Particle swarm optimization

From Wikipedia, the free encyclopedia

In computer science, **particle swarm optimization** (**PSO**) is a computational method that optimizes a problem by iteratively trying to improve a candidate solution with regard to a given measure of quality. PSO optimizes a problem by having a population of candidate solutions, here dubbed particles, and moving these particles around in the search-space according to simple mathematical formulae over the particle's position and velocity. Each particle's movement is influenced by its local best known position and is also guided toward the best known positions in the search-space, which are updated as better positions are found by other particles. This is expected to move the swarm toward the best solutions.

PSO is originally attributed to Kennedy, Eberhart and Shi[1][2] and was first intended for simulating social behaviour,[3] as a stylized representation of the movement of organisms in a bird flock or fish school. The algorithm was simplified and it was observed to be performing optimization. The book by Kennedy and Eberhart[4] describes many philosophical aspects of PSO and swarm intelligence. An extensive survey of PSO applications is made by Poli.[5][6]

PSO is a metaheuristic as it makes few or no assumptions about the problem being optimized and can search very large spaces of candidate solutions. However, metaheuristics such as PSO do not guarantee an optimal solution is ever found. More specifically, PSO does not use the gradient of the problem being optimized, which means PSO does not require that the optimization problem be differentiable as is required by classic optimization methods such as gradient descent and quasi-newton methods. PSO can therefore also be used on optimization problems that are partially irregular, noisy, change over time, etc.

Contents

Algorithm

A basic variant of the PSO algorithm works by having a population (called a swarm) of candidate solutions (called

particles). These particles are moved around in the search-space according to a few simple formulae. The movements of the particles are guided by their own best known position in the search-space as well as the entire swarm's best known position. When improved positions are being discovered these will then come to guide the movements of the swarm. The process is repeated and by doing so it is hoped, but not guaranteed, that a satisfactory solution will eventually be discovered.

Formally, let $f: \mathbb{R}^n \to \mathbb{R}$ be the cost function which must be minimized. The function takes a candidate solution as argument in the form of a vector of real numbers and produces a real number as output which indicates the objective function value of the given candidate solution. The gradient of f is not known. The goal is to find a solution $\mathbf{a}$ for which $f(\mathbf{a}) \leq f(\mathbf{b})$ for all $\mathbf{b}$ in the search-space, which would mean $\mathbf{a}$ is the global minimum. Maximization can be performed by considering the function $h = -f$ instead.

Let S be the number of particles in the swarm, each having a position $\mathbf{x}_i \in \mathbb{R}^n$ in the search-space and a velocity $\mathbf{v}_i \in \mathbb{R}^n$. Let $\mathbf{p}_i$ be the best known position of particle i and let $\mathbf{g}$ be the best known position of the entire swarm. A basic PSO algorithm is then:

- For each particle $i = 1, ..., S$ do:
  - Initialize the particle's position with a uniformly distributed random vector: $\mathbf{x}_i \sim U(\mathbf{b_{lo}}, \mathbf{b_{up}})$, where $\mathbf{b_{lo}}$ and $\mathbf{b_{up}}$ are the lower and upper boundaries of the search-space.
  - Initialize the particle's best known position to its initial position: $\mathbf{p}_i \leftarrow \mathbf{x}_i$
  - If $(f(\mathbf{p}_i) < f(\mathbf{g}))$ update the swarm's best known position: $\mathbf{g} \leftarrow \mathbf{p}_i$
  - Initialize the particle's velocity: $\mathbf{v}_i \sim U(-|\mathbf{b_{up}}-\mathbf{b_{lo}}|, |\mathbf{b_{up}}-\mathbf{b_{lo}}|)$
- Until a termination criterion is met (e.g. number of iterations performed, or a solution with adequate objective function value is found), repeat:
  - For each particle $i = 1, ..., S$ do:
    - For each dimension $d = 1, ..., n$ do:
      - Pick random numbers: $r_p, r_g \sim U(0,1)$
      - Update the particle's velocity: $\mathbf{v}_{i,d} \leftarrow \omega\, \mathbf{v}_{i,d} + \varphi_p\, r_p\, (\mathbf{p}_{i,d}-\mathbf{x}_{i,d}) + \varphi_g\, r_g\, (\mathbf{g}_d-\mathbf{x}_{i,d})$
    - Update the particle's position: $\mathbf{x}_i \leftarrow \mathbf{x}_i + \mathbf{v}_i$
    - If $(f(\mathbf{x}_i) < f(\mathbf{p}_i))$ do:
      - Update the particle's best known position: $\mathbf{p}_i \leftarrow \mathbf{x}_i$
      - If $(f(\mathbf{p}_i) < f(\mathbf{g}))$ update the swarm's best known position: $\mathbf{g} \leftarrow \mathbf{p}_i$
- Now $\mathbf{g}$ holds the best found solution.

