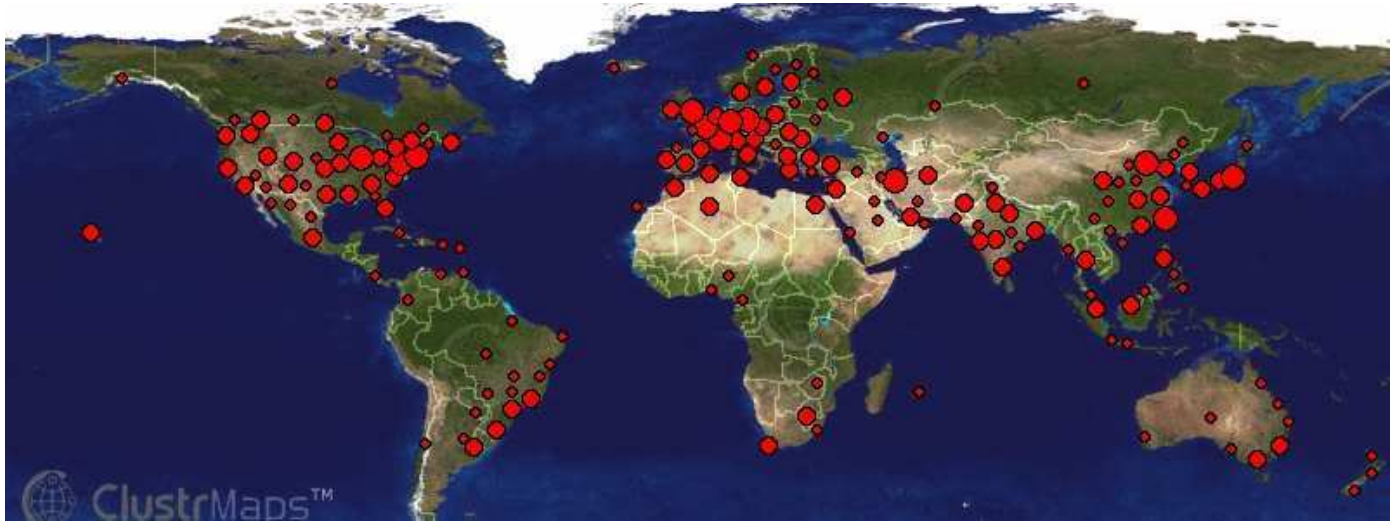
One major disadvantage of using penalty function, in which case all constraints must be combined into a single objective function (this is also called weighted-sum approach), is that a user must specify a weight coefficient for each constraint. However, finding optimal weight coefficients is no easy task.

A preferred approach is to use a multiobjective approach where the concept of "dominance" can be used to identify better solutions which are non-dominated with respect to the current population. The user is no longer required to specify any weight coefficient.

Another useful technique described by Clerc (2006) is "confinement by dichotomy", which makes use of an iterative procedure to find points that are close to the boundaries defined by constraints.

More information

Particle Swarm Central: http://www.particleswarm.info



Visitors' hits since 12 June 2006 (updated daily).

References

Background:

1) Reynolds, C.W.: Flocks, herds and schools: a distributed behavioral model. *Computer Graphics*, 21(4), p.25-34, 1987.
2) Heppner, F. and Grenander, U.: A stochastic nonlinear model for coordinated bird flocks. In S. Krasner, Ed., The Ubiquity of Chaos. AAAS Publications, Washington, DC, 1990.
3) Kennedy, J. and Eberhart, R.: Particle Swarm Optimization. In *Proceedings of the Fourth IEEE International Conference on Neural Networks*, Perth, Australia. IEEE Service Center(1995) 1942-1948.
4) Kennedy, J., Eberhart, R. C., and Shi, Y., *Swarm intelligence*, San Francisco: Morgan Kaufmann Publishers, 2001.
5) Clerc, M.: *Particle Swarm Optimization*, ISTE Ltd, 2006.
6) Engelbrecht , A.P., *Fundamentals of Computational Swarm Intelligence*, Wiley, 2006.

