Natural Computing

Lecture 12

Michael Herrmann
mherrman@inf.ed.ac.uk
phone: 0131 6 517177
Informatics Forum 1.42

28/10/2011

Particle Swarm Optimisation (PSO)

Swarm intelligence

- Collective intelligence: A super-organism emerges from the interaction of individuals
- The super-organism has abilities that are not present in the individuals ('is more intelligent')
- "The whole is more than the sum of its parts"
- Mechanisms: Cooperation and competition self-organisation, ... and communication
- Examples: Social animals (incl. ants), smart mobs, immune system, neural networks, internet, swarm robotics

Beni, G., Wang, J.: Swarm Intelligence in Cellular Robotic Systems, Proc. NATO Adv. Workshop on Robots and Biological Systems, Tuscany, Italy, 26–30/6 (1989)

Swarm intelligence: Application areas

- Biological and social modelling
- Movie effects
- Dynamic optimization
  - routing optimization
  - structure optimization
  - data mining, data clustering
- Organic computing
- Swarm robotics

Swarms in robotics and biology

AI/Robotics

- Main interest in pattern synthesis
  - Self-organization
  - Self-reproduction
  - Self-healing
  - Self-configuration
- Construction

Biology/Sociology

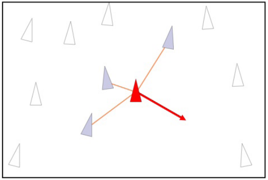
- Main interest in pattern analysis
  - Recognizing best pattern
  - Optimizing path
  - Minimal conditions
  - not "what", but "why"
- Modelling

Dumb parts, properly connected into a swarm, yield smart results.

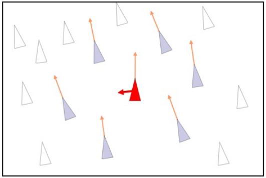
Kevin Kelly

Complex behaviour from simple rules

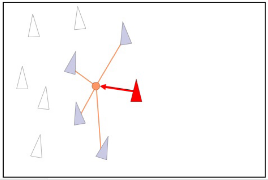
Rule 1: **Separation**
Avoid Collision with neighbouring agents



Rule 2: **Alignment**
Match the velocity of neighbouring agents



Rule 3: **Cohesion**
Stay near neighbouring agents



Towards a computational principle

- Evaluate your present position
- Compare it to your previous best and neighbourhood best
- Imitate self and others

Hypothesis: There are two major sources of cognition, namely, own experience and communication from others.

Leon Festinger, 1954/1999, Social Communication and Cognition

Particle Swarm Optimization (PSO)

- Methods for finding an optimal solution to an objective function
- Direct search, i.e. gradient free
- Simple and quasi-identical units
- Asynchronous; decentralized control
- 'Intermediate' number of units: $\sim 10^1 - 10^3$ (or more)
- Redundancy leads to reliability and adaptation
- PSO is one of the computational algorithms in the field of swarm intelligence (another one is ACO)

J. Kennedy, and R. Eberhart, Particle swarm optimization, in Proc. IEEE. Int. Conf. on Neural Networks, Piscataway, NJ, pp. 1942–1948, 1995.

PSO algorithm: Initialization

Fitness function	$f : \mathbb{R}^m \to \mathbb{R}$
Number of particles	$n = 20 \ldots 200$
Particle positions	$x_i \in \mathbb{R}^m, \; i = 1 \ldots n$
Particle velocities	$v_i \in \mathbb{R}^m, \; i = 1 \ldots n$
Current best of each particle (“simple nostalgia”)	$\hat{x}_i$
Global best (“group norm”)	$\hat{g}$
Constants	$\omega, \; \alpha_1, \; \alpha_2$

The canonical PSO algorithm

For each particle (for all members in the swarm) $i = 1 \ldots n$

- Create random vectors r_1, r_2 with components in $U[0,1]$
- update velocities

$$v_i \leftarrow \omega v_i + \alpha_1 r_1 \circ (\hat{x}_i - x_i) + \alpha_2 r_2 \circ (\hat{g} - x_i)$$

$\circ$: componentwise multiplication

- update positions

$$x_i \leftarrow x_i + v_i$$

- update local bests (for a minimisation problem)
  $\hat{x}_i \leftarrow x_i$ if $f(x_i) < f(\hat{x}_i)$
- update global best (for a minimisation problem)
  $\hat{g} \leftarrow x_i$ if $f(x_i) < f(\hat{g})$