The parameters ω, φ_p, and φ_g are selected by the practitioner and control the behaviour and efficacy of the PSO method, see below.

Parameter selection

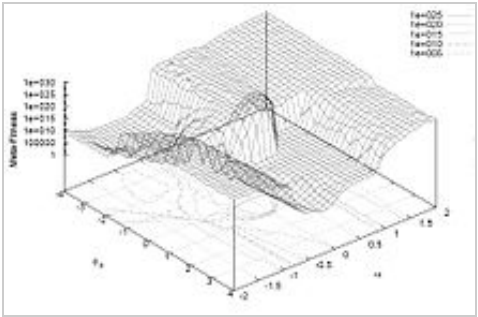
The choice of PSO parameters can have a large impact on optimization performance. Selecting PSO parameters that yield good performance has therefore been the subject of much research.[7][8][9][10][11][12][13][14] [15]

The PSO parameters can also be tuned by using another overlaying optimizer, a concept known as meta-optimization.[16][17][18] Parameters have also been tuned for various optimization scenarios.[19][15]

Neighbourhoods and Topologies

The basic PSO is easily trapped into a local minimum. This premature convergence can be avoided by not using the entire swarm's best known position **g** but just the best known position **l** of a sub-swarm "around" the particle that is moved. Such a sub-swarm can be a geometrical one - for example "the *m* nearest particles" - or, more often, a social one, i.e. a set of particles that is not depending on any distance. In such a case, the PSO variant is said to be local best (vs global best for the basic PSO).

If we suppose there is an information link between each particle and its neighbours, the set of these links builds a graph, a communication network, that is called the topology of the PSO variant. A commonly used social topology is the ring, in which each particle has just two neighbours, but there are far more.[20] The topology is not necessarily fixed, and can be adaptive (SPSO,[21] stochastic star,[22] TRIBES,[23] Cyber Swarm,[24] C-PSO[25]).



Performance landscape showing how a simple PSO variant performs in aggregate on several benchmark problems when varying two PSO parameters.

Inner workings

There are several schools of thought as to why and how the PSO algorithm can perform optimization.

A common belief amongst researchers is that the swarm behaviour varies between exploratory behaviour, that is, searching a broader region of the search-space, and exploitative behaviour, that is, a locally oriented search so as to get closer to a (possibly local) optimum. This school of thought has been prevalent since the inception of PSO.[2][3][7][11] This school of thought contends that the PSO algorithm and its parameters must be chosen so as to properly balance between exploration and exploitation to avoid premature convergence to a local optimum yet still ensure a good rate of convergence to the optimum. This belief is the precursor of many PSO variants, see below.

Another school of thought is that the behaviour of a PSO swarm is not well understood in terms of how it affects actual optimization performance, especially for higher dimensional search-spaces and optimization problems that may be discontinuous, noisy, and time-varying. This school of thought merely tries to find PSO algorithms and parameters that cause good performance regardless of how the swarm behaviour can be interpreted in relation to e.g. exploration and exploitation. Such studies have led to the simplification of the PSO algorithm, see below.

Convergence

In relation to PSO the word *convergence* typically means one of two things, although it is often not clarified which definition is meant and sometimes they are mistakenly thought to be identical.

- Convergence may refer to the swarm's best known position **g** approaching (converging to) the optimum of the problem, regardless of how the swarm behaves.
- Convergence may refer to a swarm collapse in which all particles have converged to a point in the search-space, which may or may not be the optimum.

Several attempts at mathematically analyzing PSO convergence exist in the literature.[10][11][12] These analyses have resulted in guidelines for selecting PSO parameters that are believed to cause convergence, divergence or oscillation of the swarm's particles, and the analyses have also given rise to several PSO variants. However, the

analyses were criticized by Pedersen[18] for being oversimplified as they assume the swarm has only one particle, that it does not use stochastic variables and that the points of attraction, that is, the particle's best known position **p** and the swarm's best known position **g**, remain constant throughout the optimization process. Furthermore, some analyses allow for an infinite number of optimization iterations which is not possible in reality. This means that determining convergence capabilities of different PSO algorithms and parameters therefore still depends on empirical results.

