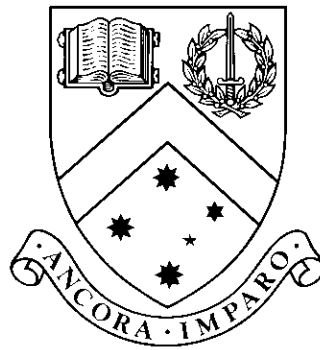


**Simplified and Optimised Ant Sort for Complex Problems:
Document Classification**

by

Benjamin Walsham



Thesis

Submitted by Benjamin Walsham

in partial fulfillment of the Requirements for the Degree of

Bachelor of Software Engineering with Honours (2150)

in the School of Computer Science and Software Engineering at

Monash University

Monash University

November, 2003

© Copyright

by

Benjamin Walsham

2003

Melissa for always wanting to help, even at those times when you don't know how. And to my family for being there, always.

Contents

List of Figures	vi
Abstract	viii
1 Introduction	1
1.1 Direction of the thesis (Aim, goals)	3
1.1.1 Increasing Complexity	4
1.1.2 Generic and Iterative Sorting	5
2 Literature Review	6
2.1 Natural Computation	7
2.2 Swam Intelligence	8
2.3 Ant Sort Algorithm	10
2.3.1 Technical Description of Recent Works	12
2.4 Automated Classification and Semantic Similarity	17
2.4.1 Classification and Automated Classification	17
2.4.2 Semantic Similarity Measures	17
3 Methods	20
3.1 Document Space Mapping	20
3.2 Metrics	21
3.3 Data Selected	21
3.4 Similarity Measure versus Definition of Characteristics	22
3.5 Pre-existing Ant Sort Parameters	23
3.6 Software/Hardware Used	23

3.6.1	Comparison of WordNet Similarity measures and justification for the selection	24
3.6.2	Anty Software	24
3.7	Method of Evaluation	25
3.7.1	Evaluate Increasing Complexity	25
3.7.2	Evaluate Document Sorting	26
3.7.3	Evaluate Internal Sorting	26
3.7.4	Evaluate Generic and Iterative Sorting	26
3.7.5	Evaluate Optimal Parameters	27
4	Results	29
4.1	Effect of Increasing Complexity	29
4.1.1	Optimal Parameters and Values	33
4.2	Generic and Iterative Sorting	47
4.2.1	Internal Sorting	47
4.2.2	Document Sorting	48
5	Discussion/Conclusions	51
5.1	Effect of Increasing Complexity and Optimal Parameters and Values	51
5.2	Generic and Iterative Sort	52
5.2.1	Internal Sort	52
5.2.2	Document Sort	52
6	Future Work	54
7	Appendix - Text Documents Used	56
8	Appendix - Extended Word Clustering Results	57
	References	60

List of Figures

1.1	Left: Intial random scatter of objects. Middle: Partially clustered objects after 5,000 turns. Right: Dense clusters forming 20,000 turns.	2
2.1	On the left, unsorted randomly located objects. On the right, partially clustered objects.	9
2.2	Pheromone trails used to determine the shortest path to the food source [Perez-Uribe, n.d.]	11
2.3	Illustration of Chialvo and Millonas' directional change weighting measure .	19
4.1	Comparison of differing complexities effect on impurity levels	30
4.2	Comparison of differing complexities effect on quantities of clusters formed	32
4.3	Increasing iterations using default for other parameters effect on quantity of clusters (piles), seconds, impurity and density (2-Colour objects). Y-axis values are arbitrary values in order to show the trends of all metrics together	34
4.4	Increasing iterations using default for other parameters effect on quantity of clusters (piles), seconds, impurity and density (10 Attribute objects). Y-axis values are arbitrary values in order to show the trends of all metrics together	35
4.5	Increasing quantity of objects on a fixed grid size effect on quantity of clusters (2 Colour objects)	36
4.6	Increasing quantity of ants effect on quantity of clusters (2 Colour objects)	38
4.7	Increasing stagnation control effect on the impurity levels of clusters (2 Colour objects)	40
4.8	Increasing directional change frequency effect on the quantity of clusters (2 Colour objects)	42
4.9	Increasing directional change frequency effect on the quantity of clusters (100 Attribute objects)	44

