Particle Swarm Optimization Algorithm Article summaries

William Bezuidenhout

Monday, 28 June 2010

1 Adaptive diversity in PSO

```
@inproceedings{1144006,
 author = {Monson, Christopher K. and Seppi, Kevin D.},
 title = {Adaptive diversity in PSO},
 booktitle = {GECCO '06: Proceedings of the 8th annual conference on Genetic and e
 year = {2006},
 isbn = {1-59593-186-4},
 pages = {59--66},
 location = {Seattle, Washington, USA},
 doi = {http://doi.acm.org/10.1145/1143997.1144006},
 publisher = {ACM},
 address = {New York, NY, USA},
 }
```

1.1 Introduction

Particle Swarm Optimization (PSO) is a social or evolutionary optimization algorithm that was discovered during experiments with simulated bird flocking [6]. Its discover has led to an algorithm which has gained popularity in recent years for its simplicity, relatively small number of tuning parameters, and surprising effectiveness on a large class of functions. Classical PSO begins by scattering particles in the function domain space, often by means of a uniform distribution bounded by a function-specific region of feasibility. Each particle is a data structure that maintains its current position x and its current velocity x. Additionally, each particle remembers the most fit position it has obtained in the past, denoted p for "personal best". The most fit p among all particles is written g for "global best".

A valuable variant on classical approaches is constricted PSO, where each particle updates its state using the follow- ing equations (written in a slightly non-traditional way to accentuate the role of acceleration):

$$\ddot{x}_{t+1} = \phi_1 \mathrm{U}() \otimes (\mathrm{p} - \mathrm{x}_t) + \phi_2 \mathrm{U}() \otimes (\mathrm{g} - \mathrm{x}_t) \quad (1)$$

$$\dot{x}_{t+1} = \chi(\dot{x}_t + \ddot{x}_{t+1}) \quad (2)$$

$$x_{t+1} = x_t + \dot{x}_{t+1} \quad (3)$$

where $\phi_1 = \phi_2 = 2.05, U()$ is a vector whose elements are drawn from a standard uniform distribution, and $\otimes$ represents element wise multiplication. The constriction coefficient χ is in this case defined to be

$$\chi = \frac{2\kappa}{|2 - \phi - \sqrt{\phi^2 - 4\phi}|} \quad (4)$$

where $\kappa = 1.0$ and $\phi = \phi_1 + \phi_2$.

Though effective, PSO sometimes suffers from premature convergence on problems with many local minima. Convergence is in general a desirable property, allowing the swarm to search regions near the global minimum at increasing levels of detail as time progresses. Unfortunately, in the context of many local minima, the convergence property may cause a swarm to become trapped in one of them and fail to explore more promising neighboring minima.

Designers of optimization algorithms therefore face a fundamental tradeoff: search the current local minimum in detail through quick convergence, or consume resources exploring other areas of the domain [9]. In an effort to handle this tradeoff more explicitly in PSO, some notable diversityincreasing approaches have been proposed. One such approach, the Spatial Extension PSO (SEPSO), involves endowing each particle with a radius, then causing particles to bounce off of one another [7]. A related approach, called Attractive-Repulsive PSO (ARPSO), measures the global diversity of the swarm, triggering modes of global attraction or repulsion when it crosses predefined thresholds [9]. Though effective when well-tuned, finding good functionspecific tuning parameters for these methods is non-trivial.

The tuning parameters in SEPSO and ARPSO alike represent a threshold that dictates when diversity will be artificially added to either a single particle or to the swarm as a whole, respectively. Especially in the case of SEPSO, the threshold is easier to tune and the algorithms performance improves when a simple adaptation strategy is applied.

We begin by describing SEPSO and demonstrating the issues implicit in setting its radius parameter. We then describe the proposed adaptation

methodology used to improve robustness of parameters and performance on multimodal functions. We then briefly describe ARPSO, a successor to SEPSO that is less amenable to improvements using our adaptation strategy and that rarely outperforms the easily implemented SEPSO extensions presented here.