Biases

As the basic PSO works dimension by dimension, the solution point is easier found when it lies on an axis of the search space, on a diagonal, and even easier if it is right on the centre.[26][27]

A first approach to avoid this bias, and for fair comparisons, is precisely to use non-biased benchmark problems, that are shifted or rotated.[28]

Another approach is to modify the algorithm itself so that it is not any more sensitive to the system of coordinates.[29][30]

Variants

Numerous variants of even a basic PSO algorithm are possible [31]. For example, there are different ways to initialize the particles and velocities (e.g. start with zero velocities instead), how to dampen the velocity, only update $\mathbf{p}_i$ and **g** after the entire swarm has been updated, etc. Some of these choices and their possible performance impact have been discussed in the literature.[9]

New and more sophisticated PSO variants are also continually being introduced in an attempt to improve optimization performance. There are certain trends in that research; one is to make a hybrid optimization method using PSO combined with other optimizers,[32][33] another research trend is to try and alleviate premature convergence (that is, optimization stagnation) e.g. by reversing or perturbing the movement of the PSO particles,[14][34][35] another approach to deal with premature convergence is the use of multiple swarms (multi-swarm optimization), and then there are also attempts at adapting the behavioural parameters of PSO during optimization.[36]

Simplifications

Another school of thought is that PSO should be simplified as much as possible without impairing its performance; a general concept often referred to as Occam's razor. Simplifying PSO was originally suggested by Kennedy[3] and has been studied more extensively,[13][17][18][37] where it appeared that optimization performance was improved, and the parameters were easier to tune and they performed more consistently across different optimization problems.

Another argument in favour of simplifying PSO is that metaheuristics can only have their efficacy demonstrated empirically by doing computational experiments on a finite number of optimization problems. This means a metaheuristic such as PSO cannot be proven correct and this increases the risk of making errors in its description and implementation. A good example of this[38] presented a promising variant of a genetic algorithm (another popular metaheuristic) but it was later found to be defective as it was strongly biased in its optimization search

towards similar values for different dimensions in the search space, which happened to be the optimum of the benchmark problems considered. This bias was because of a programming error, and has now been fixed.[39]

Initialization of velocities may require extra inputs. A simpler variant is the accelerated particle swarm optimization (APSO),[40] which does not need to use velocity at all and can speed up the convergence in many applications. A simple demo code of APSO is available[41]

Multi-objective optimization

PSO has also been applied to multi-objective problems,[42][43] in which the objective function comparison takes pareto dominance into account when moving the PSO particles and non-dominated solutions are stored so as to approximate the pareto front.

Binary, Discrete, and Combinatorial PSO

As the PSO equations given above work on real numbers, a commonly used method to solve discrete problems is to map the discrete search space to a continuous domain, to apply a classical PSO, and then to demap the result. Such a mapping can be very simple (for example by just using rounded values) or more sophisticated.[44]

However, it can be noted that the equations of movement make use of operators that perform four actions:

- computing the difference of two positions. The result is a velocity (more precisely a displacement)
- multiplying a velocity by a numerical coefficient
- adding two velocities
- applying a velocity to a position

Usually a position and a velocity are represented by *n* real numbers, and these operators are simply -, *, +, and again +. But all these mathematical objects can be defined in a completely different way, in order to cope with binary problems (or more generally discrete ones), or even combinatorial ones [45] [46] [47] .[48] One approach is to redefine the operators based on sets.[49]

See also

- Swarm intelligence
- Multi-swarm optimization
- Bees algorithm / Artificial Bee Colony Algorithm
- Particle filter

References

1. ^ Kennedy, J.; Eberhart, R. (1995). "Particle Swarm Optimization" (http://www.engr.iupui.edu/~shi/Coference/psopap4.html) . *Proceedings of IEEE International Conference on Neural Networks*. **IV**. pp. 1942–1948. doi:10.1109/ICNN.1995.488968 (http://dx.doi.org/10.1109%2FICNN.1995.488968) . http://www.engr.iupui.edu/~shi/Coference/psopap4.html .
2. ^ ***a b*** Shi, Y.; Eberhart, R.C. (1998). "A modified particle swarm optimizer". *Proceedings of IEEE International Conference on Evolutionary Computation*. pp. 69–73.