Initialisation

- Initialize the particle positions and their velocities

$X = \text{lower}_{\text{limit}} + (\text{upper}_{\text{limit}} - \text{lower}_{\text{limit}}) \times \text{rand}(n_{\text{particles}}, m_{\text{dimensions}})$

assert X.shape $== (n_{\mathrm{particles}}, m_{\mathrm{dimensions}})$

$V = \text{zeros}(X.\text{shape})$

- Initialize the global and local fitness to the worst possible

$\mathit{fitness}_{\text{gbest}} = \infty$

$\mathit{fitness}_{\text{lbest}} = \mathit{fitness}_{\text{gbest}} \times \text{ones}(n_{\text{particles}})$

- Initialize parameters

$\omega = 0.1$ (range $0.01 \ldots 0.7$)

$\alpha_1 = \alpha_2 = 2$ (range $0 \ldots 4$, both equal)

$n = 25$ (range $20 \ldots 200$)

max velocity no larger than range of x (or 10-20% of this range)

PSO:

```
                do:                                   [starts outer loop until termination]
cost_X          = evaluate_cost(X)                    [evaluate cost of each particle]
                for i = 1 .. n_particles:             [update local bests]
                if cost_X[i] < cost_lbest[i]:
cost_lbest[i]   = cost_X[i]
                for j = 1 .. m_dimensions:
X_lbest[i][j]   = X[i][j];
                end j; end i;
min_cost_idx    = argmin(cost_X)                      [update global best]
min_cost        = cost_X[min_cost_idx]
                if min_cost < cost_gbest:
cost_gbest      = min_cost;
X_gbest         = X[min_cost_idx,:]
                for i = 1 .. n_particles:             [update velocities and positions]
                for j = 0 .. m_dimensions:
R1/2            = uniform_random_number()
V[i][j]         = (w*V[i][j] + α1*r2*(X_lbest[i][j] - X[i][j]) +
                α2*r2*(X_gbest[j] - X[i][j])) X[i][j] = X[i][j] + V[i][j]
                end j; end i; while not terminated;
```

How does it work?

- **Exploratory behaviour**: Search a broad region of space
- **Exploitative behaviour**: Locally oriented search to approach a (possibly local) optimum

Obviously: Parameters to be chosen to properly balance between exploration and exploitation, i.e. to avoid premature convergence to a local optimum yet still ensure a good rate of convergence to the optimum.

Convergence

- **Failure**: Swarm diverges or remains itinerant
- **Optimally**: Global best approaches global optimum (swarm may still oscillate)
- **Typically**: Global best approaches a local optimum (premature collapse of the swarm)

Mathematical attempts (typically oversimplified): Convergence to global optimum for a 1-particle swarm after infinite time (F. v. d. Bergh, 2001)

see PSO at en.wikipedia.org

Analysis of PSO: Simplified algorithm

- Consider a single particle only ("the view from inside")
- Ignore randomness (use a homogeneous mean value)
- Ignore the global best (assume it equals personal best)
- Keep the personal best constant (changes are rare)
- Set inertia to unity (for the moment only) i.e. what we had (vector equation of i-th particle)

$$v_i(t+1) \leftarrow \omega v_i + \alpha_1 r_1 \circ (\hat{x}_i - x_i(t)) + \alpha_2 r_2 \circ (\hat{g} - x_i(t))$$

- becomes now

$$v_{id}(t+1) = v_{id}(t) + \phi(p_{id} - x_{id}(t)),\ i = 1 \ldots n,\ d = 1 \ldots m$$

- (in component form:$d = 1 \ldots m$ with $p_i = \hat{x}_i$)

Algebraic point of view

Introduce $y(t) = p - x(t)$ in $v(t+1) = v(t) + \phi(p - x(t))$

$$x(t+1) = x(t) + v(t+1) \Rightarrow \begin{cases} v(t+1) & = v(t) + \phi y(t) \\ y(t+1) & = -v(t) + (1-\phi)\, y(t) \end{cases}$$

Introduce state vector $P(t) = (v(t), y(t))^\top$ and

$$M = \begin{pmatrix} 1 & \phi \\ -1 & 1-\phi \end{pmatrix}$$

Starting from the initial state $P(0)$ we have $P(t) = M^t P(0)$

M. Clerc & J. Kennedy (2002) The particle swarm – Explosion, stability, and convergence in a multidimensional complex space. IEEE Transactions on Evolutionary Computation 7:1, 58-73.

