COMP815 Nature Inspired Computing

Particle Swarm Optimization

sakana.ai

We raised $30M to develop nature-inspired AI in Japan

January 16, 2024

Sakana AI is a new AI research company based in Tokyo, Japan. Our founding team have proven track records of developing breakthroughs in AI, where we take pride in setting the trend in AI development. We aim to develop transformative AI that will bring us into the next paradigm. The main focus of our research and development of new kinds of foundation models based on nature-inspired intelligence.

The name *sakana* is derived from the Japanese word さかな (sa-ka-na) which means *fish*. Our logo is meant to invoke the idea of a school of fish coming together and forming a coherent entity from simple rules as we want to make use of ideas from nature such as evolution and collective intelligence in our research. The red fish swimming away represents our desire to not simply do what everyone else does, but to pursue what we believe is coming next!

- Nature-Inspired AI at the LLM era
- Evolution
- Collective Intelligence
  e.g., Swarm Intelligence
  Fish schooling

Collective Intelligence

Collective Intelligence
Volume 1, Issue 1, August 2022

https://doi.org/10.1177/26339137221114874

SAGE journals

Synthetic Article & Review

Collective intelligence for deep learning: A survey of recent developments

David Ha and Yujin Tang

Neuro Evolution
(our Lecture 10)

NeuroEvoBench: Benchmarking Evolutionary Optimizers for Deep Learning Applications

Robert Tjarko Lange*
Technical University Berlin
Science of Intelligence Cluster of Excellence

Yujin Tang
Google DeepMind

Yingtao Tian
Google DeepMind

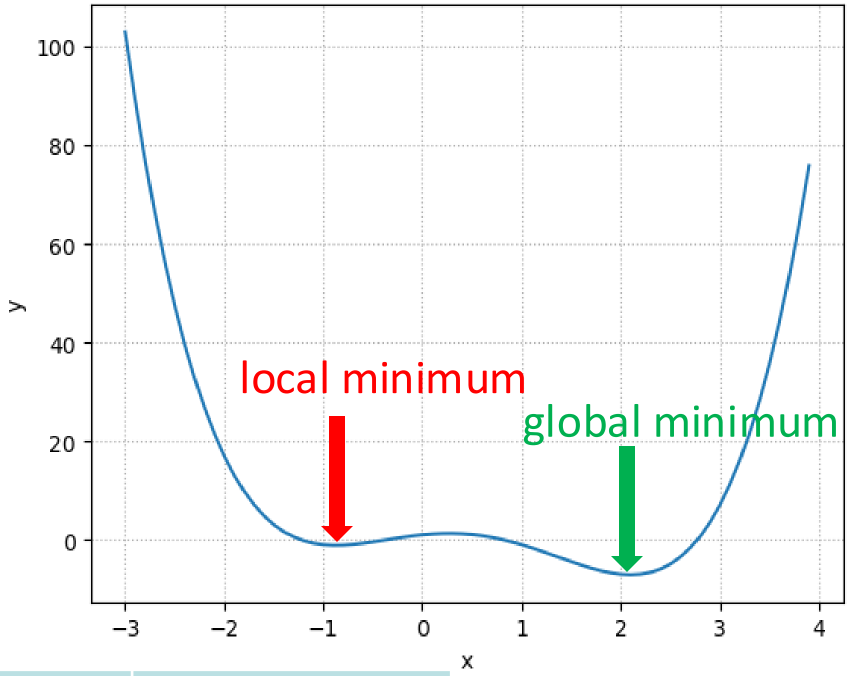
NeurIPS 2023

Previous Lecture – Evolution Algorithm

- Genetic Algorithm
  chromosome, crossover, mutation, rank/fitness

  min $f(x) = x^4 - 2x^3 - 3x^2 + 2x + 1$

  subject to $-3 \leq x \leq 4$



x	chromosome
-3	00 0000 0000
0	01 1011 0110
4	11 1111 1111

chromosome	x
00 0000 0000	-3
01 1011 0110	-0.003
11 1111 1111	4

Previous Lecture – Evolution Algorithm

- 8-Queen Problem
- Travelling Salesman Problem
- Chromosome, Crossove itness

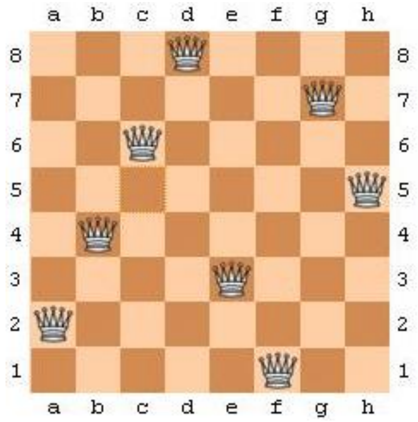
Crossover: important to al !!!!

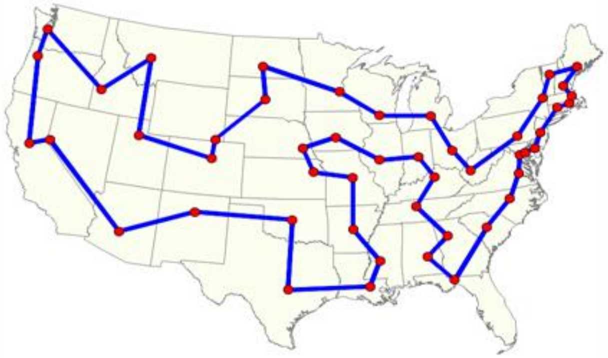
Assignment 1 Part 1

— tools.cxOrdered()

Crossover

- cxOnePoint()
- cxTwoPoint()
- cxUniform()
- cxPartialyMatched()
- cxUniformPartialyMatched()
- cxOrdered()
- cxBlend()





Previous Lecture – Evolution Algorithm

- Evolution Strategy

  Continuous variables

  Individual representation

  - Decision variables $\mathbf{X}_k = (x_1, \dots, x_D)_k$
  - Fitness $f\left(\mathbf{x}_k\right)$
  - A set of evolvable strategy parameters $\mathbf{s}_k$