3. ^ ***a b c*** Kennedy, J. (1997). "The particle swarm: social adaptation of knowledge". *Proceedings of IEEE International Conference on Evolutionary Computation*. pp. 303–308.
4. ^ Kennedy, J.; Eberhart, R.C. (2001). *Swarm Intelligence*. Morgan Kaufmann. ISBN 1-55860-595-9.
5. ^ Poli, R. (2007). "An analysis of publications on particle swarm optimisation applications" (http://cswww.essex.ac.uk/technical-reports/2007/tr-csm469.pdf) . *Technical Report CSM-469* (Department of Computer Science, University of Essex, UK). http://cswww.essex.ac.uk/technical-reports/2007/tr-csm469.pdf .
6. ^ Poli, R. (2008). "Analysis of the publications on the applications of particle swarm optimisation" (http://downloads.hindawi.com/archive/2008/685175.pdf) . *Journal of Artificial Evolution and Applications* **2008**: 1–10. doi:10.1155/2008/685175 (http://dx.doi.org/10.1155%2F2008%2F685175) . http://downloads.hindawi.com/archive/2008/685175.pdf .
7. ^ ***a b*** Shi, Y.; Eberhart, R.C. (1998). "Parameter selection in particle swarm optimization". *Proceedings of Evolutionary Programming VII (EP98)*. pp. 591–600.
8. ^ Eberhart, R.C.; Shi, Y. (2000). "Comparing inertia weights and constriction factors in particle swarm optimization". *Proceedings of the Congress on Evolutionary Computation*. **1**. pp. 84–88.
9. ^ ***a b*** Carlisle, A.; Dozier, G. (2001). "An Off-The-Shelf PSO". *Proceedings of the Particle Swarm Optimization Workshop*. pp. 1–6.
10. ^ ***a b*** van den Bergh, F. (2001) (PhD thesis). *An Analysis of Particle Swarm Optimizers*. University of Pretoria, Faculty of Natural and Agricultural Science.
11. ^ ***a b c*** Clerc, M.; Kennedy, J. (2002). "The particle swarm - explosion, stability, and convergence in a multidimensional complex space". *IEEE Transactions on Evolutionary Computation* **6** (1): 58–73. doi:10.1109/4235.985692 (http://dx.doi.org/10.1109%2F4235.985692) .
12. ^ ***a b*** Trelea, I.C. (2003). "The Particle Swarm Optimization Algorithm: convergence analysis and parameter selection". *Information Processing Letters* **85** (6): 317–325. doi:10.1016/S0020-0190(02)00447-7 (http://dx.doi.org/10.1016%2FS0020-0190%2802%2900447-7) .
13. ^ ***a b*** Bratton, D.; Blackwell, T. (2008). "A Simplified Recombinant PSO". *Journal of Artificial Evolution and Applications*.
14. ^ ***a b*** Evers, G. (2009) (Master's thesis). *An Automatic Regrouping Mechanism to Deal with Stagnation in Particle Swarm Optimization* (http://www.georgeevers.org/publications.htm) . The University of Texas - Pan American, Department of Electrical Engineering. http://www.georgeevers.org/publications.htm .
15. ^ ***a b*** Rocca, P.; Benedetti, M.; Donelli, M.; Franceschini, D.; Massa, A. (2009). "Evolutionary optimization as applied to inverse scattering problems". *Inverse Problems* **25**: 1-41. doi:10.1088/0266-5611/25/12/123003 (http://dx.doi.org/10.1088%2F0266-5611%2F25%2F12%2F123003) .
16. ^ Meissner, M.; Schmuker, M.; Schneider, G. (2006). "Optimized Particle Swarm Optimization (OPSO) and its application to artificial neural network training" (http://www.pubmedcentral.nih.gov/articlerender.fcgi?tool=pmcentrez&artid=1464136) . *BMC Bioinformatics* **7**: 125. doi:10.1186/1471-2105-7-125 (http://dx.doi.org/10.1186%2F1471-2105-7-125) . PMC 1464136 (//www.ncbi.nlm.nih.gov/pmc/articles/PMC1464136) . PMID 16529661 (//www.ncbi.nlm.nih.gov/pubmed/16529661) . http://www.pubmedcentral.nih.gov/articlerender.fcgi?tool=pmcentrez&artid=1464136 .
17. ^ ***a b*** Pedersen, M.E.H. (2010) (PhD thesis). *Tuning & Simplifying Heuristical Optimization* (http://www.hvass-labs.org/people/magnus/thesis/pedersen08thesis.pdf) . University of Southampton, School of Engineering Sciences, Computational Engineering and Design Group. http://www.hvass-labs.org/people/magnus/thesis/pedersen08thesis.pdf .
18. ^ ***a b c*** Pedersen, M.E.H.; Chipperfield, A.J. (2010). "Simplifying particle swarm optimization" (http://www.hvass-labs.org/people/magnus/publications/pedersen08simplifying.pdf) . *Applied Soft Computing* **10** (2): 618–628. doi:10.1016/j.asoc.2009.08.029 (http://dx.doi.org/10.1016%2Fj.asoc.2009.08.029) . http://www.hvass-labs.org/people/magnus/publications/pedersen08simplifying.pdf .
19. ^ Pedersen, M.E.H. (2010). "Good parameters for particle swarm optimization" (http://www.hvass-labs.org/people/magnus/publications/pedersen10good-pso.pdf) . *Technical Report HL1001* (Hvass Laboratories). http://www.hvass-labs.org/people/magnus/publications/pedersen10good-pso.pdf .
20. ^ Mendes, R. (2004). Population Topologies and Their Influence in Particle Swarm Performance (PhD thesis).