ion

References – continued…

Theoretical work:

1) E. Ozcan and C.K. Mohan. Analysis of a simple particle swarm optimization system. In Intelligent Engineering Systems through Artificial Neural Networks, pages 253–258, 1998.
2) J. Kennedy. The behaviour of particle. In Proc. 7th Annu. Conf. Evol. Program., pages 581–589, San Diego, CA, 1998.
3) I.C. Trelea. The particle swarm optimization algorithm: convergence analysis and parameter selection, 2003.
4) F. van den Bergh. *Analysis of Particle Swarm Optimizers. PhD thesis*, Department of Computer Science, University of Pretoria, Pretoria, South Africa, 2002.
5) M. Clerc and J. Kennedy. The particle swarm - explosion, stability, and convergence in a multidimensional complex space. *IEEE Transactions on Evolutionary Computation*, 6:58–73, 2002.
6) F. van den Bergh and A.P. Engelbrecht. A study of particle swarm optimization particle trajectories. *Information Sciences*, 176:937–971, 2006.
7) V. Kadirkamanathan, K. Selvarajah, and P. Fleming. Stability analysis of the particle dynamics in particle swarm optimizer. *IEEE Transactions on Evolutionary Computation*, 10(3):245–255, June 2006.

References – continued…

New improvements and variants:

1) Y. Shi and R. C. Eberhart, "A modified particle swarm optimizer," in Proc. IEEE Congr. Evol. Comput., 1998, pp. 69–73.
2) Clerc, M. and Kennedy, J.: The particle swarm-explosion, stability and convergence in a multidimensional complex space. *IEEE Trans. Evol. Comput.* Vol.**6**, no.2, pp.58-73, Feb. 2002.
3) Kennedy, J., and Mendes, R. (2002). Population structure and particle swarm performance. Proc. of the 2002 World Congress on Computational Intelligence.
4) T. Krink, J. S. Vesterstroem, and J. Riget, "Particle swarm optimization with spatial particle extension," in Proc. Congr. Evolut. Comput., Honolulu, HI, 2002, pp. 1474–1479.
5) M. Lovbjerg and T. Krink, "Extending particle swarm optimizers with self-organized criticality," in Proc. Congr. Evol. Comput., Honolulu, HI, 2002, pp. 1588–1593.
6) X. Xie, W. Zhang, and Z. Yang, "A dissipative particle swarm optimization," in Proc. Congr. Evol. Comput., Honolulu, HI, 2002, pp.1456–1461.
7) T. Peram, K. Veeramachaneni, and C. K. Mohan, "Fitness-distance-ratio based particle swarm optimization," in Proc. Swarm Intelligence Symp., 2003, pp. 174–181.
8) Kennedy, J.: Bare bones particle swarms. In Proc. of the Swarm Intelligence Symposium (SIS 2003), 2003.
9) Mendes, R. (2004). Population Topologies and Their Influence in Particle Swarm Performance. PhD Thesis, Universidade do Minho, Portugal.
10) R. Mendes, J. Kennedy, and J. Neves, "The fully informed particle swarm: Simpler, maybe better," *IEEE Trans. Evol. Comput.*, vol. 8, pp.204–210, Jun. 2004.
11) F. van den Bergh and A.P. Engelbrecht: A cooperative approach to particle swarm optimization. *IEEE Trans. Evol. Comput.*, vol.8, pp.225-239, Jun. 2004.
12) A. Ratnaweera, S. Halgamuge, and H. Watson, "Self-organizing hierarchical particle swarm optimizer with time varying accelerating coefficients," *IEEE Trans. Evol. Comput.*, vol. 8, pp. 240–255, Jun. 2004.
13) J.J. Liang, A.K.Qin, P.N. Suganthan, and S. Baskar: Comprehensive Learning Particle Swarm Optimizer for Global Optimization of Multimodal Functions. *IEEE Trans. Evol. Comput.*, vol.10, No.3, pp.281 – 295, Jun. 2006.

