Particle Swarm Optimization applied to Pattern Recognition

by
Abel Mengistu

Advisor: Dr. Raheel Ahmad

CS Senior Research 2011
Manchester College
May, 2011
# Table of Contents

Introduction ................................................................................................................................. - 3 -

Objectives...................................................................................................................................... - 4 -

The Algorithm ............................................................................................................................... - 4 -
  Formal definition: .................................................................................................................. - 5 -

Initial Implementation .................................................................................................................. - 7 -

PSO and chess .............................................................................................................................. - 8 -
  Problems: ................................................................................................................................. - 8 -

Pattern recognition: Data Clustering .......................................................................................... - 9 -
  Similarity Measures ............................................................................................................. - 10 -

PSO based Clustering Algorithm ............................................................................................... - 10 -

Conclusion .................................................................................................................................... - 12 -
  Implementation .................................................................................................................... - 12 -

Parameters ................................................................................................................................... - 13 -

Problems with clustering ........................................................................................................... - 15 -

Future work ................................................................................................................................. - 16 -

Acknowledgement ....................................................................................................................... - 17 -

Bibliography .............................................................................................................................. - 18 -
Introduction

Particle Swarm Optimization (hereafter, PSO) as its name suggests is an optimization algorithm. The problem optimization algorithms try to solve can be expressed as choosing the best element from a set of available alternatives. In the simplest case, this can be seeking to minimize or maximize a given real valued function. In general however, it can be as complex as choosing a set of best available values from a variety of different objective functions with different types of domains. The first known optimization algorithms go back to the days of Carl Friedrich Gauss who introduced the “steepest descent” technique.

PSO was originally invented by (Kennedy & Eberhart, 1995) who first intended it to be for simulating social behavior. However, they noticed that it was performing optimization, and after some modifications they ended up with what is now considered to be the basic variant of PSO.

The algorithm is a metaheuristic, that is, it tries to reach its goal by iteratively improving a candidate solution with some measure of quality. With such an approach there are very little or no assumptions made about the problem; this however, means PSO does not guarantee convergence to the most optimum solution.

Also, PSO is a population-based stochastic (nondeterministic) algorithm. It systematically searches through a problem-space as the particles “flock” around their individual and swarm best positions. For the basic variation, there are user-input parameters that control the flocking behavior.

There have been hundreds of papers published within a decade that reported successful applications of PSO. A lot of these papers were cataloged by (Poli, 2007). Typical fields of application include: Antennas, Biomedical, Communication Networks, Clustering and Classification, Combinatorial Optimization, etc.
Objectives

The objective of the research project was to find atypical applications of PSO. The idea behind this was to take a different approach in applying PSO to see if it can be useful in areas where it has not been applied before. We found a few areas where PSO was not studied intensively, but ultimately in the end due to time constraints we had to resort to a typical application and implemented the PSO algorithm to data clustering. Our final work is based on the PhD thesis of (Omran, 2004) where he applies his clustering algorithm to imaging. He reports that PSO based clustering algorithm outperforms other state-of-the-art clustering algorithms in that particular application. Our objective is implementing his algorithm to cluster a set of data points into $K$ different clusters.

The Algorithm

In PSO there is a population of particles, called a swarm, that occupy a position and move around in the problem-space in a systematic way. The problem-space in many cases is a subset of the n-dimensional real coordinate space. In the beginning, every particle is initialized to randomly occupy a point in the problem-space with some initial velocity. Each particle represents a candidate solution, and in every iteration of the algorithm, we use an objective function, also called evaluation function, to evaluate the position of the particle. The evaluation function takes a vector, the position of a particle, as input and returns a real number that represents the quality of that solution. As a particle goes around the problem-space it keeps track of its best solution found so far. Also, the swarm as a whole keeps track of the best solution found among the particles. These two pieces of information are then used to dictate how the particles move.
Formal definition:

- let \( f: \mathbb{R}^n \rightarrow \mathbb{R} \) be the evaluation function

- goal: find absolute minimum (without loss of generalization)