  Mutation

  $$x_i = x_i + z, \qquad \text{where} \quad z = \mathbb{N}(0, \sigma)$$

- Differential Evolution

Nature		Computing
Evolution Genetics	→	Evolutionary Algorithms Genetic Algorithms
Social Behaviour	→	Swarm Intelligence Algorithms

Particle Swarm Optimization (PSO) Algorithm

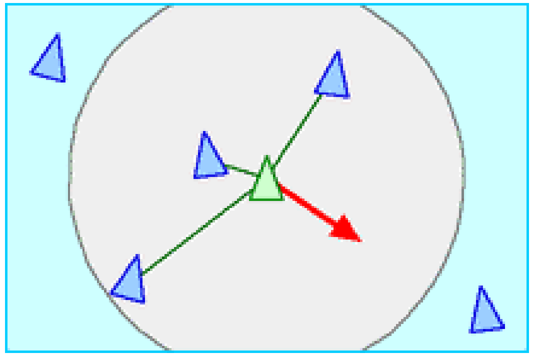
- Introduced by Kennedy and Eberhart in 1995
- Inspired by the flocking behaviours of birds and fish

Particle Swarm Optimization (PSO) Algorithm

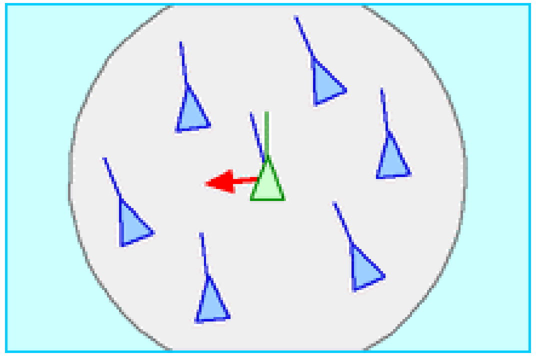
[Swarm Intelligence: Unlocking Nature's Secrets]

Elements of Flocking

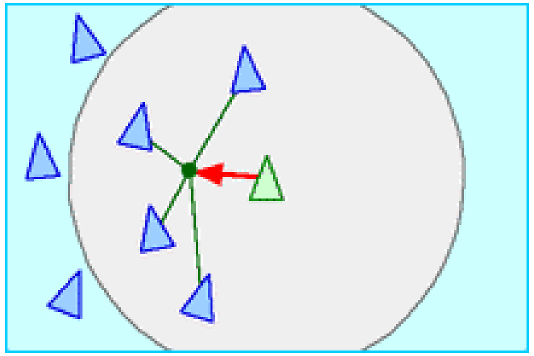
- Reynolds generated realistic visual simulation of bird flocking in 1986
- Three individual behaviour required to flock



Separation



Alignment



Cohesion

Concept of PSO

- Combines self-experience with social experience

- Particles (individuals) constitute a swarm moving around in the search space looking for the **best solution**
- Each particle adjust its movement according to its own experience and that of other particles
  - Its own best solution (personal best)
  - Best value among all particles (global best)

Algorithm Parameters

- $\boldsymbol{X}$: population of particles $\{x_i\}$
- $\boldsymbol{x_i}$: position of particle x_i
- $\boldsymbol{f}$: objective function
- $\boldsymbol{v_i}$: velocity of particle x_i
- c_1 : *weight of local information*
- c_2 : *weight of global information*

The Algorithm

```
Initialize_Particles(X)
for i=1 to n_iter
    for each particle x in X do
        fp = f(x);
        if i == 1 or fp is better than f(pBest)
            pBest(x) = x;
        end
    end
    gBest = best x in X;
    for each particle x in X do
        v = v + c1*rand*(pBest - x) + c2*rand*(gBest - x);
        x = x + v;
    end
end
```

Positions and velocities are randomly generated

Update personal best *w.r.t* iters

Find the best particle

Adjust velocity

Move particle to next position

Updates

- Synchronous
  - Compute new velocities of all particles
  - Positions are updated with new velocities
  - Then find the global best
- Asynchronous
  - Update global best immediately after the new position of a particle is updated

Velocity Update

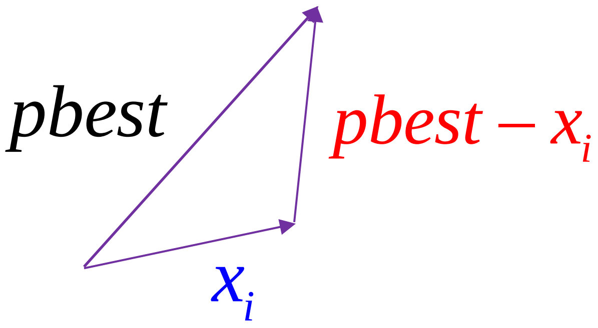
$$v_i^{(k+1)} = v_i^{(k)} + \underbrace{c_1 \cdot rand \cdot \left(pbest_i^{(k)} - x_i^{(k)} \right)}_{\text{Personal experience}} + \underbrace{c_2 \cdot rand \cdot \left(gbest^{(k)} - x_i^{(k)} \right)}_{\text{Social influence}}$$