Algebraic point of view

Determine eigenvalues of

$$M = \begin{pmatrix} 1 & \phi \\ -1 & 1-\phi \end{pmatrix}, \text{ i.e. } AMA^{-1} = L = \begin{pmatrix} e_1 & 0 \\ 0 & e_2 \end{pmatrix}$$

$$\Rightarrow \qquad e_{1/2} = 1 - \frac{\phi}{2} \pm \frac{\sqrt{\phi^2 - 4\phi}}{2}$$

Transformation matrix: $P(t) = M^t P(0)$

$$\begin{aligned} P(t+1) &= A^{-1}LAP(t) \\ \Rightarrow \qquad AP(t+1) &= LAP(t) \\ Q(t+1) &= LQ(t) \end{aligned}$$

with $Q = AP$ and $A = \begin{pmatrix} \phi + \sqrt{\phi^2 - 4\phi} & 2\phi \\ \phi - \sqrt{\phi^2 - 4\phi} & 2\phi \end{pmatrix}$

Thus $Q(t) = L^t Q(0)$ where $L^t = \begin{pmatrix} e_1 & 0 \\ 0 & e_2 \end{pmatrix}^t = \begin{pmatrix} e_1^t & 0 \\ 0 & e_2^t \end{pmatrix}$

Algebraic point of view

Three cases:

$0 < \phi < 4$

EV complex

$$e_{1/2} = \cos(\theta) \pm i \sin(\theta)$$
$$e_{1/2}^t = \cos(t\theta) \pm i \sin(t\theta)$$

Oscillatory with period k if$\theta = \frac{2k\pi}{t}$

$\phi = 4$

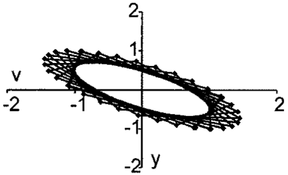
Both EV equal -1

$$MV = -V$$
$$P(t+1) = -P(t)$$

$\phi > 4$

Exponentially divergent

Oscillatory behaviour:



Implications from the algebra

Oscillation for $\phi < 4$: Exploration near current best

Divergence for $\phi > 4$: Exploration of the wider environment

$\phi = \alpha_1 + \alpha_2$ is a combination of the attractiveness of the personal and global best. Since these might be not the same (often they are), a slightly larger ϕ might be needed.

ϕ slightly above 4 (e.g. 4.1): particle stays somewhere in between or near personal and global best. If these two coincide the algorithm tends to diverge, i.e. the particle moves on searching elsewhere.

Divergence can be counteracted by V_{max} or by constriction.

Remember that we were considering an averaged version of the algorithm.

Constriction factor in canonical PSO

- Introduced by Clerc (1999/2000)

$$v_i \leftarrow K\left(\omega v_i + \alpha_1 r_1 \circ (\hat{x}_i - x_i) + \alpha_2 r_2 \circ (\hat{g} - x_i)\right)$$

- Simplest form (other definitions possible):

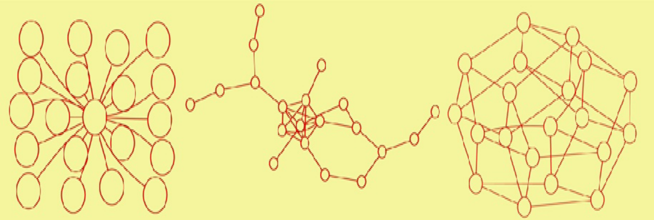
$$K = \frac{2}{\left|2-\phi-\sqrt{\phi^2-4\phi}\right|}, \text{ where } \phi = \alpha_1 + \alpha_2 > 4$$

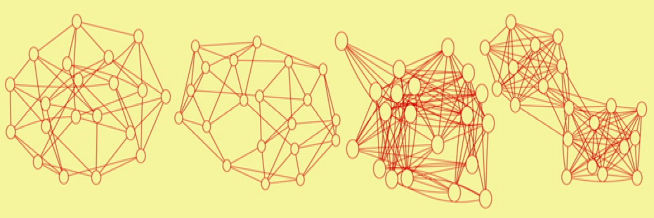
e.g. $\phi = 4.1 \Rightarrow K = 0.729$, i.e. effectively $\alpha \approx 1.5$

- By definition $K = 1$ for $\phi \leq 4$
- K may include inertia ω (i.e. set $\omega = 1$)
- Can improve convergence by a forced decay

Innovative topologies

Specified by: Mean degree, clustering, heterogeneity etc.