- \( S \in \mathbb{R} \): number of particles

- \( x_i \in \mathbb{R}^n \): position of particle \( i \)

- \( v_i \in \mathbb{R}^n \): velocity of particle \( i \)

- \( p_i \in \mathbb{R}^n \): best known position of particle \( i \)

- \( g \in \mathbb{R}^n \): best known position of swarm

- For each particle \( i = 1, ..., S \) do:
  
  - initialize particle \( i \) with a uniformly distributed random vector:
    \( x_i \sim U(b_{lo}, b_{up}) \) where \( b_{lo} \) and \( b_{up} \) are the boundaries of the problem space
  
  - initialize the particles best known position, \( p_i \leftarrow x_i \)
  
  - if \( f(p_i) < f(g) \) then
    
    - \( g \leftarrow p_i \)
  
  - initialize velocity of particle \( i \),
    \( v_i \sim U(-|b_{up} - b_{lo}|, |b_{up} - b_{lo}|) \)

  - Until some criterion is met (e.g. number of iterations, adequate fitness)

  - For each particle \( i = 1, ..., S \) do:
• Pick random numbers: \( r_p, r_g \sim U(0,1) \)

• Update the particle’s velocity:
  \[ v_i \leftarrow \omega v_i + \phi_p r_p (p_i - x_i) + \phi_g r_g (g - x_i) \]

• Update the particle’s position:
  \[ x_i \leftarrow x_i + v_i \]

• If \( f(x_i) < f(p_i) \) do:
  • Update the particle’s best known position:
    \[ p_i \leftarrow x_i \]

  • If \( f(p_i) < f(g) \) then update the swarm’s best known position
    \[ g \leftarrow p_i \]

• \( g \) holds the best solution found.

The parameters \( \phi_p, \phi_g \) and \( \omega \) are user-chosen values that dictate the behavior of the swarm. In particular:

• \( \omega \) can be thought of as the inertia of the particles. The larger \( \omega \) is, it increases the “exploring” nature of the swarm.

• \( \phi_p \) as (Omran, 2004) said is parameter for the cognitive component which represents the significance of the particle’s own experience as to where the best position is.

• \( \phi_g \) also as (Omran, 2004) described is the parameter for the social component which represents the acceleration of the particle towards the swarm’s best found position
Initial Implementation

As an introduction to PSO our initial implementation uses PSO to redraw an image by solving for the brightest spot after every run. In this particular case our problem space is an array of pixels, which means that we have to adjust our equations to support a discrete problem-space. This was done by rounding a particles position to the nearest integer.

The implementation was based on the Model-View-Controller (MVC) software architecture. Generally the control flow of MVC is as follows:

- User interacts with interface
- The controller then handles the event
- The controller then notifies the model of the user action, possibly modifying the model
- A view gets its data from the model and displays the possibly changed information
- The user interface waits for further action which restarts the cycle

In our implementation however the MVC is modified a bit. The controller in this case is the a process that runs the PSO algorithm on the image for a given number of steps and puts the resulting best solution into the model which was represented by a stack, which simply is a LIFO( Last in first out) data structure. The view in the meanwhile keeps pulling “pixels” from the stack and draws them onto the interface. The following graph represents this cycle:
This application of PSO is not practical and was primarily done for aesthetic reasons and as stated earlier to become familiarized with the algorithm.

**PSO and chess**

The next step we took in applying PSO was to try to implement a chess engine using PSO. A chess engine is a piece of software/hardware that is able to play the game of chess autonomously without human guidance.

Typically a chess engine uses a quiescence search technique to skip “bad” moves and go deeper into the search tree only analyzing a few “good” moves. The quality of a move is decided by an evaluation function which would play the role of a fitness function for a PSO implementation.

