# Pollen-Based Bee Algorithm for Data Clustering – A Computational Model

[1]David Bradford Jr.
*[1] Center for Biometrics Research, Southern Polytechnic State University, Marietta GA, USA, dbradfor@spsu.edu*
*[2]Chih-Cheng Hung*
*[2]Professor of Computer Science at Southern Polytechnic,*
*Center for Biometrics Research, Southern Polytechnic State University, Marietta GA, USA*
*Member, IEEE, chung@spsu.edu*

## *Abstract*

*The emergent collective intelligence of social insect groups, including honey bees, presents an appealing model for problem solving. Our newly designed algorithm (which embeds bee, hive, environmental interactions, and season change) presents a more precise swarm analogy that allows our bee model to converge autonomously upon high-quality solutions. We develop in a novel way the natural concept of pollen depletion which, along with creating unique methods of field and honeycomb management, increases optimal solution exploitation – while at the same time dramatically reducing user-facing complexity. The experimental results presented, on pattern data clustering and image segmentation problems, show that the innovations of our proposed model increased both accuracy and reliability over that achieved by traditional bee models.*

**Keywords:** Data Clustering Algorithm, Image Classification, Bees Algorithm

## 1. Introduction

Bees have been cultivated and studied for millennia and their processes, by analogy, applied to many different 'fields' (for example [1]). This long history of study and bee-husbandry has provided detailed documented knowledge of their interactions, both with their hive-mates and with their environment. There are scores of publications about how they mate, how they live, how they die, how they communicate [2, 3], how they build, how they eat, and even how they store their food and raise and feed their young. Their introduction to the world of computation can be said to have begun evolving as far back as "…the very idea of taking inspiration from a bee hive model to represent knowledge for a knowledge based system can be traced back to 1986" [4].

Since that time numerous researchers have explored and expounded upon the various interactions between hive-mates and the environment (and problems they might solve) by reinterpreting those interactions computationally. There have been, to mention only a few, mating harvesting, and new nesting site search algorithms that have taken a bee swarm as their starting point for producing a functioning problem solving model (we will briefly visit some of them to provide further information in the Related Work section). Those algorithms can be used for image classification, remote sensing using robotics, damping control systems, transportation, packet routing as a form of task allocation, temperature control, data mining, and more.

In this article we will present a self-convergent algorithm we have termed Pollen-Based Bee Algorithm (PBA) which architects the bee swarm concept in a new fashion and re-engineers algorithm execution control. Initially, we'll present a brief introduction to work already existing, which has pointers in the direction of PBA, including how those 'forefathers' present their solutions using their swarm (to be contrasted with PBA). We then will introduce PBA and move through each of its important points. We offer several experiments to illustrate how PBA performs above the standard of other comparable algorithms when accomplishing that same task. The results achieved by PBA are followed by a conclusion which we use to unite the various sections and offer a broader comprehensive understanding of PBA and its place in the bee algorithm family model.

## 2. Related work

The following expresses the two current main categories, or branches, of bee-swarm algorithmic research:

> *"A number of swarm intelligence algorithms, based on the behaviour of the bees have been presented. These algorithms are divided, mainly, in two categories according to their behaviour in the nature, the foraging behaviour and the mating behaviour." [5].*

Mating behavior categories are not considered in this article and not incorporated into PBA. This is not meant to slight the algorithms that utilize bee mating behavior, but, for our algorithmic model, mating behavior is not currently relevant.

There is a third, emerging, category - that of bee swarm 'new-swarm' site selection [6, 7, 8]. This is when a swarm is ready to break away from the 'parent' swarm and fly into the environment to a newly discovered hive-home. As yet, this emerging model (which is not concerned with mating behavior nor is it intrinsically the same as foraging behavior) is not important for PBA and will not be presented here.

Many bee algorithms have been proposed and researched that are in ways influential and relevant to PBA and its modeling. Among them, we briefly review Bees Algorithm (BA), Virtual Bee Algorithm (VBA), Artificial Bee Colony Algorithm (ABC), Bee Hive Algorithm (BeeHive), Bee Swarm Optimization Algorithm (BSO), and Bee Colony Optimization Algorithm (BCO). A general pseudo-code for bee algorithms is shown below [9]. In this pseudo-code algorithm, it assumes that the problems to be solved have been formulated as the objective function with constraints. The goal of the bee algorithm is trying to find an optimal solution for the objective function defined.

**begin**
Objective function f(**x**), **x** =$(x_1,\ldots,x_n)^T$ and constraints
Encode f(**x**) into virtual nectar levels
Define dance routine (strength, direction) or protocol
**while** (criterion)
**for** loop over all n dimensions
(or nodes for routing and scheduling problems)
Generate new solutions
Evaluate the new solutions
**end for**
Communicate and update the optimal solution set
**end while**
Decode and output the best results
**end**

**Algorithm 1: Pseudo-code for Bee Algorithms**

### 2.1 Bees Algorithm

BA endeavors to codify the behavior of a swarm of bees, interacting as they might in nature, to produce a solution to an optimization problem. This is mainly based on the natural foraging behavior and broadcasting ability of honey bees to achieve the optimization task. A noted limitation of BA is its number of parameters through which a user must interact with the model; for example, the user must specify [10]:

1) *number of scout bees (n),*
2) *number of sites selected out of n visited sites (m),*
3) *number of best sites out of m selected sites (e),*
4) *number of bees recruited for best e sites (nep),*
5) *number of bees recruited for the other (m-e) selected sites (nsp), i.e. nsp = m - e,*

6) *initial size of patches (ngh) which includes site and its neighborhood*
7) *stopping criterion.*

These seven parameters can leave the user confused and wondering if they may have made a better choice than the values they ultimately decided to use (what if the next choices would have yielded even better results). This uncertainty of user entry does not lead to a confidence of use and even allows doubt as to the problem solution's quality.

## 2.2 Virtual Bee Algorithm

The Virtual Bee Algorithm (VBA) uses bees and bee interaction intensity to evaluate solution/food sources [11]. It was developed for optimizing objective functions for engineering problems such as presented in [12] for the design of damping control systems.

By taking advantage of one of the strengths of another swarm intelligence algorithm, Particle Swarm Optimization (PSO), VBA gives each bee a 'broadcasting' capability [9]. This feature allows the bees to update a 'swarm-best found location' so that each bee (as is each particle within PSO) can remain constantly aware of the swarm's best location and can modify its searching location within the landscape accordingly. This frees the bees in VBA from having to return to the hive to update and communicate locations through some means (such as the waggle dance).

## 2.3 Artificial Bee Colony Algorithm

In the Artificial Bee Colony algorithm (ABC), which was developed by Karaboga and Basturk, there are three types of bees: employed (forager), onlooker (observer) and scouts [13]. Each ABC food source will be assigned to an employed bee for determining if a better solution can be found in its neighborhood. If that employed bee abandons its search (due to lack of success), then it will become a scout who searches, randomly, for a brand new solution. Upon returning from a candidate location, any employed bees will share their information, via waggle dance, with the onlookers waiting in the dance area. An onlooker will, should condition be favorable according to that onlooker's requirements, then be recruited to become an employed bee and start exploiting the returned solution locations.

The bees of ABC must know the "number of food sources around the hive" [5]. The food sources being sought correspond to the solution of the problem and each employed bee carries that solution. ABC also has the ability to abandon a food source which fails to improve after some predetermined number of visits by the swarm's foragers. In ABC the bees are referenced as generations of bees, first, second, third, and so on.

## 2.4 Bee Hive Algorithm

The Bee Hive Algorithm (BeeHive) uses foraging behavior and waggle dances in its method. Abstracting the bees/hive into routing tables, the foragers of BeeHive address foraging regions as either short or long distance agents. These limited-lifespan agents operate to update routing nodes (best paths) for data packets [14]. This type of behavior, as already seen with VBA, is similar in nature to another swarm algorithm Ant Colony Optimization.

## 2.5 Bee Swarm Optimization Algorithm

The Bee Swarm Optimization Algorithm (BSO) is also based on foraging behaviors of honey bee swarms; however, three different types of bees including experienced foragers, onlookers and scouts are simulated in the algorithm. These bees employ different flying patterns for adjusting trajectories in the search space [15].