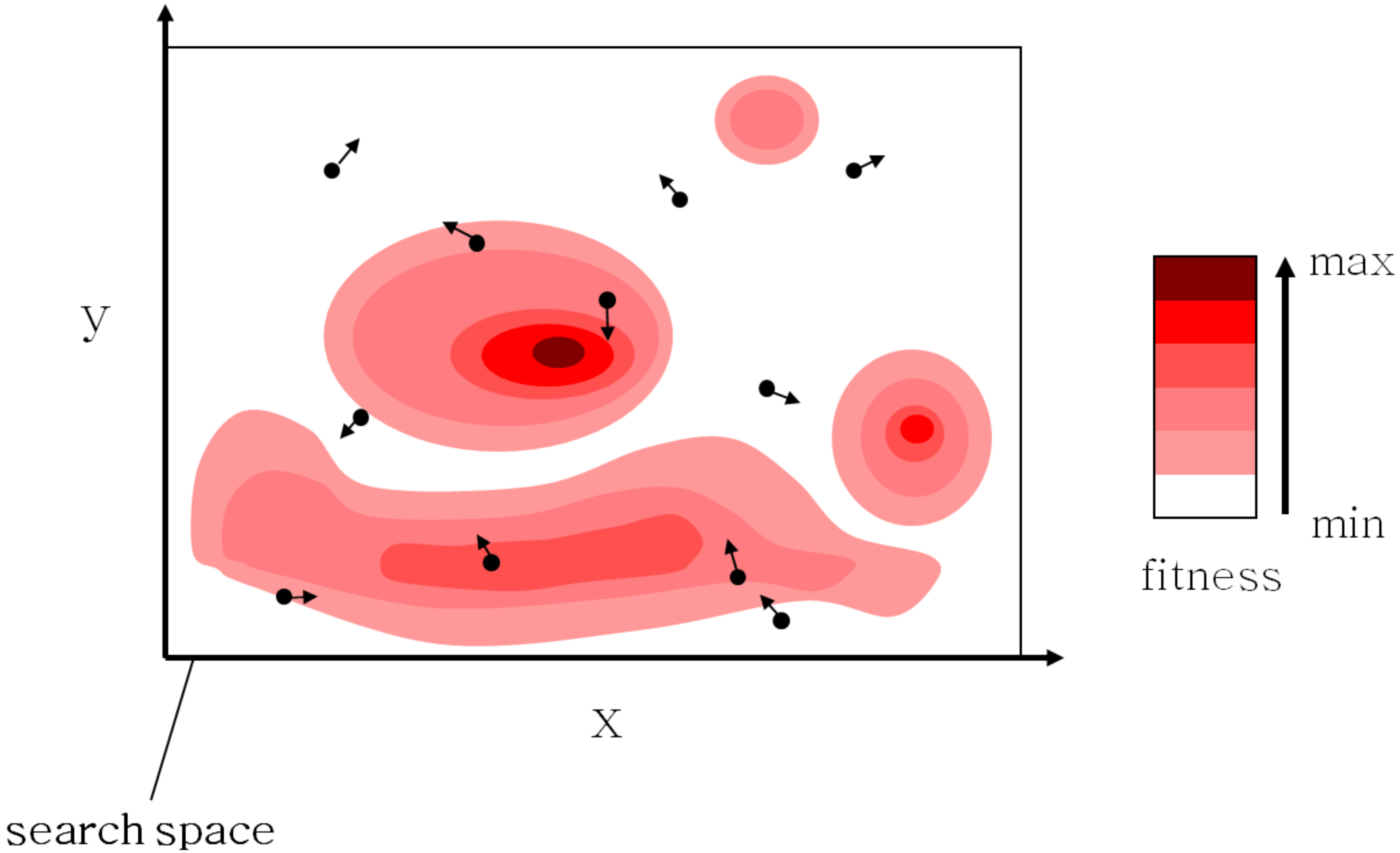
Exploration

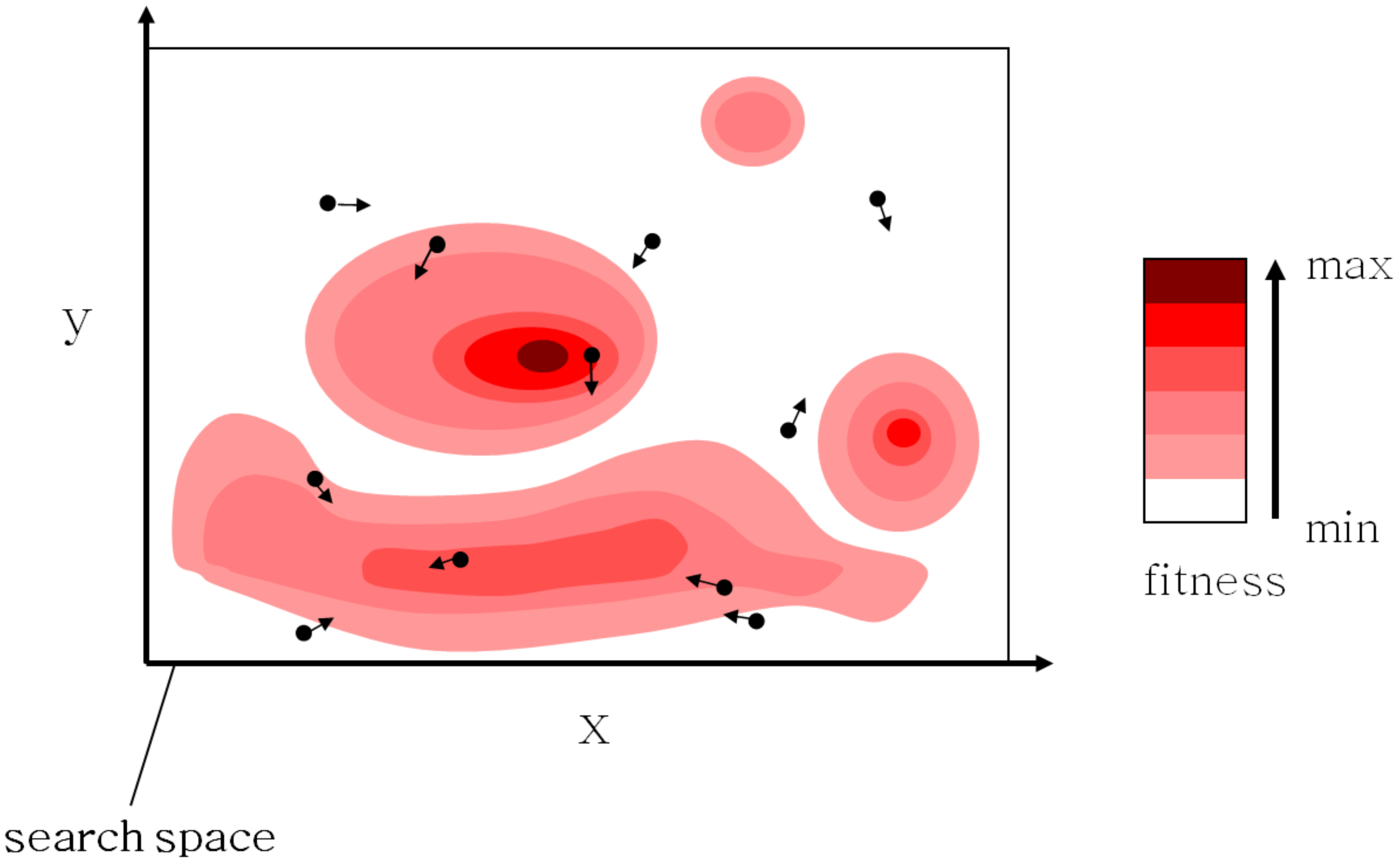
Exploitation