Topology: Restricted competition/coordination

- Topology: Restricted competition/coordination
- Topology determines with whom to compare and thus how solutions spread through the population
- g_{best} is determined only among neighbours; l_{best} as usual
- Global version is faster but might converge to local optimum for some problems.
- Local version is a somewhat slower, not easily trapped into local optimum.
- Combination: Use global version to get rough estimate. Then use local version to refine the search.
- For some topologies analogous to islands in GA

Fully Informed Particle Swarm (FIPS)

- Rui Mendes et al. (2004): "Simpler, maybe better"
- Distribute total ϕ across neighbours using weights $\mathcal{W}(k)$ which are chosen according to quality

- All neighbours contribute to the velocity adjustment
- Best neighbour is not selected, all contribute with prob. p_m
- Individual m itself is not included in its own neighbourhood $\mathcal{N}_m$

$$r_k = U\left[0, \frac{r_{\max}}{|\mathcal{N}|}\right] \quad \forall k \in \mathcal{N}$$

U equi-distr. random vector

$w \in \mathbb{R}^{|\mathcal{N}|}$ weight vector:

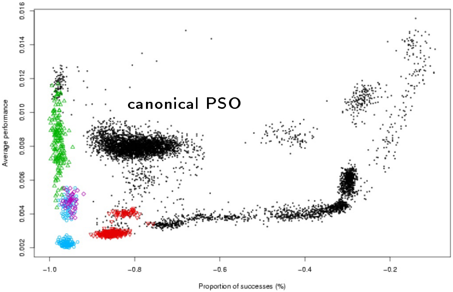
swarm follows weighted average rather than best

$$P_m = \frac{\sum_{k\in\mathcal{N}} w_k r_k \circ \hat{x}_k}{\sum_{k\in\mathcal{N}} w_k r_k}$$

- w can be constant or fitness-dependent
- Fails quite often, but results are, if successful, good (strongly dependent on good topology)

FIPS: Two performance metrics (Pareto plot)

- Cyan: Topologies with average degree in the interval (4, 4.25).
- Green: Topologies with average degree in the interval (3, 3.25) and clustering coefficient in the interval (0.1, 0.6).



- Magenta: Topologies with average degree in the interval (3, 3.25) and clustering coefficient in the interval (0.7, 0.9).
- Red: Topologies with average degree in the interval (5,6) and clustering coefficient in the interval (0.025, 0.4).
- Black: All other topologies (including canonical PSO)

$\Longrightarrow$ Topology is important if appropriately exploited.

Remarks on PSO

- Consider boundaries the counteract divergence as physical (e.g. by reflection from walls)
- Try adaptive versions: variable swarm size, variable ratios α_1/α_2
- Try different topologies (e.g. “tribes”)
- For local variants, consider using other norms in high-dimensional spaces (volume of the Euclidean unit sphere volume decays as $N \to \infty$)

Parameters, Conditions, & Tweaks

- Initialization methods
- Population size
- Population diameter
- Absolute vs. signed velocities
- Population topology
- Births, deaths, migration
- Limiting domain (X_{MAX}, V_{MAX})
- Multiobjective optimization
- "Subvector" techniques
- Comparison over problem spaces
- Hybrids

Jim Kennedy Russ Eberhart: Tutorial on Particle Swarm Optimization IEEE Swarm Intelligence Symposium 2005 Pasadena, California USA, June 8, 2005

Applications

- Evolving structure and weights of neural networks
- Complex control involving complex and continuous variables (power systems)
- Industrial mixer in combination with classical optimization
- Image analysis
- Medical diagnosis
- Job scheduling
- Robot path planning, localization
- Electrical generator
- Electrical vehicle
- Sampling of probabilistic models

Literature on swarms

- Eric Bonabeau, Marco Dorigo, Guy Theraulaz: Swarm Intelligence: From Natural to Artificial Systems (Santa Fe Institute Studies on the Sciences of Complexity) OUP USA (1999)
- J. Kennedy, and R. Eberhart, Particle swarm optimization, in Proc. of the IEEE Int. Conf. on Neural Networks, Piscataway, NJ, pp. 1942–1948, 1995.
- Y Shi, RC Eberhart (1999) Parameter selection in particle swarm optimization. Springer.
- RC Eberhart Y. Shi (2001) PSO: Developments, applications resources. IEEE.
- www.engr.iupui.edu/~eberhart/web/PSObook.html (content only)
- Advanced problems (free book) www.intechopen.com/books/show/title/particle_swarm_optimizatio
- Tutorials: www.particleswarm.info/
- Applications: http://cswww.essex.ac.uk/technical-reports/2007/tr-csm469.pdf