In detail, BSO has its bees "probabilistically adjust their trajectories in the search space for finding new food sources" [15] and are assigned to one of the three types based on an evaluation of their fitness. The scout bees inspect the landscape beginning at a large radius, r, whose diameter constricts until the bee is inspecting only a small area.

### 2.6 Bee Colony Optimization Algorithm

The Bee Colony Optimization Algorithm (BCO) uses bees, each of which incrementally builds a solution to a problem (for a user-given number of iterations). The bees perform forward and backward passes (outbound from the hive and inbound toward the hive) to build their solutions, maintain a 'solution loyalty' and attempt to expand the best most feasible solution they collectively bring back to the hive [3].

For [16] the implementation of BCO has the following structure; the bees explore and exploit the landscape in order to construct an acceptable solution (for this particular case a solution to the Traveling Salesman Problem). The bees will communicate their found solutions (if better than a previous solution) by dancing and recruiting new bees to go to the same solution location to exploit it too. The dancing bees will not stop unless a bee returns with a better food source (a better solution). If all bees stop dancing because better sources have been returned then a stratagem to reset the swarm is employed, otherwise the algorithm executes until meeting the supplied stopping criteria.

## 3. Proposed Pollen-Based Bee Algorithm

There are three core conceptualizations behind PBA. First is the construction of a bee so that a single bee is a single element of the solution set and is not comprised of the entire solution set (as opposed to "*Each bee represents a potential clustering solution as a set of k cluster centres"* [10]). Second, the storage of pollen solutions within the hive, as honey, helps define the landscape of active fields, independent of any one bee's ability, so that 'source fields' can be rated for suitability of solution. The third conceptualization is the introduction of pollen depletion to model the advancing seasons and mimic the natural changes in the landscape of pollen availability and bee industry, leading to convergence. How these conceptualizations contrast with the above related work is now presented.

Our model, PBA, is descended from BA, but, rather than allow the user to assign a large number of parameters to control the execution of the algorithm the user has only two variables to assign. These two variables work behind the scenes to automate the assignment of all other variables used in PBA. In BA the bees are paramount in their carrying of the solution set and communicating their solution set to other bees – not so in PBA, where the bees are individual elements without any understanding of the solution set required as a whole. This division of the solution as a whole from an individual of the swarm led to the development of the 'field' and of 'field solution storage' at the hive, in the form of honey which then allows the solution sets to improve by the individual work of the bees and not by the efforts of any single bee alone.

In our PBA model we also take advantage of PSO (as does VBA) and thereby consider one bee as one particle of the swarm. PBA does not, however, communicate a global best solution amongst its bees and instead allows the quality of the pollen gathered from the fields to be rated at the hive in the form of honey in the honeycomb.

PBA does have two types of bees, but, unlike ABC, these bees do not change roles nor are they treated like 'gererations' where one set of bees is replaced by another later generation. Another similarity between ABC and PBA is the abandoment of food souces (or fields when speaking of PBA) so that our model will abandon a field only if another better field is discovered by the scouts.

What BeeHive shares in common with PBA is the treatment of the bees as two types, long and short distance agents, in which the agents can be interpreted as individual particles like PSO, and thus like PBA's bees. The long distance agents would be more comparable to the scout bees in PBA (who explore the landscape to return undiscovered pollen in a random fashion) and the short distance agents as comparable to the forager bees (who return to a given field and pollen location in order to exploit the area around an already given and rated pollen source).

Contrasting PBA with BSO, by using the pollen depletion method, PBA allows the scouts to search the entire landscape without constriction to a certain radius. Also, the foragers are encouraged to begin their search for better solutions in reverse of that described by BSO, starting in a small area and, given enough time (if it is early enough in the harvesting season and there are multiple foragers employed) they are dispersed further and further afield from that solution which allows easy escape from local optima.

BCO is most similar to Ant Colony Optimization [17] in its handling of its members. PBA, unlike BCO,

does not use a reset strategy as the bees are not able to influence each other in their tasks – the scouts will explore and the foragers will exploit without regard to any particular member finding a better or worse pollen location. The individuals in PBA do their individual best to perform up to the task assigned and so do not change their performance level nor actions based on other individuals of the hive – this allows the swarm as a whole to survive the 'missteps' that single individuals can make as they perform their duties.

The PBA algorithm pseudo-code is presented below. Full detail will be given in the later sections for pollen depletion, scouts, fields, landcape, foragers, environment, and interactions.

1. Initialize variables
2. do while pollen not depleted and scout-able fields exist
    a. start scout bees loop
        i. explore landscape (find pollen)
        ii. create fields (ranked) as found by scout bees
        iii. evaluate/replace stored fields with any better found fields
    b. end scout bees loop
    c. apply pollen depletion (modifying numbers of bees and viable fields)
    d. start forager bees loop
        i. forager bees are assigned a stored field to exploit
        ii. explore/exploit within the recruited field
        iii. evaluate/replace stored pollen source with better returned pollen sources
    e. end forager bees loop
    f. repeat at 2.
3. best field's value (having the best honey in the honeycomb) is solution.

**Algorithm 2: PBA**

## 3.1 Pollen depletion

Pollen depletion is an insight that can be said to have grown from the business practice of bee owners moving their bees and bee hives to different locations as a blooming season progresses. What this implies (the movement of bees and hives) is that fields do not stay in bloom forever and that a field's pollen changes. More important is the realization that the "number of fields" in bloom, and even the blooms within a single field, does not remain the same. This realization lets PBA, through calculation, modify the number of fields its scouts return, as viable, while it runs. This pollen usage is different from Flower Pollination by Artificial Bees (FPAB) which uses bees and pollen such that "*the bees will pick up the pollen of the flower with lowest growth and pollinate the pollen where it will grow better*" in order to "*select the flower with best growth of one species to survive*" [18] which is, in essence, a pollen accretion method.

The 'pollen depletion percent' entered by the user is our method of implementing the idea of pollen depletion in a landscape. Given that a landscape can have, at its beginning, 100% pollen, the user supplies a pollen depletion percent value which indicates how much pollen is removed from the landscape by the bee swarm each time they venture into that landscape. When the pollen remaining in the landscape is so low as to make excursion by the bee swarm to harvest that pollen counterproductive (conceptually speaking, the bees expenditure of effort to harvest fewer solutions being greater than the quality of the return on that effort) then the pollen season is over and the bee swarm (the algorithm) has converged finally upon a single best field in the landscape. What this means is the pollen depletion rate must initially be high enough (greater than 2%) to entice the bees into the landscape in the first place.

## 3.2 Scouts and landscape

The user supplied number of clusters, C, (with the understanding that the number-of-clusters is a way to state 'the number of variables that create a single solution-set') creates the initial number of investigative (scout) bees for the swarm, all within the bee hive:

$$\text{Scouts } S = \text{int}(C/2)*D \quad (1)$$

Here, as elsewhere in this article, int denotes the integral part of the value. This automation of the creation of the number of scouts, S, derived using the number of clusters, C, and the pollen depletion rate, D (given as a numerator), was designed to create the presumed optimal number of scouts necessary to explore the landscape effectively. Division by 2 in equation (1) is chosen to 'divide' the effort between the two types of bees (i.e. scouts and foragers). No additional scouts are created nor are any scouts destroyed once they are created. Using the concept as expressed in particle swarm optimization, and in contrast to the bee models presented in the related work section of this article, each bee becomes same as a single particle. The other type of bee PBA uses (foragers) is discussed in a later section, but, in general, the foragers work to refine the solutions of the scouts.

Another thought behind pollen depletion is the idea that a hive, given that it is comprised of a number of bees (as calculated in equation 1), knows with its finite number of bees there is only a limited amount, or measure, of landscape that they as a hive can harvest. Such an idea leads to a psychological supposition, made about the thinking a bee may have about its pollen harvesting, and can be stated thusly; actual bees do not want to waste their limited seasonal time on less than plentiful fields of pollen – however, first they must discover where those plentiful fields are located in their landscape. So too does PBA need to discover the landscape of fields from which to harvest solutions. Our proposed algorithm calculates an estimate of the maximum number of fields in the landscape that the swarm as a whole has the resources (the number of scout bees) to explore in the given season. The greater the number of scouts means the landscape created for exploration can be larger. The expectation is that the number of scouts is always greater than the number of clusters or that the fields in the landscape, L, always greater than 1 (i.e. int(S/C) > 1).

$$\text{Fields in the Landscape } L = \text{int}(S/C) \quad (2)$$

This landscape equation is carefully derived to keep from creating an arena of exploration so small that it leads to algorithmic churn or an arena of exploration that is too vast to be logically and effectively explored and exploited in a reasonable amount of time. PBA takes that landscape and specifies within it 'fields of interest' based on the scouts exploration.