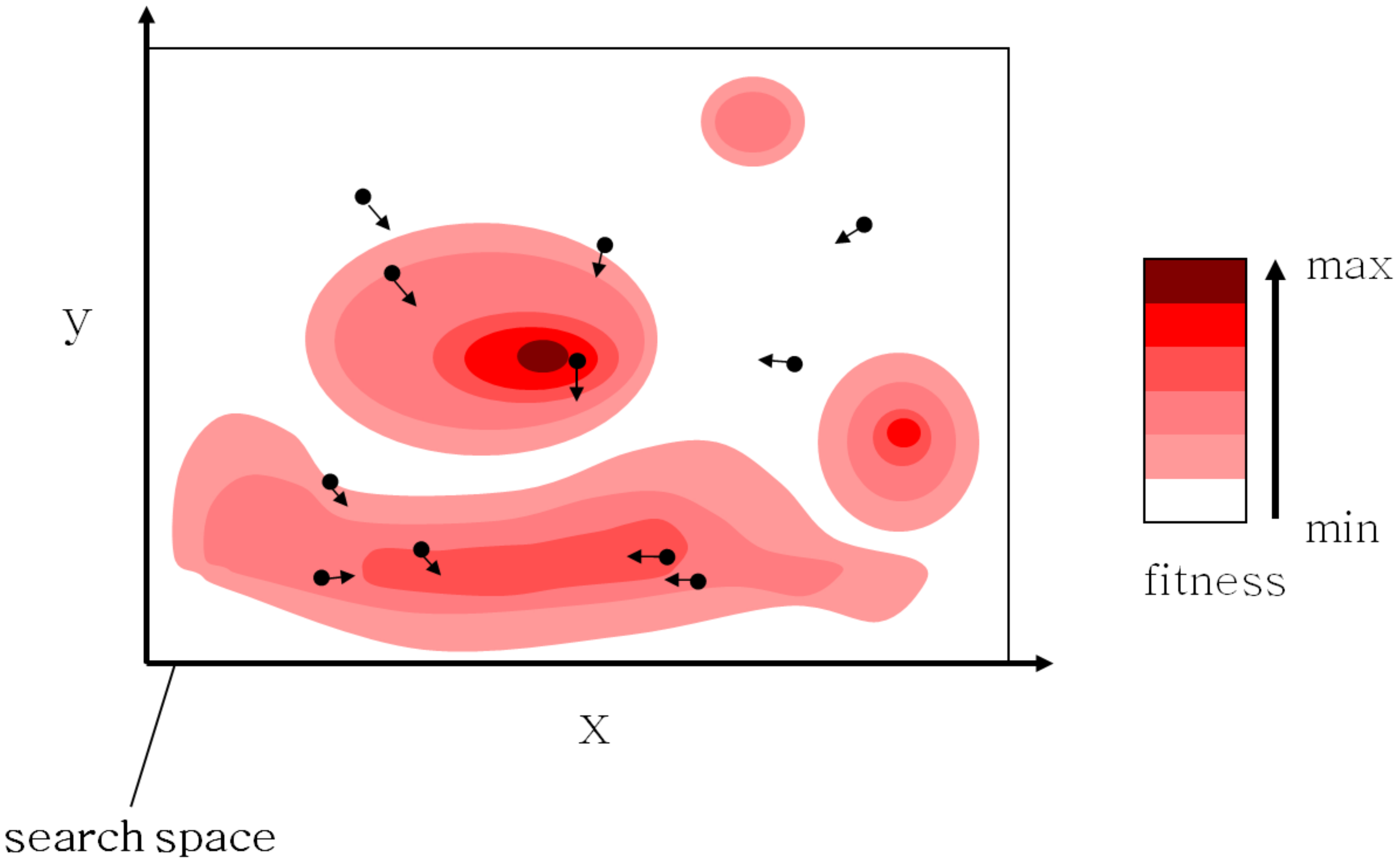
Note:

v_i, x_i, *pbest*, and *gbest* are *D*-dimensional vectors for problems with *D* decision variables