4.10	The effect of incorporating pheromone trails on the quantity of clusters (2 Colour objects)	46
4.11	The partially accurate clustering of 50 different derived document objects; each shade represents 1 of 5 actual texts	49

Simplified and Optimised Ant Sort for Complex Problems: Document Classification

Benjamin Walsham, BSE(Hons)
Monash University, 2003

Supervisor: David G. Green

Abstract

This thesis builds on established work, making use of natural solutions to solve computer-based problems, specifically inspired by the clustering that real ants perform on their larvae and corpses. We take the Ant Sort algorithm in a different direction to the approaches previously used when sorting complex objects. This is achieved in a divide and conquer style, initially sorting complex objects by their simpler attributes, before using the resultant information to sort the complex objects themselves. The other key goal is to examine the effect of complexity on the Ant Sort algorithm and uncover the data-to-parameter dependencies of the Ant Sort process.

Simplified and Optimised Ant Sort for Complex Problems: Document Classification

Declaration

I declare that this thesis is my own work and has not been submitted in any form for another degree or diploma at any university or other institute of tertiary education. Information derived from the published and unpublished work of others has been acknowledged in the text and a list of references is given.

Benjamin Walsham
November 3, 2003

Chapter 1

Introduction

Information Technology solutions are used for solving numerous problems. A computer's programs are able to solve these problems using algorithms (simply, a listed set of instructions). The variety and changeability of the technological landscape and social and financial pressures for better (faster, cheaper, etc.) solutions continues to drive computer hardware and software advances. Software is an interesting domain because for the most part, the functionality and the limitations are created by software itself, compared to the domain of hardware, which has specific physical properties that guide and determine possible limits and function.

Software algorithms are numerous and varied. Frequently, new hardware technology or user-demand creates software problems in new fields, previously without software solutions. In some cases, the inspiration for the required algorithms comes from observations of natural processes. This is the field of 'Natural Computing', which is further divided into many additional computer science fields, such as AI (Artificial Intelligence). AI is an ongoing attempt to replicate the behaviours associated with observed human intelligence. Ants and other social insects form the basis for another example of Natural Computing. They have provided the inspiration for a number of computing problem areas, resolved using different facets of observed 'Swarm Intelligence'. Some areas of application have been network routing optimisation, factory scheduling, and sorting and categorisation. Categorisation is a process used to create manageable and useful retrieval methods for large quantities of information or items. In the 'Information Age' huge amounts of information exist, especially within computer systems, the prime and obvious example being the Internet.

The Ant Sort algorithm derived from *Pheidole pallidula* [Deneubourg, Goss, Franks, Sendova-Franks, Detrain and Chretien, 1991], *Lasius niger* [Chretien, 1996a] and *Messor sancta* [Chretien, 1996b] ant species natural behaviour has a number of interesting and useful (to computing fields) properties, some of which exist in other insects and Swarm Intelligence

fields. The key aspect, in all these cases, is that complex solutions can be resolved using numbers of simple homogeneous individual components. Using this model a good level of fault tolerance, distributed processing and variable conditions can be handled by the algorithms. The behaviour noted in the three aforementioned species of ant is their ability to sort larvae and/or corpses into clusters (see Figure 1.1). The process can be simplified into these steps: an ant wanders at random, picks up an item when it comes across one, continues wandering at random until it comes across a similar item (to the carried item), then it drops the carried item next to it. The process is reinforced by autocatalytic (positive) feedback, wherein the surrounding environment regulates the appropriate behaviour. For example, assuming an ant is more likely to pick up an item if there are few items in the perceived surroundings, then when the item is picked up the likelihood of removal of further items from the same area is increased. Stigmergy is the general term describing the phenomenon where self-organisation is achieved through the environment (indirect interactions).