The landscape is, through analogy once the scouts return, represented within the hive by the swarm's honeycomb. The number of fields within that external landscape is represented by the number of hexagonal cells that make-up the honeycomb. Each solution-set value (i.e. the best found pollen source of a given field (and one variable of a multivariable solution-set)) is identifiable as the stored pollen within each individual cell of the hive's honeycomb (see Figure 1).

This honeycomb thus contains the mixture of pollens brought back by the scouts (or improved upon by the foragers, as will be presented) and as a whole is referenced as the "honey" of the hive. In a similar programmatic sense, the solutions bought back by the bees are only 'parts' of the whole solution desired, so each of the bees returned data can be manipulated if needed (by calculations) to determine if its individual worth is of value to the hive. In this way the scouts explore a wide landscape, return to the hive where they are sorted, and deliver their pollen in 'field-sized' bundles.

We should mention that the concept of Landscape is one that is difficult to put into words when trying to quantify, via a calculation, that analogy. The landscape is the available data from which all pollen is harvested (from which all solutions originate) without necessarily being at any one point a specific answer. This necessitates the bees, when retrieving this pollen, to calculate whether their found pollen is applicable and to also calculate/analyze at the honeycomb whether a returned pollen improves the quality of the honey in an existing honeycomb cell. This "data of the landscape" will be referred to as Originating Data (OD) and will be that data which is the 'ground' necessary from which solutions can be calculated and conceived. This also requires us to conceive of the scouts' returned pollen differently, as it can be calculated to be something other than OD. Therefore, a scout who goes into the landscape and returns with pollen (i.e., a solution as based upon OD) will have "Found Pollen" (FP).

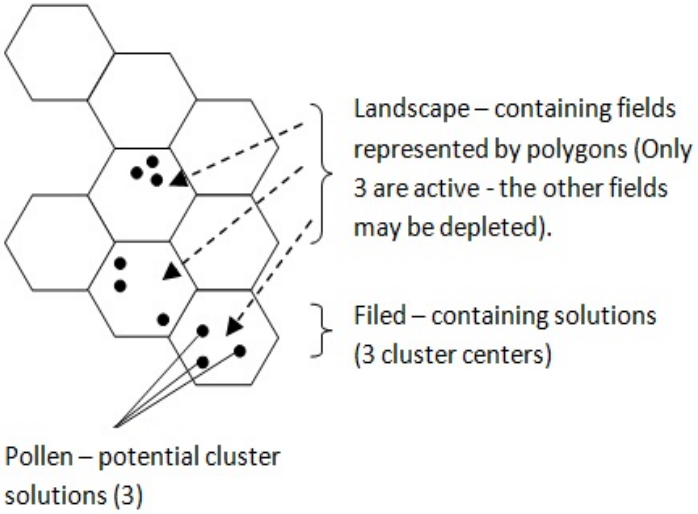


**Figure 1.** Honeycomb in the Hive. Programmatically speaking, the honeycomb is equivalent to a multidimensional array where one array element is the cell and the other is the bee's pollen solutions.

### 3.3 Scouts and fields

As an artifact of bee behavior, the waggle dance is well explained in the references that detail current Bee algorithmic interpretations [5]. Though it is presumed that a bee hive's actual dance floor can have multiple live bees performing their waggle dance at any one time, PBA controls its scout bees in two ways. First, the hive's scout bees (once all have returned to the hive) must form a line as they approach the dance floor, sorting themselves by their found pollen quality, in descending order of fitness – the best scouts dance first. The second requirement of PBA is that the waggle dance floor must be occupied by exactly the number of scout bees equal to the multiple of user-supplied number of clusters. If that exact number of scout bees cannot be met then the waggle dance floor remains empty.

For example, if there are ten scouts and three user-specified 'clusters' (i.e., three variables that make up one single solution-set) then, sequentially, three groups of three scouts will be able to dance (thus describing three fields a.k.a. building three cells of the honeycomb, within each of which the 'FP' (the variables of a single solution-set) the bees represent were found) whilst the last, tenth, scout bee will not be allowed on the waggle dance floor.

One of the benefits of sorting the scouts (contained within the pseudo code above, at step 2 a. ii) from best to worst solution fitness is the fact that not all scouts will find a solution every time. Those scouts who return from the landscape without FP to report are naturally excluded from the waggle dance floor as they have no pollen location to waggle about to the swarm. Only those scouts returning with a solution can be considered "True Scouts" (TS). If returning scouts have found a better field then a new cell in the honeycomb is created for this new, better, field and a lesser quality cell abandoned.

The pollen depletion method of PBA, over its course of execution (iterations) reduces the amount of pollen in the field. Let V equal pollen in the field (starting as would be standard, at 100%) and D equal the user supplied pollen depletion rate percentage. The following carefully considered equation is the driving force behind the automation of the convergence of the algorithm and is responsible for the changes made, as the algorithm iterates, to the number of scouts (which, as will be explained, affects all other automated calculations in PBA).

$$\text{Pollen in the field} \quad V = V - D, \text{ if } V < 0, \; V = 0 \tag{3}$$

Therefore, after pollen in the field has been reduced, the season has moved forward and there is less pollen in the landscape to harvest so it is time to recalculate the number of scout bees (which then, through calculation, changes the size of the scouts perceived landscape, purposefully designed to advance the season and manage convergence),

$$\text{Recalculate Scouts } RS = \text{int}(\min(TS, S)*V/100) \quad (4)$$

where min denotes minimum function, scouts S, as defined earlier, and "TS" = successful scouts. Different from forager bees (which will be presented momentarily), the scout bees do not have any preconceived idea what the landscape holds in terms of pollen. They explore the landscape precisely because they are searching for new sources of pollen for the swarm.

As the number of scout bees diminishes, and the number of scout-perceived viable fields in the landscape also diminishes, the hives' scout bees begin to make multiple excursions (modeled in equation (19)) from the hive into the landscape before the forager bees are allowed out. This allows a chance for successful escape from suboptimal solutions or local optima even as the algorithm automatically progresses towards convergence and an ultimate solution. It is the scouts who determine the harvesting season – once they indicate there is not enough pollen in the field/landscape, when the landscapes' pollen has been depleted (and thus when too few scouts are calculated to waggle about a field to the foragers) then the algorithm has converged.

## 3.4 Foragers

There are three types of forager bees created in our model (compared to just one type of scout bee) – elite, regular, and random. These three types are inspired directly by the bees algorithm's types of forager bees [10]. These equations take into consideration the number of scouts (S) and the amount of pollen (V), purposefully optimized through iterative experimentation, and whose calculation has been automated, to create a presumed number of foragers that would be appropriate for best harvesting and exploiting the pollen discovered and brought back by the scouts which has been grouped into cells in the honeycomb of the hive. The forager counts are calculated as:

$$\text{Elite Foragers: } E = \max(1, \text{int}(S/2)) \quad (5)$$

$$\text{Regular Foragers: } R = \max(1, \text{int}((V/2)/100)*E)) \quad (6)$$

$$\text{Random Foragers: } G = C - (X+Y) \quad (7)$$

Here 'max' is the maximum evaluation, S, V and C as defined before, and X and Y as defined below. Please note that 2 refers to two types of bees (scouts and foragers).

It is these forager bees who watch the waggle dancing of the scout bees on the dance floor (and who are thus 'recruited' by a particular scout bee to fly to that scout bees' indicated found pollen source). The foragers, once created, do not increase nor are they decreased or killed, but, like the scouts, as the algorithm iterates fewer and fewer foragers are 'employed' in the harvesting of pollen solutions.