Universidade do Minho.
21. ^ SPSO, Particle Swarm Central (http://www.particleswarm.info)
22. ^ Miranda, V., Keko, H. and Duque, Á. J. (2008). Stochastic Star Communication Topology in Evolutionary Particle Swarms (EPSO). International Journal of Computational Intelligence Research (IJCIR), Volume 4, Number 2, pp. 105-116
23. ^ Clerc, M. (2006). Particle Swarm Optimization. ISTE (International Scientific and Technical Encyclopedia), 2006
24. ^ Yin, P., Glover, F., Laguna, M., & Zhu, J. (2011). A Complementary Cyber Swarm Algorithm. International Journal of Swarm Intelligence Research (IJSIR), 2(2), 22-41
25. ^ Elshamy, W.; Rashad, H.; Bahgat, A. (2007). "Clubs-based Particle Swarm Optimization" (http://people.cis.ksu.edu/~welshamy/pubs/ieee_sis07.pdf) . *IEEE Swarm Intelligence Symposium 2007 (SIS2007)*. Honolulu, HI. pp. 289--296. http://people.cis.ksu.edu/~welshamy/pubs/ieee_sis07.pdf .
26. ^ Monson, C. K. & Seppi, K. D. (2005). Exposing Origin-Seeking Bias in PSO GECCO'05, pp. 241-248
27. ^ Spears, W. M., Green, D. T. & Spears, D. F. (2010). Biases in Particle Swarm Optimization. International Journal of Swarm Intelligence Research, Vol. 1(2), pp. 34-57
28. ^ Suganthan, P. N., Hansen, N., Liang, J. J., Deb, K.; Chen, Y. P., Auger, A. & Tiwari, S. (2005). Problem definitions and evaluation criteria for the CEC 2005 Special Session on Real Parameter Optimization. Nanyang Technological University
29. ^ Wilke, D. N., Kok, S. & Groenwold, A. A. (2007). Comparison of linear and classical velocity update rules in particle swarm optimization: notes on scale and frame invariance. International Journal for Numerical Methods in Engineering, John Wiley & Sons, Ltd., 70, pp. 985-1008
30. ^ SPSO 2011, Particle Swarm Central (http://www.particleswarm.info)
31. ^ Çivicioglu, P. (2012). "Transforming geocentric cartesian coordinates to geodetic coordinates by using differential search algorithm". *Computers & Geosciences* **46**: 229-247. doi:10.1016/j.cageo.2011.12.011 (http://dx.doi.org/10.1016%2Fj.cageo.2011.12.011) .
32. ^ Lovbjerg, M.; Krink, T. (2002). "The LifeCycle Model: combining particle swarm optimisation, genetic algorithms and hillclimbers". *Proceedings of Parallel Problem Solving from Nature VII (PPSN)*. pp. 621–630.
33. ^ Niknam, T.; Amiri, B. (2010). "An efficient hybrid approach based on PSO, ACO and k-means for cluster analysis". *Applied Soft Computing* **10** (1): 183–197. doi:10.1016/j.asoc.2009.07.001 (http://dx.doi.org/10.1016%2Fj.asoc.2009.07.001) .
34. ^ Lovbjerg, M.; Krink, T. (2002). "Extending Particle Swarm Optimisers with Self-Organized Criticality". *Proceedings of the Fourth Congress on Evolutionary Computation (CEC)*. **2**. pp. 1588–1593.
35. ^ Xinchao, Z. (2010). "A perturbed particle swarm algorithm for numerical optimization". *Applied Soft Computing* **10** (1): 119–124. doi:10.1016/j.asoc.2009.06.010 (http://dx.doi.org/10.1016%2Fj.asoc.2009.06.010) .
36. ^ Zhan, Z-H.; Zhang, J.; Li, Y; Chung, H.S-H. (2009). "Adaptive Particle Swarm Optimization". *IEEE Transactions on Systems, Man, and Cybernetics* **39** (6): 1362–1381. doi:10.1109/TSMCB.2009.2015956 (http://dx.doi.org/10.1109%2FTSMCB.2009.2015956) .
37. ^ Yang, X.S. (2008). *Nature-Inspired Metaheuristic Algorithms*. Luniver Press. ISBN 978-1-905986-10-1.
38. ^ Tu, Z.; Lu, Y. (2004). "A robust stochastic genetic algorithm (StGA) for global numerical optimization". *IEEE Transactions on Evolutionary Computation* **8** (5): 456–470. doi:10.1109/TEVC.2004.831258 (http://dx.doi.org/10.1109%2FTEVC.2004.831258) .
39. ^ Tu, Z.; Lu, Y. (2008). "Corrections to "A Robust Stochastic Genetic Algorithm (StGA) for Global Numerical Optimization". IEEE Transactions on Evolutionary Computation ***12** (6): 781–781. doi:10.1109/TEVC.2008.926734 (http://dx.doi.org/10.1109%2FTEVC.2008.926734)* .
40. ^ X. S. Yang, S. Deb and S. Fong, Accelerated particle swarm optimization and support vector machine for business optimization and applications, NDT 2011, Springer CCIS 136, pp. 53-66 (2011).
41. ^ http://www.mathworks.com/matlabcentral/fileexchange/?term=APSO
42. ^ Parsopoulos, K.; Vrahatis, M. (2002). "Particle swarm optimization method in multiobjective problems" (http://doi.acm.org/10.1145/508791.508907) . *Proceedings of the ACM Symposium on Applied Computing (SAC)*. pp. 603–607. http://doi.acm.org/10.1145/508791.508907 .
43. ^ Coello Coello, C.; Salazar Lechuga, M. (2002). "MOPSO: A Proposal for Multiple Objective Particle Swarm Optimization" (http://portal.acm.org/citation.cfm?id=1252327) . *Congress on Evolutionary Computation*