References – continued…

Speciation and niching:

1) A. Petrowski, “A clearing procedure as a niching method for Genetic Algorithms,” in Proc. of the 1996 IEEE International Conference on Evolutionary Computation, 1996, pp.798–803.
2) R. Brits, A.P. Engelbrecht, and F. van den Bergh, “A niching particle swarm optimizer,” in Proc. of the 4th Asia-Pacific Conference on Simulated Evolution and Learning 2002 (SEAL 2002), 2002, pp.692–696.
3) J.P. Li, M.E. Balazs, G. Parks and P.J. Clarkson, “A species conserving genetic algorithm for multimodal function optimization,” Evol. Comput., vol.10, no.3, pp.207–234, 2002.
4) X. Li, “Adaptively choosing neighbourhood bests using species in a particle swarm optimizer for multimodal function optimization,” in Proc. of Genetic and Evolutionary Computation Conference 2004 (GECCO’04), LNCS 3102, eds. Deb, K. et al., 2004, pp.105–116.
5) K.E. Parsopoulos and M.N. Vrahatis, “On the computation of all global minimizers through Particle Swarm Optimization,” IEEE Trans. Evol. Comput., vol.8, no.3, Jun. 2004, pp.211–224.
6) Bird, S. and Li, X.(2006), "Adaptively Choosing Niching Parameters in a PSO", in Proceeding of Genetic and Evolutionary Computation Conference 2006 (GECCO'06), eds. M. Keijzer, et al., p.3 - 9, ACM Press.
7) Bird, S. and Li, X.(2006), "Enhancing the robustness of a speciation-based PSO", in Proceeding of Congress of 2006 Evolutionary Computation (CEC'06), p.3185 - 3192, IEEE Service Center, Piscataway, NJ 08855-1331.

References – continued…

Optimization in dynamic environments:

1) R. C. Eberhart and Y. Shi. Tracking and optimizing dynamic systems with particle swarms. In Proc. the 2001 Congress on Evolutionary Computation CEC2001, p.94–100. IEEE Press, 27-30 May 2001.
2) J. Branke, Evolutionary Optimization in Dynamic Environments. Norwell, MA: Kluwer Academic Publishers, 2002.
3) A. Carlisle and G. Dozier. Tracking changing extrema with adaptive particle swarm optimizer. In Proc. World Automation Cong.,, pages 265–270, Orlando FL USA, 2002.
4) X. Hu and R. Eberhart. Adaptive particle swarm optimisation: detection and response to dynamic systems. In Proc. Congress on Evolutionary Computation, p.1666–1670, 2002.
5) T. Blackwell and P. Bentley. Dynamic search with charged swarms. In Proc. the Workshop on Evolutionary Algorithms Dynamic Optimization Problems (EvoDOP-2003), pages 19–26, 2002.
6) T. Blackwell and J. Branke. Multi-swarm optimization in dynamic environments. In LNCS, No. 3005, Proc. Of Applications of Evolutionary Computing: EvoWorkshops 2004: EvoBIO, EvoCOMNET, EvoHOT, EvoISAP, EvoMUSART, and EvoSTOC, pages 489–500, 2004.
7) D. Parrott and X. Li, “A particle swarm model for tacking multiple peaks in a dynamic environment using speciation,” in Proc. of the 2004 Congress on Evolutionary Computation, 2004, pp.98–103.
8) T. Blackwell and J. Branke. Multi-swarms, exclusion, and anti-convergence in dynamic environments. *IEEE Trans. on Evol. Compu.*, Vol.**10**, No.4, August 2006, pp.459-472.
9) Parrott, D. and Li, X. (2006), "Locating and Tracking Multiple Dynamic Optima by a Particle Swarm Model using Speciation", *IEEE Trans on Evol. Compu.*, Vol.**10**, No.4, August 2006, pp.440-458.
10) Li, X., Branke, J. and Blackwell, T. (2006), "Particle Swarm with Speciation and Adaptation in a Dynamic Environment ", in Proceeding of Genetic and Evolutionary Computation Conference 2006 (GECCO'06), eds. M. Keijzer, et al., p.51 - 58, ACM Press.