**Problems:**

The major problem we faced in attempting to use PSO to search for good moves was in constructing a meaningful problem-space. In order for PSO to be successful the movement of the particles should not be completely random. To achieve that, we have to arrange all the possible moves in an n-dimensional space in such a way that the swarm’s movement will either represent region-based convergence or a piece-based convergence. In a region-based system, we have to arrange moves that originate/end at a particular square together. In a piece-based system, moves that are can be made by a particular piece would be arranged together. This way the movement of the particles around the problem-space would be somewhat meaningful. Nevertheless, speed is critical in such applications and therefore constructing the problem-space in an efficient way, we realized, was not possible within the given time frame. Finally we resorted to implementing PSO to data clustering.
Pattern recognition: Data Clustering

Clustering and supervised classification are two main branches of pattern recognition. In supervised classification, we are provided with a collection of labeled patterns; the problem then is to classify a newly encountered unclassified pattern (Jain, Murty, & Flynn, 1999). On the other hand data clustering is the classification of unlabeled patterns based on similarity. Clustering algorithms are used for many applications, which include image segmentation, vector and color image quantization, etc. The following figure shows an example clustering problem having the input on the left and desired output on the right:

![Example Clustering Problem](image)

**Figure 1.** Data clustering.

Generally there are two objectives that data clustering algorithms try to achieve. One is to minimize the distance between all patterns belonging to a cluster and the center of that cluster; another is to maximize the distance between the cluster centroids. These two different objectives will be handled by simply giving weights to each objective; going to advanced method of handling multi-objective optimization problems is beyond the scope of this research.
Similarity Measures

Similarity measures are fundamental for most clustering algorithms. The most popular method of measuring similarity is using distance measures. The Euclidean distance is the most widely used distance measure, it is defined as:

\[ d(x, y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} \]  

(1)

The Euclidean distance is a special case (when \( \alpha = 2 \)) of the Minkowski metric:

\[ d^\alpha(x, y) = \sqrt[\alpha]{\sum_{i=1}^{n} (x_i - y_i)^\alpha} \]  

(2)

PSO based Clustering Algorithm

PSO based clustering algorithm was presented by (Omran, 2004). The algorithm finds optimal positions of cluster centroids of a user specified number of clusters where each cluster groups similar patterns.

The measure of quality is described by the quantization error defined as follows:

\[ J_e = \sum_{k=1}^{K} \left[ \frac{\sum_{z_p \in C_k} d(z_p, m_k)}{n_k K} \right] \]  

(3)

Here \( C_k \) is the \( k^{th} \) cluster, \( m_k \) is the centroid of the \( k^{th} \) cluster, and \( n_k \) is the number of patterns in cluster \( K \). (Omran, 2004)
Each particle in the swarm will represent the K cluster centroids. A particle’s position in this context will be defined as a vector in a K-dimensional space. That is, \( x_i \leftarrow (m_{i1}, m_{i2}, ..., m_{ik}) \) where \( m_{ij} \) represents the \( j^{th} \) cluster centroid vector of the \( i^{th} \) particle. A swarm of particles constructed as such will represent a set of possible data clustering of the patterns. Now given a candidate data clustering, our goal is to quantify the quality of that particular clustering. Before we do that however, we will define matrices, \( Z_i \) that will hold information about the pairings between clusters and patterns. That is, \( z_{lip} \) is 1 if pattern \( p \) belongs to cluster \( j \) of particle \( i \), or 0 otherwise. Given this matrix and weights \( w_1, w_2 \) we define the fitness function as follows:

\[
f(x_i, Z_i) = w_1 \, d_{\text{max}}(Z_i, x_i) + w_2(z_{\text{max}} - d_{\text{min}}(x_i))
\]

(4)

Here \( z_{\text{max}} \) is the maximum value in the data set. Also,

\[
d_{\text{max}}(Z_i, x_i) = \max_{k=1..K} \left\{ \sum_{z_{kp} \in c_{ik}} \frac{d(z_{kp}, m_{ik})}{n_{ik}} \right\}
\]

(5)

\( d(z_{kp}, m_{ik}) \) is the Euclidean distance between a pattern and a cluster centroid. Note that we are summing over all the patterns belonging to a particular cluster. \( n_{ik} \) is the number of patterns in that cluster. Therefore, \( d_{\text{max}}(Z_i, x_i) \) is the maximum average distance of particles to their clusters, which quantifies the first objective. (Omran, 2004) For the second objective,

\[
d_{\text{min}}(x_i) = \min_{\forall i, j \in x_i} \{d(m_{ki}, m_{li})\}
\]