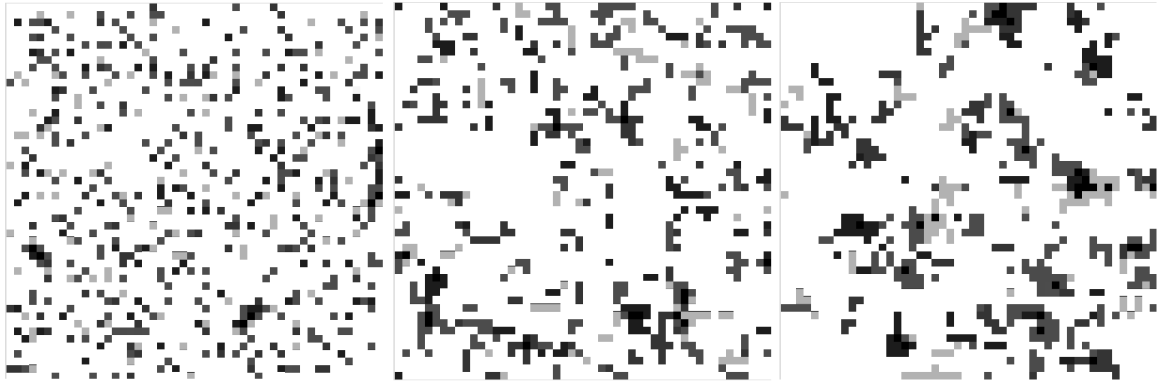


Figure 1.1: Left: Intial random scatter of objects. Middle: Partially clustered objects after 5,000 turns. Right: Dense clusters forming 20,000 turns.

Using the Ant Sort to cluster various items has already occurred. It is achieved with a distance (or similarity) measure between each item, determining the likelihood of clustering the items together. Solutions that are more complex have also been solved with modified versions of the Ant Sort however only by adding behaviour that is more complex. Examples are: Lumer and Faieta [33] suggested using ant memory to improve performance, Handl and Meyer [21] used pre-processing to organize data and Monmarche [37] alternated Ant Sort and another classification algorithm (K-means) to achieve more accurate classifications overall. The added complexity affects the application of the algorithm to its main areas of advantage.

This thesis provides an examination of the effect of complexity on the Ant Sort algorithm and provides an alternative solution to sorting complex items.

1.1 Direction of the thesis (Aim, goals)

The major goals of this thesis are to attempt to provide the following: a thorough examination of the relationship between specific parameters and methodologies associated with the Ant Sort algorithm and the effect of object complexity upon them; a new direction for the Ant Sort algorithm when sorting complex objects via a generic and iterative approach. Whilst achieving these main goals, the secondary objective, efficiency, will be addressed: first, as it applies to using the Ant Sort iteratively to sort complex objects; second, as determined by the dependency or independency of the data sets, to the key parameters of the algorithm.

This new direction is provided through the following goals:

1. Sorting simple objects and examining the effect on the Ant Sort algorithm's efficiency as their complexity is increased.
 - (a) Evaluating existing parameters to find the optimal functionality and values. Examining parameters' relationships to data sets, increased efficiency and the effects upon the final clustered output of the sorting process.
 2. Determining whether the iterative application of the Ant Sort can cluster complex objects.
 - (a) Identifying the effectiveness of sorting and scoring complex objects internally.
 - (b) Determining whether this can successfully be applied to text documents.
- Generically applying the Ant Sort algorithm consists of attempting to use the simple algorithm iteratively and through this, apply the method identically over varying data sets.

- Varying data sets refers to all the different types of data for which there exists reason to sort.
- In order to use a simple Ant Sort to sort complex objects, it is necessary to give these objects a (comparable) score, allowing the ants to make simple decisions on the similarity of each object.
- The problem is approached from both simple and complex angles. In the first stage, a simple object (a single attribute object) is examined by increasing its complexity. In the second stage, a complex object (text document) is examined by dividing it into less complex components.
- The text document object is too complex to be sorted simply. Thus, it is first sorted internally by its word objects and then given a simplified score before the document objects are sorted.

To restate, if there are complex objects (without existing comparable ratings) then can an Ant Sort be used to rate each of these objects internally (first iteration) and then classify the whole system (second iteration)? Can this be further expanded so that the first iteration is divided into multiple iterations, if the objects are suitably complex? Ultimately, this begs the question, does a simple Ant Sort algorithm allow an extensible and generic tool for the sorting of various sets of complex data?

1.1.1 Increasing Complexity

In order to examine the effect of sorting increasingly complex objects, we apply the Ant Sort algorithm to a fixed set of colour objects. Each level of complexity is evaluated for parameter and functionality (see chapter 3 for detail) independent of the results of other complexity levels. This process is repeated over sets of objects with 1-attribute, 10-attributes, 100-attributes, 1,000-attributes and 10,000-attributes. The resulting metrics are then used to establish the observed trends and constants of the Ant Sort caused by increasing object complexity. In addition, complexity's effects on clustering density and speed are examined.