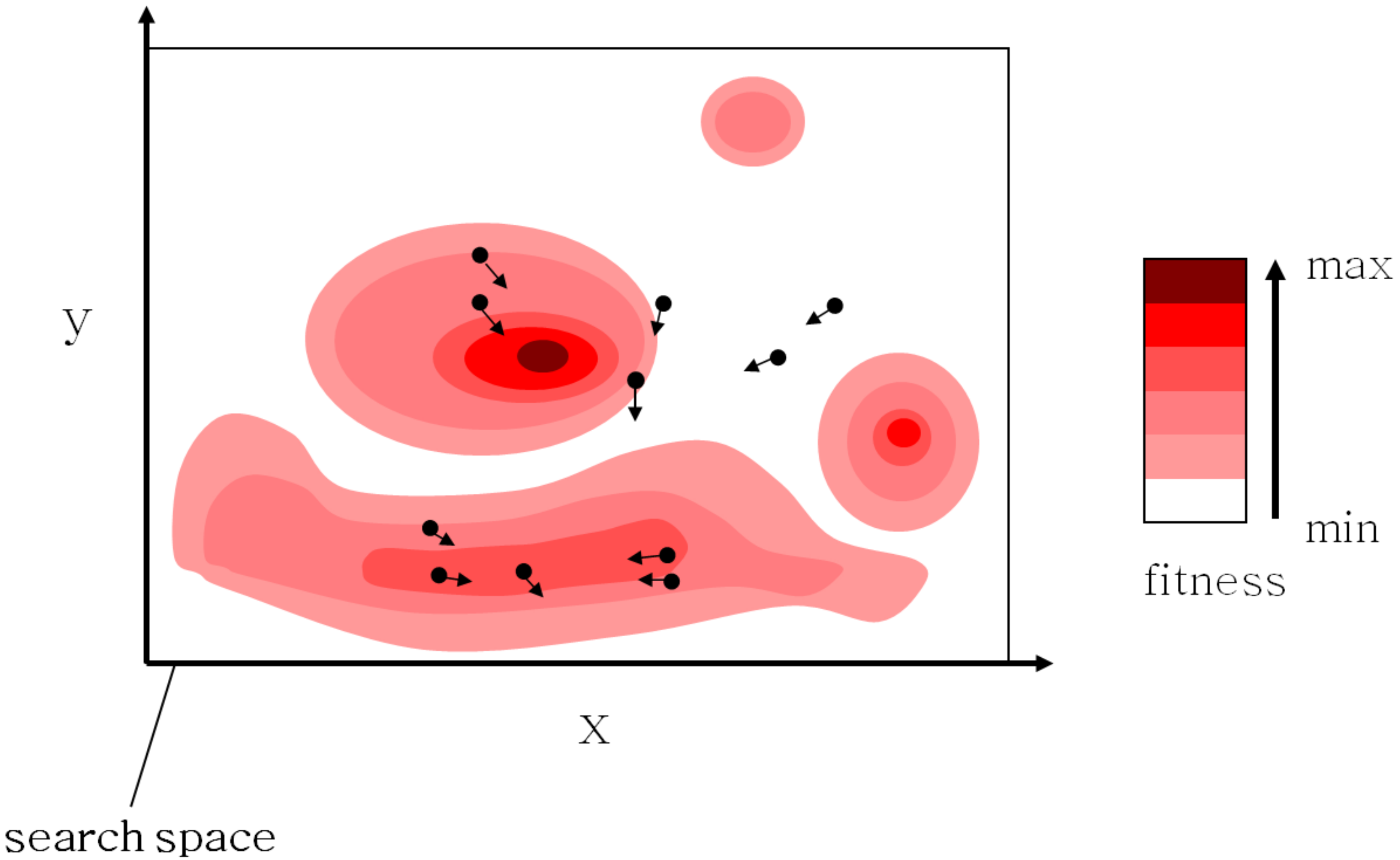


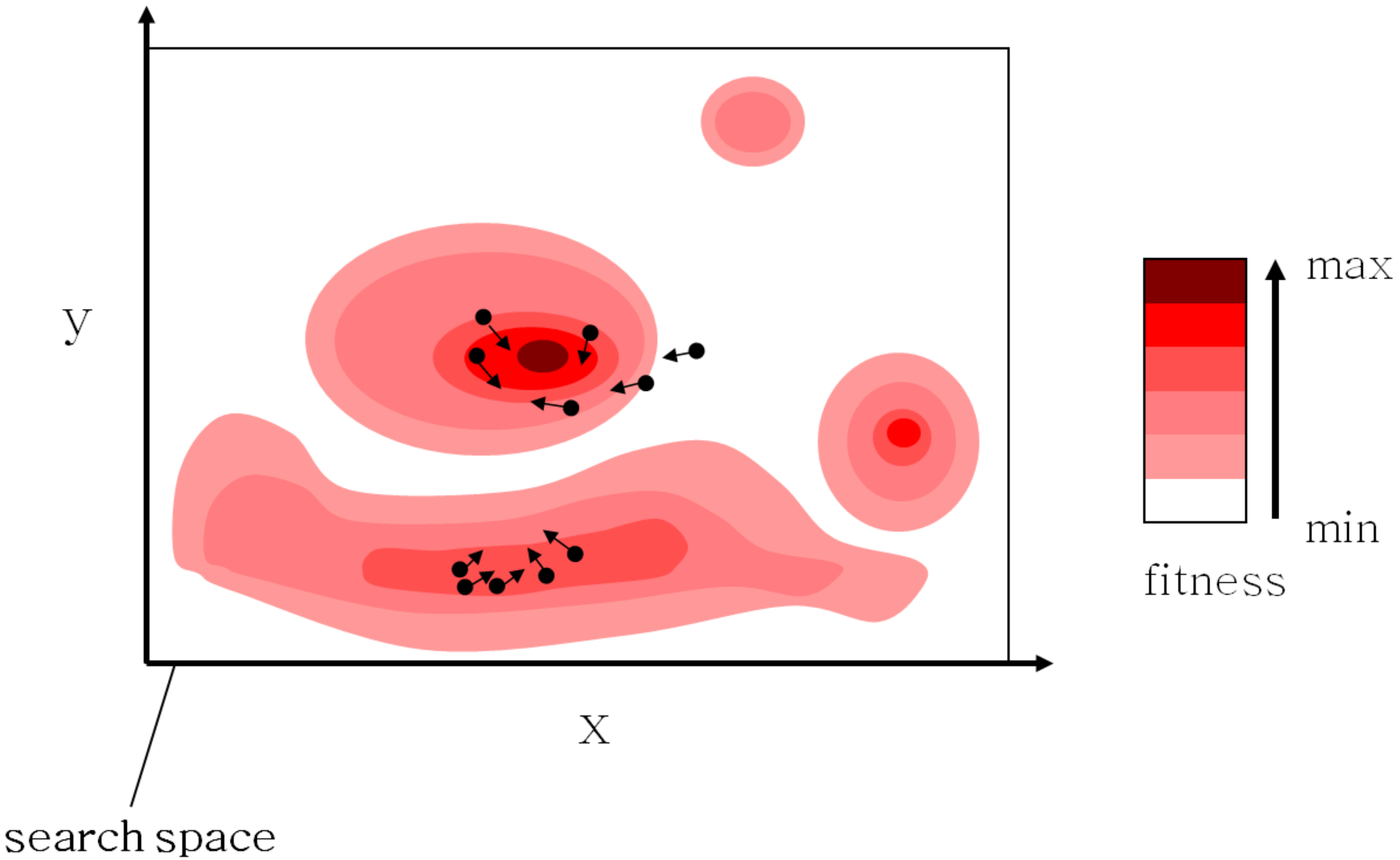

y
x
search space
max
min
fitness

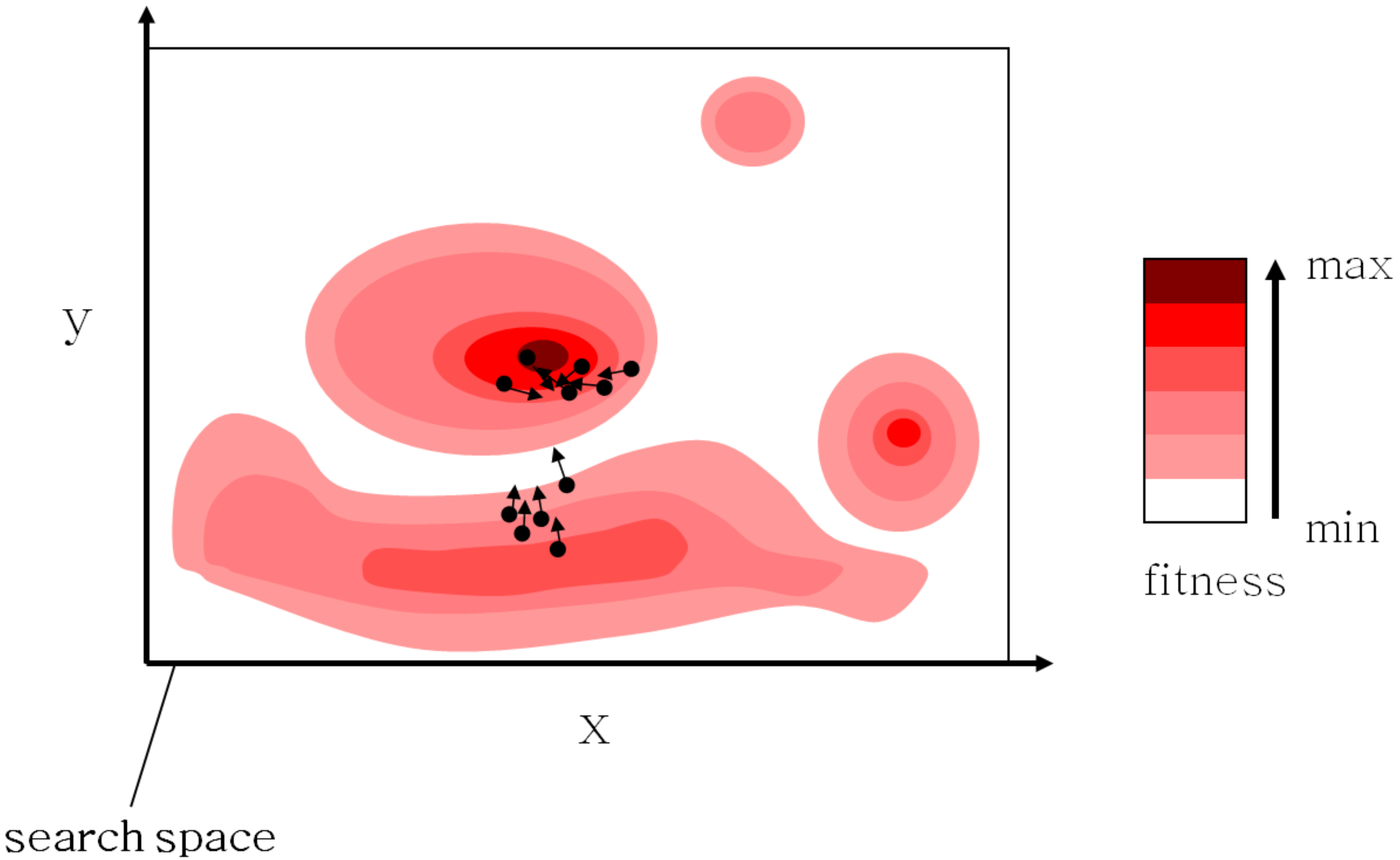

y
X
search space
max
min
fitness

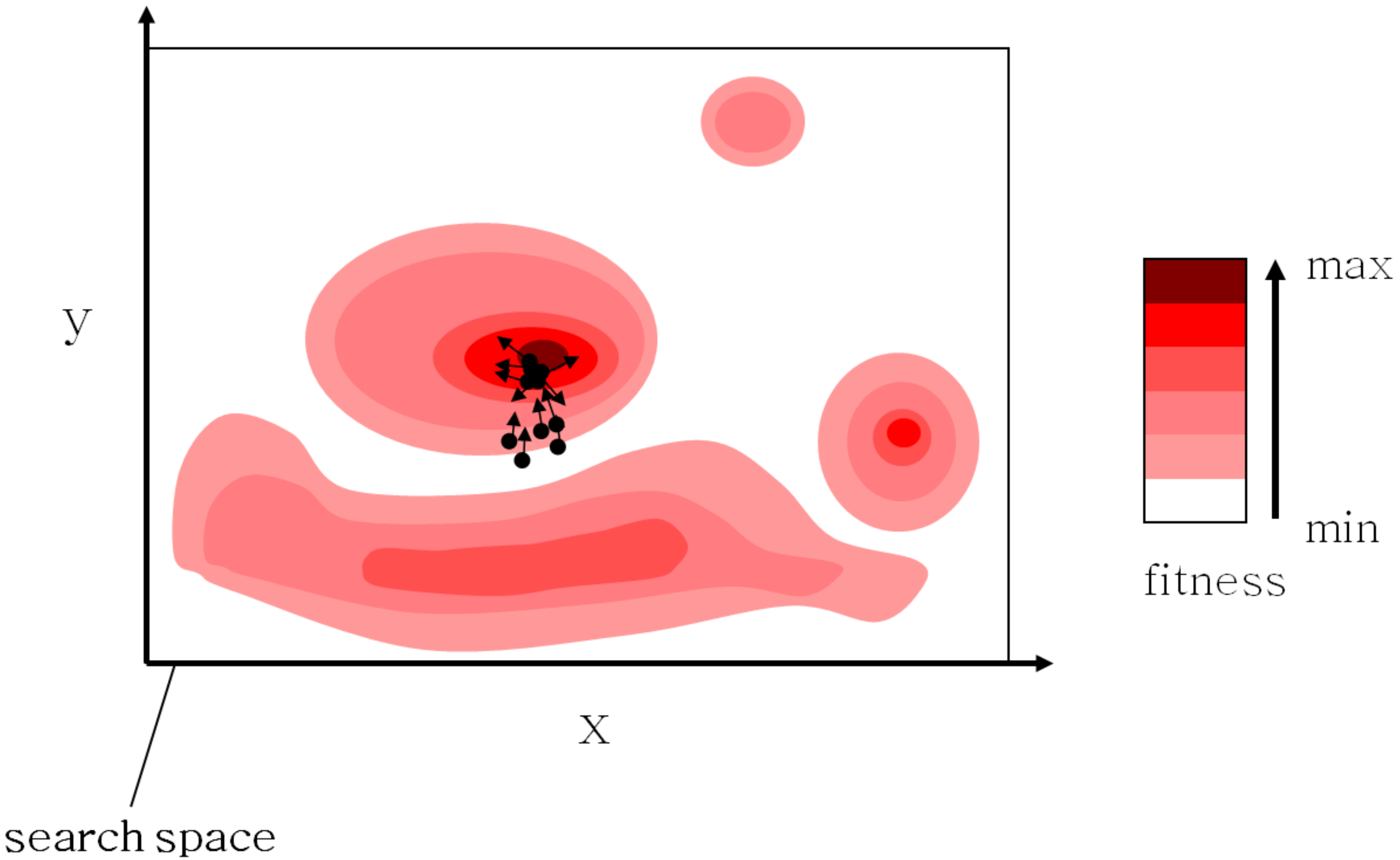


y
x
search space
max
min
fitness


y
x
search space
max
min
fitness


y
x
search space
max
min
fitness


y
X
search space
max
min
fitness


y
max
min
fitness
x