Because scout bees indirectly recruit foragers to harvest pollen (to improve a found field/solution) the 'ranking' of a scout bee on the waggle dance floor is important regarding the number, and type, of forager bees any one particular scout can recruit. As a scout bee enters the waggle dance floor (and the scout bees are already sorted by best to worst fitness, descending) it is ranked based on the quality of the found pollen/center it has brought back to the hive. There are three ranks – elite (having high quality pollen), regular, and random; this is purposeful and corresponds to the types of forager bees. There are only a certain number of scout bees allowed, as they enter the dance floor, to be ranked elite, or regular, or random as defined in equations (8), (9), and (10). The analogy is that the scouts entering the dance floor represent the landscape sites from which they came and so the forager bees are, by watching the scouts, determining which sites in the landscape to return to and exploit. This 'number of sites of interest to a type of forager' is a fixed value assigned at the start of the algorithm based upon the user supplied number of clusters and this number is automatically calculated and assigned by the algorithm as:

$$\text{Elite Pollen Sites Inspection: } X = \text{int}(C/3) \quad (8)$$

$$\text{Regular Pollen Sites Inspection: } Y = \text{int}(C/3)+(1 \text{ if } C/3 \text{ has a remainder, } 0 \text{ if } C/3 \text{ has no remainder}) \quad (9)$$

Random Pollen Sites Inspection: Z = 1 (10)

Note: 3 corresponds to 3 forager types used in equations (8) and (9)) and C is the minimum number of clusters.

These equations were derived to create slightly fewer elite pollen sites within a described/waggled field (since the supposition is that those best sites are harder to find) and slightly more regular pollen sites within a field (since the supposition is that those sites will be generally easy to find) and random sites (which are presumed to be the worst of the sites so no forager is truly interested in these but still they must be looked into in the case they may be near an excellent site). The elite forager bees are only recruited/attracted to inspect scout bees that, upon entry to the waggle dance floor, are ranked as having found elite pollen sources, regular foragers to those scouts ranked regular and so forth (and addresses the same question from [19] about how many foragers will forage within a field).

The calculations in equations (5), (6), and (7) determine how many foragers will be interested in a particular scout's indicated pollen location. If there are, for example, 8 clusters given by the user and the number of scouts calculated is 12 then the number of elite foragers created for each elite-ranked scout on the dance floor would be "max(1, int(12/2))", or 6; therefore 6 elite foragers would be recruited to exploit each of the elite-ranked scouts' pollen source. The number of scouts ranked as elite, i.e., the number of sites deserving elite forager inspection, when there are 8 clusters, would be "int(8/3)", or 2. Therefore, in this example, there would be 2 sites ranked as elite (2 scouts on the dance floor ranked elite) that the elite foragers would be interested in exploiting, and, there would be 6 elite foragers interested in each of the 2 scouts/sites. This same type of interpretation is applied to the remaining scouts and foragers (i.e., elite, regular and random) using their respective equations.

The designated number of elite forager bees that will pay attention to the elite ranked scout(s) (as discussed above) and fly to that pollen source, due to the pollen depletion method, will change as the algorithm iterates (and is different from *"where the number of bees sent to a site is fixed*" [20]). This change is due to the conceptualization that as the field is harvested over and over again (and the season advances) the chance of finding a better and better pollen source within that one field decreases and so the assignment of resources (forager bees) is decreased to match that expectation. However, the forager bees are tasked to improve discovered fields and therefore do not limit their improvement-harvesting to only the limited number of scout-indicated fields – the foragers exploit the total number of fields known to the hive (and that the hive has calculated it has the number of bees to successfully harvest from and that are active in the hives honeycomb) even as their numbers decrease due to pollen depletion.

### 3.5 Foragers and environment

Forager bees have two calculations that influence their reaching of any "found solution" as advertized by a scout bee on the hive's waggle dance floor through their dance. One involves a deviation calculation and the other a calculation which tries to quantify the impact of the environment (or the ability of a given forager bee) to effectively harvest a solution.

In the first calculation, a forager bee has been recruited to visit a 'location in a field' (an FP) as indicated by its observed scout bee (it might alternatively be said that a scout bee who, ranked as having found an elite pollen location upon entry to the waggle dance floor, draws the attention of an allowed number of elite foragers who will fly, one after the other, to that scout's indicated field location). We make a supposition into the psychology of a forager bee when bringing the environment into consideration for PBA – it would be foolish of the forager bee to go to any location in the field being waggled about by the dancing scout bee that it wishes, so, the first forager bee that returns to its scout bees' indicated found pollen solution should arrive fairly close (with limited deviation) to where the scout bee specified and attempt to improve the pollen harvest at that point:

Forager Deviation: Q = .95 $(0 < Q <= 1)$ (11)

This is in direct contrast to [21] which uses a search radius starting at a maximum value and ending at a minimum value and is also in contrast to [15] which uses a 'neighborhood shrinking' method where

"*the local search is initially defined over a large neighborhood*" [15] and restricts. This Q value is one of our unique contributions to the bees' model, where, in our current implementation, the Q value remains constant.

When a forager bee returns to the hive and updates the cell in the honeycomb with its newly found solution (but only if its exploited/refined solution is 'fitter' than the solution before it – i.e. only if the returned foragers contribution to the pollen in the honeycomb cell improves that field/cell as a whole) then subsequent foragers of the same rank, already recruited for that scouts' solution, are allowed to venture even further from the found pollen solution indicated in the honeycomb's cell. This thus increases the exploitation range, allowed a forager, around any best solution yet recorded, in its own attempt to find a fitter 'nearby' quality of pollen – this concept is shown in Figure 2 (based upon the following equations).

The key concept here is the expectation that good solutions will clump together but that multiple better solutions may be located nearby (if only a forager would 'hop over a little bit' to discover it)). Hence, forager exploration, W, is modeled using these first two calculations as:

Forager Exploration: W = random(f)*Q*(1 or -1 (chosen randomly)) (12)

If W = 0 then W = Q-(f/100)*(1 or -1, chosen randomly) (13)

(where random is a random number generator, f = the iteration number of that forager, E or R or G, iterating 1 to E or 1 to R or 1 to G)

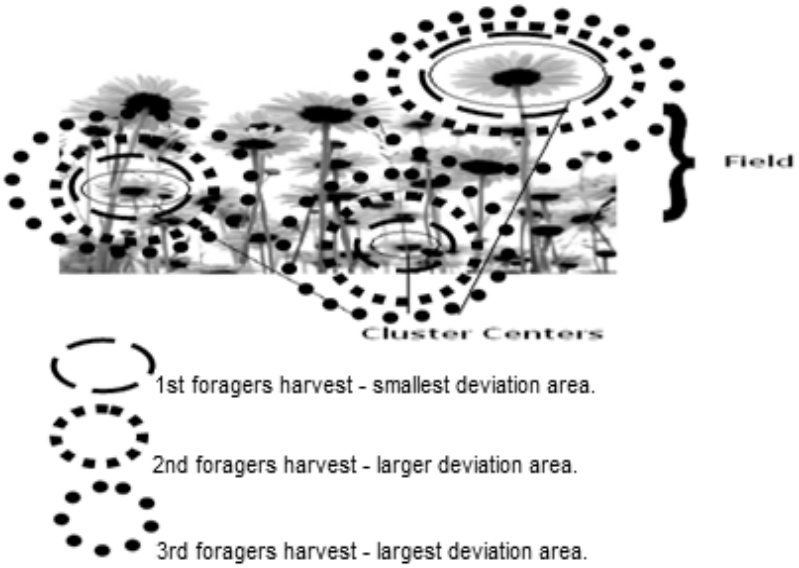


**Figure 2.** Forager exploration illustrated using three elite forager bees each forager going to three elite found pollen locations in a field (as indicated by a scout).

PBA attempts, uniquely among bee models, to quantify the interactions of the environment and a bee, both of which are not static entities. There are fires that destroy a pollen source, strong winds that make it impossibly difficult to reach a pollen source, bee insanity or damage that keep the bee physically from reaching a pollen source. Certainly, elite forager bees have much less of a chance to go astray while regular foragers have a better chance to miss their pollen-mark and, as could be obvious, random foragers are not especially successful. These hazards of the environment and life are per forager calculated (remembering that OD is 'Originating Data' as discussed earlier) as:

Forager Exploration:
(hazard recalculation may change W value from equations (12) and (13))

a. for (E) If a random decimal number <= 1-Q then W = random(OD) (14)

b. for (R) If a random decimal number <= Q*(D/100) then W= random(OD) (15)

c. for (G) If a random decimal number < Q then W = W + random(OD) (16)
where Q and D same as defined before.