ion

References – continued…

Multiobjective optimization:

1) Deb, K.: Multi-Objective Optimization using Evolutionary Algorithms, John Wiley & Sons, Chichester, UK (2001).
2) Deb, K., Agrawal,S., Pratap, A. and Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation 6(2): 182-197 (2002).
3) Hu, X. and Eberhart, R.: Multiobjective Optimization Using Dynamic Neighbourhood Particle Swarm Optimization. In Proceedings of the IEEE World Congress on Computational Intelligence, Hawaii, May 12-17, 2002. IEEE Press (2002).
4) Coello, C.A.C. and Lechuga, M.S.: MOPSO: A Proposal for Multiple Objective Particle Swarm Optimization, in Proceedings of Congress on Evolutionary Computation (CEC'2002), Vol. 2, IEEE Press (2002) 1051-1056.
5) Mostaghim, S. and Teich, J.: Strategies for finding good local guides in Multi-Objective Particle Swarm Optimization (MOPSO. In Proc. 2003 IEEE Swarm Intelligence Symp., Indianapolis, IN, Apr. 2003, pp.26-33.
6) Fieldsend, J.E. and Singh, S.: A multi-objective algorithm based upon particle swarm optimization, an efficient data structure and turbulence. In Proc. 2002 U.K. Workshop on Computational Intelligence, Birmingham, U.K., Sept. 2002, pp.37-44.
7) Li, X.: A Non-dominated Sorting Particle Swarm Optimizer for Multiobjective Optimization, in Erick Cant-Paz et al. (editors), Genetic and Evolutionary Computation - GECCO 2003. Proceedings, Part I, Springer, LNCS Vol. 2723, (2003) 37-48.
8) C. A. C. Coello, G. T. Pulido, and M. S. Lechuga MS, "Handling multiple objectives with particle swarm optimization," *IEEE Trans. Evol. Comput.*, vol. 8, no. 3, pp. 256–279, Jun. 2004.
9) M. Reyes-Sierra and C.A.C. Coello. Multi-objective particle swarm optimizers:A survey of the state-of-the-art. International Journal of ComputationalIntelligence Research, 2(3):287–308, 2006.

References – continued…

Constraint handling:

1) Z. Michalewicz and M. Schoenauer. Evolutionary Algorithms for Constrained Parameter Optimization Problems. *Evolutionary Computation*, 4(1):1–32, 1996.
2) T. P. Runarsson and X. Yao. Stochastic Ranking for Constrained Evolutionary Optimization. *IEEE Transactions on Evolutionary Computation*, 4(3):284–294, September 2000.
3) X. Hu, and R. Eberhart. Solving constrained nonlinear optimization problems with particle swarm optimization. 6th World Multiconference on Systemics, Cybernetics and Informatics (SCI 2002), Orlando, USA.
4) K. Parsopoulos and M. Vrahatis. Particle Swarm Optimization Method for Constrained Optimization Problems. In P. Sincak, J.Vascak, V. Kvasnicka, and J. Pospicha, editors, Intelligent Technologies - Theory and Applications: New Trends in Intelligent Technologies, pages 214–220. IOS Press, 2002. Frontiers in Artificial Intelligence and Applications series, Vol. 76 ISBN: 1-58603-256-9.
5) G. Coath and S. K. Halgamuge. A comparison of constraint-handling methods for the application of particle swarm optimization to constrained nonlinear optimization problems. In Proceedings of the 2003 Congress on Evolutionary Computation, p.2419 - 2425. IEEE, December 2003.
6) J. Zhang and F. Xie. DEPSO: Hybrid particle swarm with differential evolution operator. In Proceedings of IEEE International Conference on Systems, Man and Cybernetics, p.3816-3821. IEEE, October 2003.
7) G. Toscano and C. Coello. A constraint-handling mechanism for particle swarm optimization. In Proceedings of the 2004 Congress on Evolutionary Computation, p.1396 - 1403. IEEE, June 2004.
8) Clerc, M.: *Particle Swarm Optimization*, ISTE Ltd, 2006.