(6)

\( d(m_{ki}, m_{li}) \) is the Euclidean distance between to cluster centroids \( k, l \) of particle \( i \). Therefore \( d_{\text{min}}(x_i) \) will give us the minimum distance between any pair of clusters. Now the purpose of using \( z_{\text{max}} \) is more obvious since it is selected as the lowest upper bound for \( d_{\text{min}}(x_i) \).
According to our definition of $f$, a small value “suggests compact and well-separated clusters, (i.e. good clustering).” (Omran, 2004)

We will now describe the PSO Data Clustering algorithm as suggested by (Omran, 2004)

- Initialize each particle to contain K randomly selected cluster centroids
- For $t = 1$ to $t_{max}$
  - For each particle $i$:
    - For each pattern $z_p$
      - Calculate $d(z_p, m_{ik})$ for all clusters $C_{ik}$ using eq. (1)
      - Assign $z_p$ to $C_{ik}$ where
        - $d(z_p, m_{ik}) = \min_{k=1...K} \{d(z_p, m_{ik})\}$
      - Calculate the fitness, $f(x_i, Z_i)$
    - Find the personal best position for each particle and the global best for the whole swarm.
    - Update the cluster centroids using the formulae from the definition of the PSO algorithm

**Conclusion**

**Implementation**

We used processing for the implementation. Processing, according to the developers, is a programming language and development environment.

Upon writing the implementation we realized the description of the algorithm does not force all the clusters to have at least one pattern; a remedy is to discourage
clusters to be empty by making the fitness function heavily penalize empty clusters. This was done for example, by adding a constant to the fitness function whenever an empty cluster was found.

**Parameters**

After experimenting around with different variations on the parameters, we decided to allow the changing of parameter as the program is running to investigate the effects. We present input image on the left below and the corresponding output to the right, with the choice of parameters and number of iterations for convergence, \( t \). (Note, the numbers represent \( m_{ik} \))

The red colored circles correspond to the centroids for the best found clustering.

\[
w = 0.45, \phi_p = 2.00, \phi_g = 1.00, w_1 = 0.85, w_2 = 0.1, t = 17, K = 3
\]
\[ w = 0.12, \phi_p = 2.00, \phi_g = 1.00, w_1 = 0.85, w_2 = 0.1, t = 4, K = 3 \]

\[ w = 0.80, \phi_p = 2.00, \phi_g = 1.00, w_1 = 0.85, w_2 = 0.1, t = 4, K = 3 \]
Problems with clustering

In most cases, the algorithm performs well and convergences in a reasonable number of iterations. However, there are special cases where the algorithm fails. This happens when a cluster happens to be within another one. The reason why the algorithm underperforms in this case is due to the second objective which maximizes the distance between centroids and also in the way we are assigning patterns to clusters.

\[ w = 0.75, \phi_p = 2.00, \phi_g = 1.00, w_1 = 0.85, w_2 = 0.1, t > 40, K = 2 \]

Ignoring the second objective, we get similar results.
Future work

Modifying the algorithm to account for the problem mentioned above is a possible way of continuing this work.

Another possibility is to generalize the program so that the number of clusters is not a user input but rather guessed by the algorithm based on how well different values perform.

Also, an interesting idea that we considered that we were not able to incorporate to our program is using Evolutionary Algorithms to select parameters for generic images. An Evolutionary Algorithm is inspired by biological evolution and is based on the concept of “survival of the fittest.”

\[ w = 0.75, \phi_p = 2.00, \phi_g = 1.00, w_1 = 0.85, w_2 = 0, t > 40, K = 2 \]
Acknowledgement

I would like to thank Dr. Ahmad for his guidance and support without which this project would not have been successful. Also, I want to thank Dr. Lee for spending her time second-reading this paper.
Bibliography