The equations are a direct result of iterative experimentation using the unique PBA calculation hazard to optimize when a forager might be allowed to 'stray' unexpectedly from a solution and thus escape localized optima. The forager found pollen center is calculated (using the before mentioned methodology at the end of section III(B)):

Forager Found Pollen = W + scout solution (FP) that attracted the forager (17)
Note: Using FP as discussed in section III(C) .

This forager found pollen is then rated for fitness with the pollen already in the honeycomb cell. If the foragers' found pollen solution serves to improve the fitness of the honey in the honeycomb then it is added to the honeycomb and subsequent scouts use this found pollen as their initial starting point for exploration calculation; if it does not improve the honeycomb cell as a whole then that foragers' found pollen is discarded (and the forager returns to the field or to the dance floor to look for another field to go to).

### 3.6 Pollen depletion applied

The repeated coming and going of bees serves to deplete the pollen in the landscape of fields, as shown in equation (3). It is this depletion that affects the number of scouts which then recalculates the number of fields in the landscape that the hive's scouts are to inspect. This recalculation of fields in the landscape (which are allowed to be waggled about) is an artifact of PBA design purposefully created to force convergence of the swarm and the idea that the harvesting season is over:

Fields in the Landscape L = int(RS/C) (18)

This concept (where RS is from equation (4) and C is the number of clusters) is unique to PBA and is different from other bee swarm methods which *do* abandon a poor scout indicated field but *replaces* it and therefore the full number of fields in a landscape must always be explored. PBA recalculates the number of fields in the landscape, always decreasing, and assigns scouts to only search within that limitation.

Due to the recalculation of the pollen in the field and thus the number of fields in the landscape, the number of times the scouts venture into the landscape, before the foragers go out, increases (as discussed earlier). The calculation of the number of times the scouts inspect the landscape utilizes the results of equations (3), (4), and (18) thusly (and allows, as the algorithm converges due to fewer fields, the scouts an increased opportunity to discover, in the landscape, a yet discovered but superior field of solution):

Scout repeated excursions:
B = Initial L minus current L
(i.e., equation (2) above minus the result of equation (18)) (19)

The number of foragers employed are recalculated at every iteration, at the time of pollen depletion. The recalculation of the foragers does not change the equations that determine their values and so equations (5), (6), (7) are repeated using the recalculated values of equations (3) and (4) (substituting RS, being the number of successful scouts (as discussed and calculated earlier), for S, the number of scouts currently active, where applicable).

The pollen depletion method allows for the eventuality that there will be, at some point in time, a 'winter' and thus the scout bees will not find it a wise use of resources to continue searching for fields of pollen. The expectation is that only in singularly surprising circumstances will a better field be discovered at the end of the season, the foragers having exploited all the best fields yet indicated. After this point in time (after pollen depletion is applied and no field can be waggled about by the few

remaining calculated scouts) the PBA model is deemed to have converged and the best field/cell of the honeycomb contains the best solution(s) possible and thus the algorithm ends.

In summary, the algorithm used to implement the proposed PBA model is listed in Tables 1 and 2 for a comparison with a traditional BA algorithm [10]. The algorithm is listed in Appendix in detail.

**Table 1.** Parameters and Algorithm for PBA Composed in Table Form

| **Parameters** | | **Algorithm** | **BA parameters [10]** |
|---|---|---|---|
| Number of Clusters | C | User supplied number value | Each 'n' is comprised of C |
| Pollen Depletion Rate | D | User supplied, 2 < D < 100 | - |
| Pollen in the field | V | 100% (= 100 percent integer value for amount of pollen in the landscape) | - |
| Scouts | S | int(C/2)*D | n |
| Landscape (fields within) | L | int(S/C) | ngh (initially, then 'm' as pollen depletion is applied) |
| Elite Foragers | E | max(1,int(S/2)) | nep |
| Elite Pollen Sites Inspection | X | int(C/3) (use of 3 corresponds to the three forager types) | e |
| Regular Foragers | R | max(1,int((V/2)/100)*E)) | m-e |
| Regular Pollen Sites Inspection | Y | int(C/3)+(1 if C/3 has a remainder, 0 if C/3 has no remainder) | nsp |
| Random Foragers | G | C-(X+Y) | - |
| Random Pollen Sites Inspection | Z | 1 | - |
| Forager Deviation | Q | .95 (0 < Q <= 1) | - |
| Forager Exploration and Hazard recalculation | W | 1. W = (a random number 1 less than f)*Q*(1 or -1 (chosen randomly in order to overshoot or undershoot the next solution exploitation))<br>2. W = 0 then W = Q-(f/100)*(1 or -1 (chosen randomly in order to overshoot or undershoot the next solution exploitation)) (f= E or R or G, iterating through 1 to E or 1 to R or 1 to G)<br><br>a. for (E) If a random decimal number <=1-Q then W = random(OD)<br>b. for (R) If a random decimal number <=Q*(D/100) then W = random(OD)<br>c. for (G) If a random decimal number <Q then W = W + random(OD)<br><br>3. Forager Found Pollen = scout solution (FP) that attracted the forager + W (Note: Using FP as discussed in section III(C)) | - |

**Table 2.** Recalculation of Parameters Using PBA Composed into Table Form

| Pollen Depletion Recalculations per Iteration | Algorithm |
|---|---|
| Pollen in the field | V = V - D; If V<0, V=0 |
| Recalculated Scouts | RS = int(min(TS, S)*V/100) (TS = successful scouts, meaning not all returning scouts will have successfully found pollen, this 'TS' value recognizes/counts only returning scouts having found pollen) |
| Landscape (fields within) | L = int(RS/C) |
| Scout repeated excursions | B = Initial L minus Current L |
| Elite Foragers | E = max(1,int(RS/2)) |
| Regular Foragers | R = max(1,int((V/2)/100)*E)) |
| Random Foragers | G = C-(X+Y) |
| Scouts (once all foragers have returned) | S = RS |

## 4. Experimental results

The problem presented in this article, as a way to illustrate the achievements of PBA, will focus on the applications of image segmentation and pattern classification. This illustration is not meant to limit PBA to the single arena of algorithms which 'cluster' so much as to present an easily understood and conveyed analogy whereby our swarm intelligent system effectively approaches its given set of parameters and produces emergent high quality results without undue user interaction or control. This also means that the pixel data (each pixel being comprised of various values, i.e., each pixel is a vector of feature components) is the OD (the Originating Data mentioned previously).

The segmenting of an image implies the need, for any algorithm so tasked, to know how many partitions (i.e. how many clusters) are allowed. This means that every pixel of a given image (regardless of a pixel's individual component values) must be assigned as belonging to one of the allowed clusters, C. In PBA this number of clusters is a user defined entry (along with the pollen depletion percent, making up the two user supplied values) which resides in the hive independent of any bee. This allows for flexibility of use. This parameter, number of clusters, is very similar to some clustering algorithms such as K-means clustering algorithm [22].

The bees will be tasked with searching the problem landscape (a given image in our example) and will determine to which of the clusters a pixel should be assigned. As shown in Figure 1, if the user-supplied desired number of clusters is three, within each honeycomb cell (i.e. each found field in the landscape) there would be three pollen points (cluster centers). In this manner PBA maps each pixel, through a distance measurement such as Euclidean Distance, to one and only one of a given number of desired centers.

The following images (Figure 3A-F) illustrate the use of pollen depletion percentages in PBA and its effect on reaching a solution to segmenting the given image.

The results from Figure 3 show that the bee swarm finds this particular image's visually optimal solution when the pollen depletion percentage is 5%. A lesser pollen depletion percentage yields a less visually optimal clustering while a greater pollen depletion percentage (while not degrading the resultant clustering) will execute many more iterations than are necessary to produce the visually optimal solution.