search space

Velocity Control

- A range for v_i needs to be set
  - For initialization
  - To avoid its magnitude becoming too large

$$\left[-v_{\max}, v_{\max}\right]$$

- Performance suffers if the range is not set appropriately
- Further control can be implemented by adding an inertial factor

$$v_i^{(k+1)} = \lambda \cdot v_i^{(k)} + c_1 \cdot rand \cdot \left(pbest_i^{(k)} - x_i^{(k)} \right) + c_2 \cdot rand \cdot \left(gbest^{(k)} - x_i^{(k)} \right)$$

Typically, in the range [0.4, 0.9]

Constriction Coefficient

$$v_i^{(k+1)} = \chi\left(v_i^{(k)} + c_1 \cdot rand \cdot \left(pbest_i^{(k)} - x_i^{(k)}\right) + c_2 \cdot rand \cdot \left(gbest^{(k)} - x_i^{(k)}\right)\right)$$

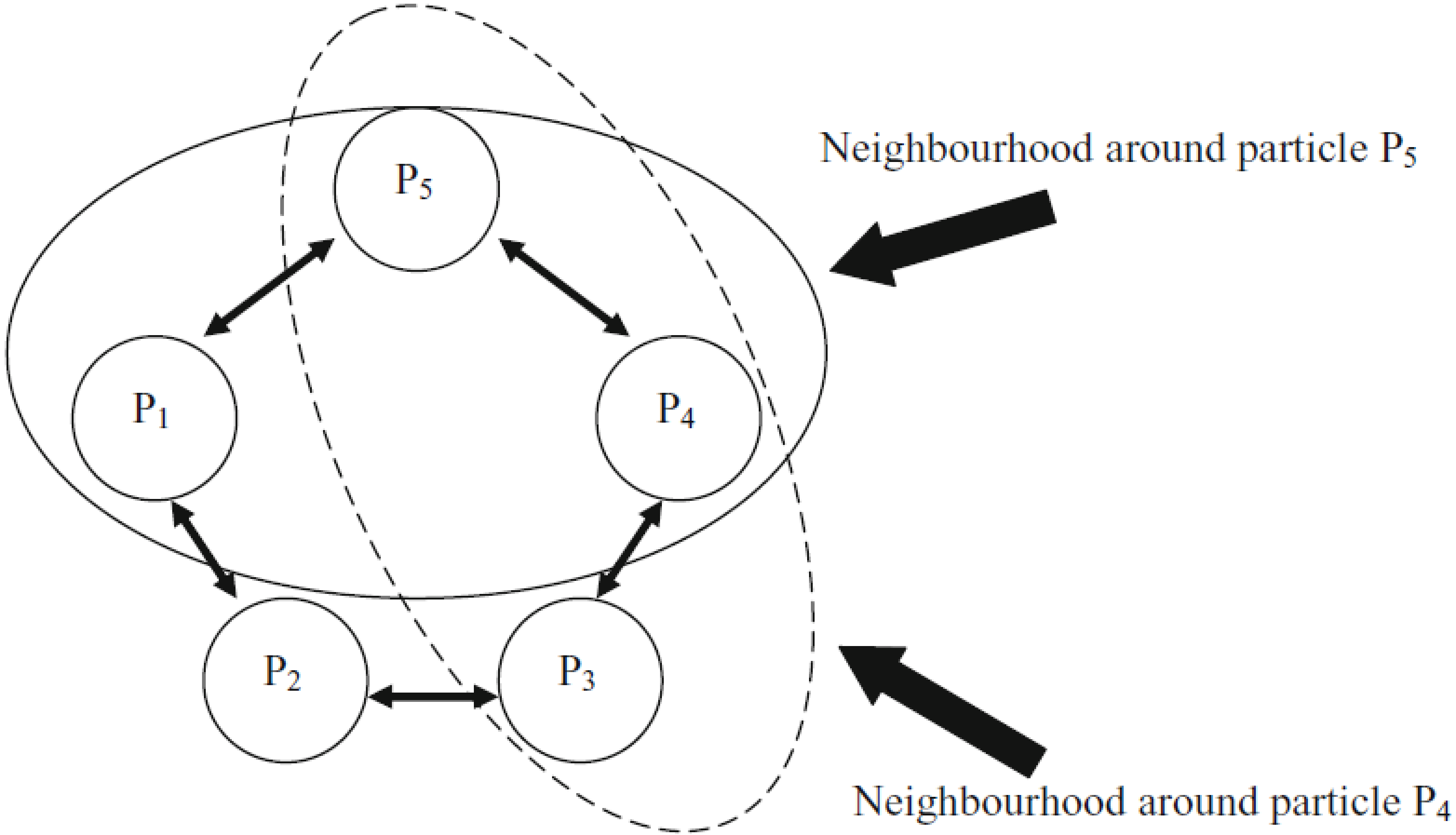
$$\chi = \frac{2}{\left|2 - c - \sqrt{\left(c^2 - 4c\right)}\right|}$$