The Spatial Extension PSO (SEPSO) is a simple method of artificially injecting diversity into a swarm. While in classical PSO, particles are conceptually volumeless and therefore never collide with one another, the basic premise of SEPSO is that particles have a spherical volume that is defined by a radius r. Two particles i and j collide when

$$\|x_i - x_j\|_2 \leq 2r \tag{5}$$

In the event of a collision, the involved particles bounce backwards, effectively moving to a point that is formed by reflecting the intended current position about the previous position, optionally reversing the velocity as well to create a post-bounce position x'_{t+1} and velocity $\dot{x}'_{t+1}$:

$$\dot{x}'_{t+1} = -\dot{x}_{t+1} \tag{6}$$

$$x'_{t+1} = x_t - (x_{t+1} - x_t) \tag{7}$$

This approach is simple to implement and has intuitive appeal: if a particle is very close to its neighbors, it is likely to be duplicating work by exploring regions that are covered by other particles and should therefore move away from them. The combination of a radius with associated notions of collisions and bouncing is an effective and intuitive way to accomplish this goal.

This method of increasing diversity is also appealing because it can be applied to nearly any variant of PSO, including those that do not have an explicit notion of velocity (e.g. Bare Bones PSO [5]): the new location is calculated according to the specified PSO algorithm, then tested against all other new locations; if a collision occurs, that location is reflected before the particles state changes. Again, velocity may optionally be reversed when present.

The choice of radius r, though not addressed in the original SEPSO work [7], is critical to the performance of the algorithm. Consider Figure 1, which illustrates the relative performance of different radius settings. The radius is set to a constant fraction of the length L of the longest diagonal of the feasible regions for Sphere and Rastrigin (Defined in Table 1). Unless otherwise stated, all figures are generated by averaging 30 runs with constricted PSO as the baseline motion, $D = 30$ dimensions, a fully-connected swarm of size 20, and velocity reversal in the event of SEPSO collisions.

The figure matches intuition. On the simple unimodal function Sphere, for which PSO is already an efficient optimizer, bouncing can only slow down desirable convergence, thereby hurting performance. As the collision radius is decreased, performance gets closer to that achieved by the baseline motion. On the highly multimodal Rastrigin, however, bouncing can be helpful, avoiding the stagnation to which PSO is generally prone for such functions. In this case, a setting of $r = .01L$ represents an improvement over baseline PSO, and the trend that is evident in the radius setting seems to indicate that a smaller radius would provide even better performance. This trend cannot continue indefinitely, however, as setting the radius to 0 simply reproduces the behavior of classical PSO. Finding a good setting for the radius is therefore a problem-dependent exercise; multiple runs may be required to obtain a useful value.

1.2 Remarks and Results

The robustness of the initial parameter settings is affected by adapting those parameters over time. In the case of SEPSO, the radius setting has a dramatic impact on the performance of the algorithm, and it is clear in Figure 1 that the parameters selected for the experiment are not low enough for Sphere but are beginning to approach appropriate values for Rastrigin. Adaptation, however, makes the choice of initial radius far less important, as illustrated in Figure 2. In each case, adapting the radius evens out the differences between the initial parameter settings, allowing them all to perform reasonably well.

In the case of multimodal functions, adapting the distance is productive because it allows a slow-moving, nearly converged particle to jump out of its current local minimum, facilitating search in other areas of the domain. Significantly, even CRIBS retains the ability to converge, but does so more slowly than CRS or baseline PSO.

Results for the benchmarks defined in Table 1 are found in Figure 4. DeJongF4, like Sphere, is smooth and unimodal. Griewank, while multimodal, begins to appear unimodal as the dimensionality increases. Ackley, SchafferF6, and Schaf- ferF7 are highly multimodal and symmetric like Rastrigin; Rosenbrock is multimodal and asymmetric but appears uni- modal when not in the region of the global minimum.

As expected, CRS and CRIBS are less effective on unimodal functions than baseline PSO. The Griewank function is interesting because it is unimodal until the proper level of detail is achieved, a fact that is evident in the slow initial drop but eventual good performance of CRIBS. With the pos-

sible exception of Rosenbrock, CRIBS works best on multimodal functions, and even on Rosenbrock it remains competitive.

Clearly, if it is known that the target function is smooth and unimodal, any kind of bouncing is a bad idea. When working with multimodal functions, however, using bounc- ing with both adaptive collision radius and bounce distance serves to improve performance while retaining reasonable convergence properties.