For another example of image segmentation we compare PBA to an implementation of BA [10] using a couple of real images (see Figure 4 A-F). The results illustrate the improvements attained by our new model. The BA parameters used in our experiments are based on the paper given in [10].

These following images (Figures 5 A-F and 6 A-E) present PBA alongside other swarm intelligent algorithm solutions [23], as well as a couple of algorithms designed specifically for image segmentation. Here we compare with the K-Means, Fuzzy C-Means, Ant Colony Optimization, and Particle Swarm Optimization algorithms [23].

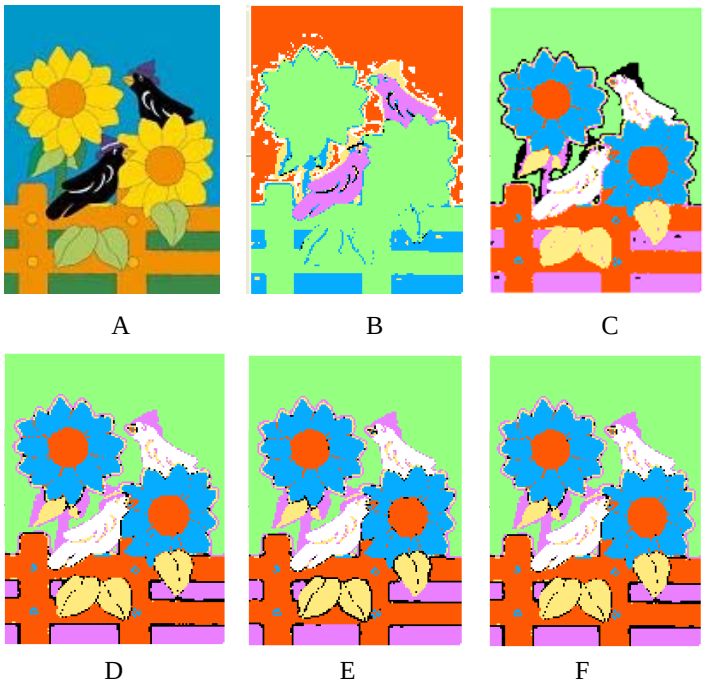


**Figure 3.** A) Original image, seven clusters desired, B) PBA with 3% pollen depletion, C) PBA with 4% pollen depletion, D) PBA with 5% pollen depletion – visually optimal, E) PBA with 10% pollen depletion – little variation over D), and F) PBA with 20% pollen depletion – little variation over D)

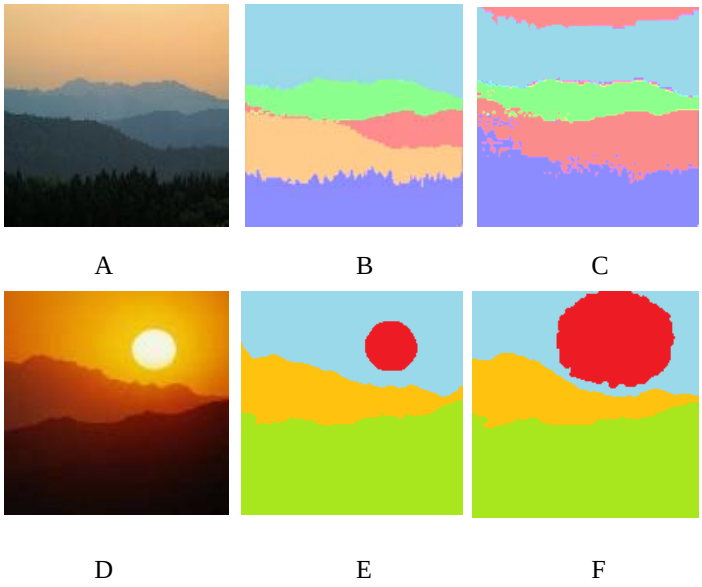


**Figure 4.** A) Original image, five clusters desired, B) PBA with 10% pollen depletion, C) BA, D) Original image, four clusters desired, E) PBA with 10% pollen depletion, and F) BA

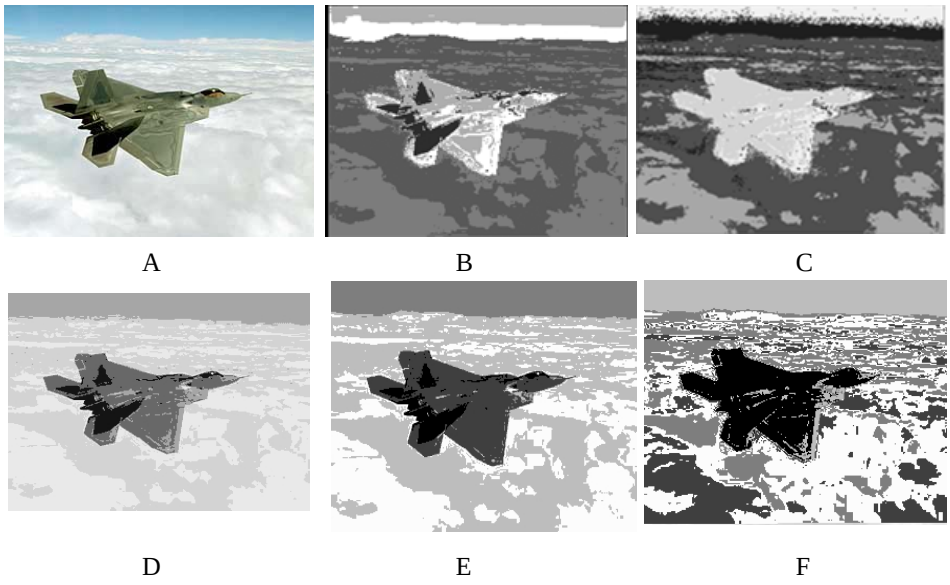


**Figure 5.** Airplane: A) Original, five clusters desired, B) PBA with 10% pollen depletion, C) Ant Colony Optimization, D) K-Means algorithm, E) Fuzzy C-Means algorithm, and F) Particle Swarm Optimization [23].

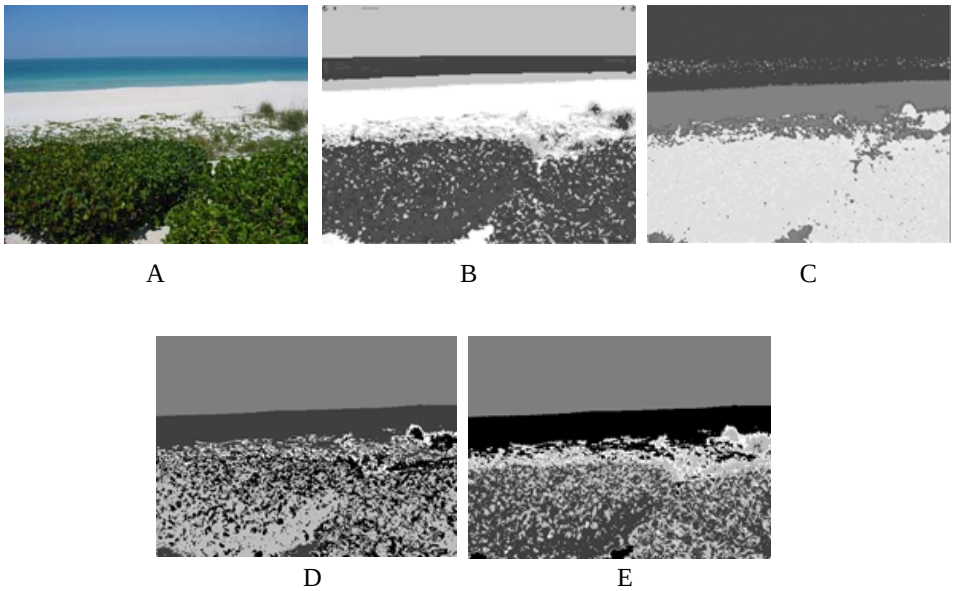


**Figure 6.** Beach: A) Original, five clusters desired, B) PBA with 10% pollen depletion, C) Ant Colony Optimization, D) Fuzzy C-Means algorithm, and E) Particle Swarm Optimization.