$$\boxed{\begin{array}{l} c = c_1 + c_2 \\ c > 4 \end{array}}$$

Neighbourhood Structure

- Instead of global best, the algorithm can use a *local best* – best value within a neighbourhood
- How neighbourhood is defined is dependent on the problem
- Neigbourhood
  - Size = 1 implies each particle is independent
  - Size = *N* is the same as the global version
  - Neigbourhood regions do not overlap – there are multiple independent subswarms

- Neigbourhoods overlap – information flow gradually between them



Maintaining Diversity

- Avoid pre-mature convergence

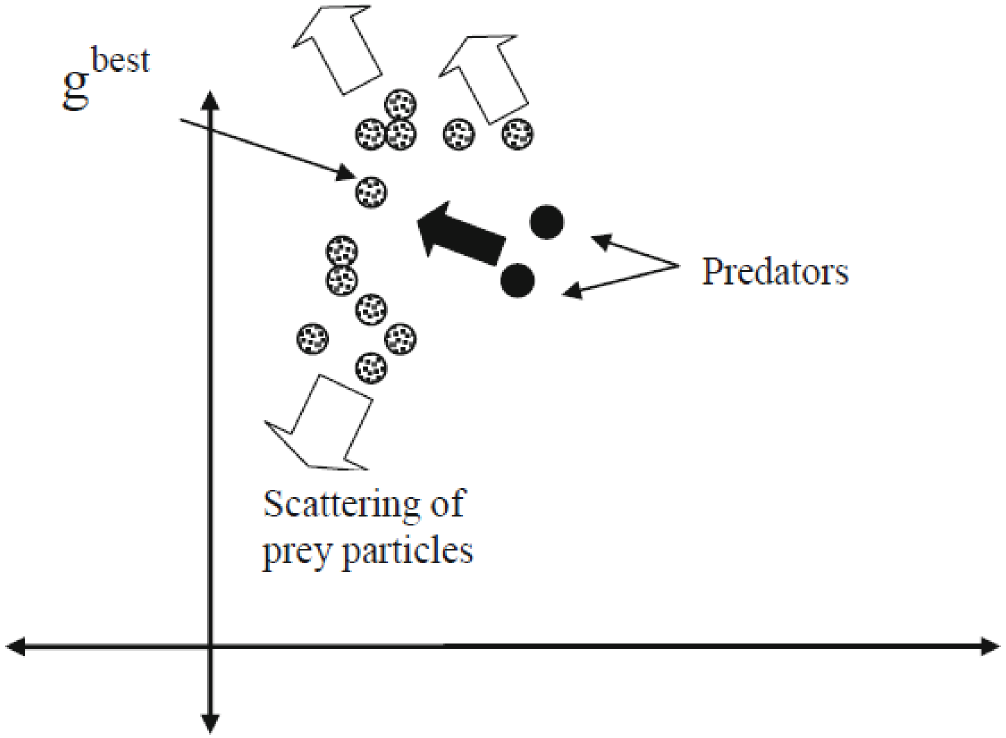
$$x_i = pbest_i = gbest \quad \Rightarrow \text{ Convergence to global optimum?}$$

- Cope with dynamic environment
  - Past experiences may have little relevance (exploration vs. exploitation)

- Uncover multiple, equally good solutions if they exist

Predator-Prey PSO

- Two types of particles
  - *Predators – attracted to the best particles in the swarm*
  - *Prey – repelled by predator particles*
- Single or a very small number of predators are used



Velocity Update

$$v_{predator}^{(k+1)} = \alpha\left(gbest - x_{predator}^{(k)}\right)$$

$$v_{prey}^{(k+1)} = \lambda v_{prey}^{(k)} + c_1 r^{(k)}\left(pbest_{prey}^{(k)} - x_{prey}^{(k)}\right)$$
$$+c_2 r^{(k)}\left(gbest^{(k)} - x_{prey}^{(k)}\right) + c_3 r^{(k)} A^{(k)}(d)$$

$$A(d) = ae^{-bd}$$

Distance between predator and prey

Magnitude of repulsion force

Scaling constant

Variety of PSO

Davoud Sedighizadeh and Ellips Masehian, "Particle Swarm Optimization Methods, Taxonomy and Applications," *International Journal of Computer Theory and Engineering*, Vol. 1, No. 5, December 2009

2-D Otsu PSO

Adaptive PSO

Dynamic and Adjustable PSO

Active Target PSO

2-D Otsu

Cooperative Multiple PSO

Adaptive Mutation PSO

Extended Particle Swarm

Summary

- Particle Swarm Optimisation (Swarm Intelligence)

- Principles → Algorithm
  Velocity update
  Velocity control/constraint

- Predator-Prey PSO