(CEC'2002). pp. 1051--1056. http://portal.acm.org/citation.cfm?id=1252327 .
44. ^ Roy, R., Dehuri, S., & Cho, S. B. (2012). A Novel Particle Swarm Optimization Algorithm for Multi-Objective Combinatorial Optimization Problem. 'International Journal of Applied Metaheuristic Computing (IJAMC)', 2(4), 41-57
45. ^ Kennedy, J. & Eberhart, R. C. (1997). A discrete binary version of the particle swarm algorithm, Conference on Systems, Man, and Cybernetics, Piscataway, NJ: IEEE Service Center, pp. 4104-4109
46. ^ Clerc, M. (2004). Discrete Particle Swarm Optimization, illustrated by the Traveling Salesman Problem, New Optimization Techniques in Engineering, Springer, pp. 219-239
47. ^ Clerc, M. (2005). Binary Particle Swarm Optimisers: toolbox, derivations, and mathematical insights, Open Archive HAL (http://hal.archives-ouvertes.fr/hal-00122809/en/)
48. ^ Jarboui, B., Damak, N., Siarry, P., and Rebai, A.R. (2008). A combinatorial particle swarm optimization for solving multi-mode resource-constrained project scheduling problems. In Proceedings of Applied Mathematics and Computation, pp. 299-308.
49. ^ Chen, Wei-neng; Zhang, Jun (2010). "A novel set-based particle swarm optimization method for discrete optimization problem". *IEEE Transactions on Evolutionary Computation* **14** (2): 278–300.

External links

- Particle Swarm Central (http://www.particleswarm.info) is a repository for information on PSO. Several source codes are freely available.
- A brief video (http://vimeo.com/17407010) of particle swarms optimizing three benchmark functions.
- Applications (http://www.vocal.com/particle-swarm-optimization/) of PSO.

Retrieved from "http://en.wikipedia.org/w/index.php?title=Particle_swarm_optimization&oldid=508440378"
Categories: Optimization algorithms and methods | Evolutionary algorithms

---