Pattern Data sets are another form of information that can be segmented and clustered using PBA. Provided in Tables 3, 4, and 5 are experimental results for the Iris, Glass, and Vowel Data Sets (the OD – Originating Data). For the iris data set the iris flower, for a particular family of irises is broken down into 3 groups of 50 each, each one having 4 identifying measurements that indicate which of the 3 groups it belongs to, the glass data set for glass mineral make-up, and vowel for vowel occurrence in words (http://archive.ics.uci.edu/ml/datasets.html). Please note that the highest and mean were chosen based on

the minimum variance from the outcomes generated with different fuzzy indexes and pollen depletion rate for the algorithms in Tables 3, 4, and 5.

**Table 3.** Data Set Clustering - The Iris Data Set - Accuracy Percentage
(Comparison Data From References [24, 25])

| Algorithm | Highest | Mean | Variance |
|---|---|---|---|
| PBA with 28% pollen | 98.00 | 89.20 | 0.74 |
| FCM | 89.33 | 89.26 | 1.53 |
| FWCM | 92.66 | 92.42 | 5.24 |
| NW-FCM | 92.66 | 90.97 | 19.17 |
| PCA93 | 92.67 | 80.00 | 281.67 |
| PCA96 | 95.33 | 77.23 | 179.36 |
| PCA06 | 92.00 | 79.64 | 263.24 |
| PCA {f1} | 92.66 | 80.01 | 274.04 |
| PCA {f2} | 92.00 | 79.27 | 266.47 |

**Table 4.** Data Set Clustering - The Glass Data Set - Accuracy Percentage
(Comparison Data From References [24, 25])

| Algorithm | Highest | Mean | Variance |
|---|---|---|---|
| PBA with 40% pollen | 61.68 | 52.63 | 2.79 |
| FCM | 55.14 | 49.12 | 2.66 |
| FWCM | 54.21 | 44.42 | 15.22 |
| NW-FCM | 47.66 | 41.11 | 5.77 |
| PCA93 | 46.26 | 38.93 | 0.97 |
| PCA96 | 45.79 | 35.75 | 6.28 |
| PCA06 | 62.62 | 45.62 | 16.71 |
| PCA {f3} | 55.61 | 48.82 | 11.41 |
| PCA {f4} | 56.08 | 46.20 | 15.51 |

**Table 5.** Data Set Clustering - The Vowel Data Set - Accuracy Percentage
(Comparison Data From References [24, 25])

| Algorithm | Highest | Mean | Variance |
|---|---|---|---|
| PBA with 10% pollen | 41.21 | 35.29 | 1.06 |
| FCM | 32.02 | 28.00 | 1.15 |
| FWCM | N/A | N/A | N/A |
| NW-FCM | N/A | N/A | N/A |
| PCA93 | 32.56 | 23.43 | 8.17 |
| PCA96 | 33.64 | 24.41 | 5.54 |
| PCA06 | 40.10 | 30.48 | 6.70 |
| PCA {f5} | 39.90 | 30.54 | 6.67 |
| PCA {f6} | 40.40 | 30.78 | 6.80 |

The results achieved indicate PBA performs better than Fuzzy C-means (FCM) clustering and its variations as well as the available variations of the Possibility Clustering Algorithm (PCA) [24, 25]. FCM, FWCM, NW-FCM, PCA93, PCA96, PCA06, PCA{F1}, PCA{F2} refer to different fuzzy clustering algorithms [24, 25]. Based on the results shown in Tables 3, 4, and 5, PBA achieves a higher mean and lower variance in most cases. In terms of the highest accuracy, our results are the best for Iris and Vowel data and a little lower than that of PCA06 (61.68% versus 62.62%) in Glass data.

## 5. Conclusions

We have presented a swarm of bees as a computational problem solving model, termed PBA. By contrasting existing bee model for problem solving with ours we have illustrated where similarities arise and where differences have purposefully been brought into play. An in-depth presentation of PBA that outlined its methods and the thought processes that originated is equations and their modeling by analogy with an actual bee swarm followed. Experiments were performed whose results indicate better than average performance by BA and clustering algorithms tested in our experiments.

PBA presents a new contribution to the bee swarm clustering algorithm family, namely its concept and computation of pollen depletion. This contribution allows the model to more accurately represent an actual landscape whose pollen, through the actions of bees and the advancement of the season, is constantly being diminished. Through the experimentation performed in the evaluation of this model, pollen depletion has been found to represent two ideas – one, for indicating the presumed difficulty level of the problem (the higher the pollen depletion rate used the more difficult the problem being solved is expected to be, but, conversely, the longer the algorithm will execute), and two, the reduction of both the number of bees and fields as the pollen season progresses and pollen diminishes being a truer representation of nature (i.e. early in the pollen season more bees are needed as there is more fields of pollen in the landscape, but, later in the season there are fewer quality 'fields in bloom' and thus a waste of energy and resources to send out masses of scout bees to search for what is certainly fewer quality fields).

Also elaborated upon and given to the bee swarm family, PBA fully fleshes out and uses a honeycomb at the hive as its: memory of the landscape, and the fields within it, and the quality of the pollen in those found fields. This frees the bees to be true singular agents who interact to improve their individually found solutions for the benefit of the hive as a whole without the need to know the solution being achieved by that whole – swarm intelligence personified.

This PBA model allows the user to specify a number of solutions desired (in this article, expressed through image segmentation and pattern classification, as a cluster count) and a means to express how quickly and easily the expected solutions might be found (expressed as the pollen depletion rate). Given by the algorithm is a guarantee of an ultimate converged solution to be presented, with the user neither guessing the proper number of iterations for the algorithm to execute nor expressing a minimum or maximum solution value to be achieved before the algorithm ends. The stability of results presented allows the user confidence of solution and, when uncertain of problem difficulty, the ability to analyze multiple runs of the algorithm at differing pollen depletion rates to select, with certainty, a rate whose solution generation reaches the level of detail desired without being overly time intensive to achieve.

The PBA model provides a unique, and very simple to use, method of problem solving. Having only two input values the user needs to supply allows the user to experience confidence of input and, by guaranteeing convergence and stability of result, high confidence in the solution presented. Our future task is to explore the PBA as an optimization tool.

## 6. Appendix

**An expansion of PBA**:

Psuedo-Code filled in with algorithmic equations/calculations

(The order of automated equation calculation is important as following equations depend upon previous equation results. Please note that the same equation number is used here):

A. Initialize variables

1. Number of Clusters C = user supplied number of clusters
2. Pollen depletion rate D = user supplied number value, $2<D<100$

3. Pollen in the Field V =100%

4. Scouts S = int(C/2)*D
   (1)
5. Fields in the Landscape L = int(S/C)
   (2)
6. Elite Pollen Sites Inspection X = int(C/3) (3 corresponds to 3 forager types)
   (8)
7. Regular Pollen Sites Inspection

   Y = int(C/3)+(1 if C/3 has remainder, 0 if C/3 has no remainder)
   (9)
8. Random Pollen Sites Inspection Z = 1
   (10)
9. Elite Foragers E = max(1,int(S/2))
   (5)
10. Regular Foragers R = max(1,int((V/2)/100)*E))
    (6)
11. Random Foragers G = C-(X+Y)
    (7)
12. Forager Deviation Q = .95 (0<Q<=1)
    (11)

B. Do while pollen not depleted and scout-able fields exist

   a. start scout bees loop – (for 1 to B)
      i.e. Scout excursions B=Initial L (equation (2)) minus current L (equation (18))
      (19)
      i. explore landscape with current Scout counts – Euclidean distance (this is where True Scouts (TS) is calculated)
      ii. create fields from found pollen (FP) of current scout bees – Build Honeycomb Cells
      iii. evaluate and replace fields with any better found fields – Store found pollen inside honeycomb cells
   b. end scout bees loop
   c. apply pollen depletion (modifying numbers of bees and fields)
      1. Pollen in the field V = V - D; if V<=0, V=0
         (3)
      2. Recalculated Scouts RS = int(min(TS,S)*V/100)
         (4)
      3. Fields in the Landscape L = int(RS/C)
         (18)
      4. Scout repeated excursions B = Initial L minus current L
         (19)
      5. Elite Foragers E = max(1,int(RS/2))
         (5)
      6. Regular Foragers R = max(1,int((V/2)/100)*E))
         (6)
      7. Random Foragers G = C-(X+Y)
         (7)

   d. start forager bee(s) loop
      (loop for initial L time;, only the fittest fields/cells, equal to the count of L, are revisited, regardless of the true size of the honeycomb and the found fields it represents)

(loop X times for E, then Y times for R, then Z times for G)

i. forager bee(s) watch waggle dancing of the scout bee(s) (selecting which found pollen they'll exploit) – Waggle Dance Floor

ii. explore/exploit within the recruited field – Uses Scouts' indicated found pollen source as the initial center, then (upon improvement by a forager) the best center yet achieved for this pollen location in the field.

Forager Exploration W=

1. W = random(f)*Q*(1 or -1 (chosen randomly in order to overshoot or undershoot the next solution exploitation)) (12)
2. If W = 0 then W = Q-(f/100)* (1 or -1 (chosen randomly in order to overshoot or undershoot the next solution exploitation )) (13)
   Forager Exploration (hazard recalculation may change W value from equations (12) and (13))
   a. for (E) If a random decimal number <=1-Q then W = random(OD) (14)
   b. for (R) If a random decimal number <=Q*(D/100) then W = random(OD) (15)
   c. for (G) If a random decimal number <Q then W = W + random(OD) (16)
3. Forager Found Pollen = (scout solution (FP) that attracted the forager) + W (Note: Using FP as discussed in section III(C)) (17)

1. evaluate/replace pollen source in field/cell – evaluate forager(s) found pollen as affects the field's fitness as a whole; if the field's fitness improves then the forager's calculated found pollen replaces the scouts' (or previous foragers') found pollen in the cell in the hive which this forager used as its initial exploration value (i.e. the FP to which this forager was recruited).

e. end forager bee(s) loop

f. Scouts S = Recalculated Scouts RS (i.e., S = RS) (These are the scouts, which have been reduced due to pollen depletion, that can explore in the next iteration)

g. repeat at "do while pollen…" (i.e. go to B above).

C. best remaining field's values is solution.

## 7. References

[1] M. O'Malley, The wisdom of bees: what the hive can teach business about leadership, efficiency, and growth. Portfolio Harcover, 2010.

[2] T. D. Seeley, P. K. Visscher and K. M. Passino, "Group Decision Making in Honey Bee Swarms," American Scientist, Vol. 94, pp. 220-229, May-June 2006.

[3] D. Teodorovic and M. Dell'Orco, "Bee Colony Optimization - A Cooperative Learning Approach to Complex Transportation Problems," Advanced OR and AI Methods in Transportation, 51-60, 2005.

[4] P. Navrat, A. B. Ezzeddine, L. Jastrzembska and T. Jelinek, Exploring the bee hive metaphor as a model for problem solving: search, optimisation, and more, Web Intelligence and Intelligent Agents , 2010. ISBN 978-953-7619-85-5

[5] D. Karaboga, “An idea based on honey bee swarm for numerical optimization,” technical report-tr06, October, 2005.
[6] K. M. Passino, T. D. Seeley and P. K. Visscher, “Swarm cognition in honey bees,” Behavioral Ecology and Sociobiology, Vol. 62, No. 3, pp. 401-414, Jan. 2008.
[7] P. M. Kevin, Honey bee swarm congition; decision-making performance and adaptation, 80 International Joumal of Swarm Intelligence Research, 1(2), 80-97, April-June 2010.
[8] V. Trianni, E. Tuci, K. M. Passino and J. A. R. Marshall, Swarm cognition: an interdisciplinary approach to the stud of sel-organizing biological collectives, Received: 18 October 2010 / Accepted: 25 November 2010 / Published online: 23 December 2010© Springer Science + Business Media, LLC 2010
[9] X. S. Yang,,Nature-Inspired Metaheuristic Algorithms: Second Edition, Luniver Press, 2008.
[10] D. T. Pham, S. Otri, A. Afify, M. Mahmuddin and H. Al-Jabbouli, “Data clustering using the bees algorithm,” Proc. 40th CIRP International Manufacturing Systems Seminar, Liverpool, 2007.
[11] X. S. Yang, “Engineering Optimization via Nature-Inspired Virtual Bee Algorithms,” IWINAC 2005, Lecture Notes in Computer Science, 3562, 317-323, 2005.
[12] L. Khan, I. Ullah, T. Saee and K. L. Lo, “Virtual bees algorithm based design of damping control system for TCSC,” Australian Journal of Basic and Applied Sciences, 4(1): 1-18, 2010.
[13] D. Karaboga and B. Basturk, “On the Performance of Artificial Bee Colony (ABC) algorithm,” Applied Soft Computing, 8, 687-697, 2008.
[14] H. F. Wedde, M. Farooq and Y. Zhang, “BeeHive: An Efficient Fault-Tolerant Routing Algorithm Inspired by Honey Bee Behavior,” Ant Colony, Optimization and Swarm Intelligence (Ed), M. Dorigo, Lecture Notes in Computer Science 3172, Springer Berlin, pp. 83-94, 2004.
[15] R. Akbari, A. Mohammadi, and K. Ziarati “A powerful bee swarm optimization algorithm,” INMIC 2009. IEEE 13th International conference, 2009.
[16] L. P. Wong, M. Y. H. Low and C. S. Chong, “A generic bee colony optimization framework for combinatorial optimization Problems,” Proceeding AMS '10 Proceedings of the 2010 Fourth Asia International Conference on Mathematical/Analytical Modeling and Computer Simulation IEEE Computer Society Washington, DC, USA ©2010.
[17] E. Bonabeau, M. Dorigo and G. Theraulaz, Swarm Intelligence: From Natural to Artificial Systems, Oxford University Press, 1999. ISBN-13: 978-0195131598
[18] M. Kazemian, Y. Ramezani, C. Lucas and B. Moshiri, “Swarm clustering based on flowers pollination by artificial bees,” Chapter 8, Swarm intelligence in data mining, Abraham, Ajith, Grosan, Crina, Ramos, Vitorino, 2006.
[19] K. Sundareswaran and V. T. Sreedevi, “Development of novel optimization procedure based on honey bee foraging behavior,” IEEE International Conference on Systems Man and Cybernetics, 2008.
[20] C. A. Tovey, “The Honey Bee Algorithm,” www.isye.gatech.edu, 2004.
[21] M. S. Packianather, M. Landy and D. T. Pham, “Enhancing the speed of the Bees Algorithm using Pheromone-based Recruitment,” IEEE International Conference on Industrial Informatics, pp. 789-794, 2009.
[22] J. T. Tou and R. C. Gonzalez, Pattern Recognition Principles, Addision-Wesley, 1974.
[23] C. C. Hung and L. Wan, “Hybridization of particle swarm optimization with the K-means algorithm for image classification,” IEEE Symposium on Computational Intelligence for Image Processing. Nashville, 2009.
[24] C. C. Hung, S. Kulkami and B.-C. Kuo,, “A new weighted fuzzy C-means clustering algorithm for image classification in remote sensing,” IEEE journal of Selected Topics in Signal Processing, vol. 5, no. 3, pp. 543-553, June 2011.
[25] J. Zhou and C. C. Hung, “A Generalized Approach to Possibilistic Clustering Algorithms,” International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 2007.

**David Bradford Jr.** received his B.S. in Fine Arts from Auburn University, AL in 1992 and his M.S. degree from Southern Polytechnic State University in Computer Science in 2010. He is currently working on Business Intelligence projects in software applications. His research interests are swarm intelligence, pattern recognition, and image processing.

**Chih-Cheng Hung** received his B.S. in business mathematics from Soochow University in Taiwan, and his M.S. and Ph.D. in Computer Science from the University of Alabama in Huntsville, AL in 1986 and 1990, respectively. He is a professor of Computer Science at Southern Polytechnic State University, GA. He developed image processing software tools with the Department of Image Processing Applications at Intergraph Corporation from 1990 to 1993 and an associate professor at Alabama A&M University from 1993 to 1999. He is the program co-chair for the Association of Computing Machinery (ACM) Symposium on Applied Computing (SAC 2012) – Riva del Garda (Treno), Italy from March 25 to 29, 2012. His research interests include image processing, pattern recognition, neural networks, genetic algorithms, and artificial intelligence.