Optimal Parameters and Values

Optimal parameters and functionality refers to those derived from the results of the effect of increasing complexity and the variation that occurs when the Ant Sort is applied to completely different data sets, such as word and document data sets.

1.1.2 Generic and Iterative Sorting

The validity of generic and iterative sorting is evaluated from the combined results of the internal and external document clustering. These results will be dependent on whether the categorisations (clusters) of duplicated textual document objects are closely coupled.

Internal Sorting

Before the larger textual documents can be sorted, each of the documents will initially be sorted ‘internally’ (by word), that is by performing the clustering process on the individual words of the document (first 200 words, refer to chapter 3). From this process, each document will be assigned attributes from the resultant cluster formations, following which they, in turn, will be clustered against other documents.

Document Sorting

A selection of freely available documents authored by Shakespeare and Edgar Alan Poe are used. These literary works have been selected due to their un-copyrighted availability (from project Gutenberg). Additionally, the differing writing styles and topics potentially provide for an interesting opportunity for empirical evaluation. Word and document sorting will use the optimal parameters found when sorting objects in the first stage.

Chapter 2

Literature Review

The Ant Sort algorithm is a bottom-up approach towards clustering and classification problems that typically rely on top-down algorithms. Ant Sort gets its name from observations of some species of ants. The observations are that an individual ant makes simple decisions, whether to pickup or drop an object, based solely on local information (the ant's immediate environment). From this numbers of ants can cluster together groups of objects. The homogenous and independent behaviour of the ants also provides high fault tolerance, as clustering will take place whether there are 1 or 100 ants. Additionally, ants do not need to know the number of objects or a defined number of categories and therefore provide a high level of flexibility for sorting unknown or changing data. These behaviours are particularly useful as it means the Ant Sort algorithm can be applied to applications unsuited for a centrally-controlled top-down approach. Such areas are distributed processing, fault tolerant applications and applications on unquantified datasets.

The Ant Sort algorithm belongs to a class of algorithms that are inspired by observations of processes in nature. To understand the Ant Sort's features and significance it is valuable to examine its history and provide an overview and justification of natural computation. The great-grandparent of Ant Sort it is also enlightening to understand the minor supporting roles that automated classification and semantic analysis play, in a number of current applications of Ant Sorting. From this broad base relevant specialisations are selected and examined from natural computation onto swarm intelligence, general ant algorithms and specifically ant sort algorithms. Meanwhile automated classification branches into textual document clustering; and semantic analysis discusses automated tools 'Similarity' through WordNet and 'LSAm Mercury' through Latent Semantic Analysis. These tools are used to achieve a form to textual based clustering. Slowly these branches all bend towards latest research possibilities: using the Ant Sort algorithm to resolve complex automated classification problems (including textual document clustering), whilst still using generalised homogeneous simple ants.

2.1 Natural Computation

Natural Computation is a general category which refers to both the underlying computing that occurs in actual processes in nature and also computer science models and algorithms derived or inspired by natural processes [LCNC, n.d.]. Natural Computation develops in response to problems in computer science and the observations of the problem being resolved in nature. Natural Computing has already provided the impetus for a number of different computer science fields, such as: (Artificial) Neural Networks (ANN), Evolutionary Computation, Artificial Intelligence (AI) and Swarm Intelligence. Each of these has a related computational problem and natural solution pair.

The need for faster processing solutions has led to the concept of parallel computing (multiple processors computing the parts of the same problem in order to calculate it faster). The goal is adaptive parallel distributed processing and ANNs use mathematical models derived from observations of the parallel processing that occurs in the brains of mammals. Computing Neuroscience is a developing area of research, based upon both neurosciences' expanding understanding of the brain and the interesting application of concepts and theories discovered by the Artificial Neural Networks field. This is interesting because concepts originally taken from biology and transformed into artificial processes are now being applied to help further the understanding of the biological processes themselves.

It is often important to determine the most efficient ('fittest') solutions in computing problems. Evolutionary Computation is based upon the natural selection process in which the 'fittest' members (solutions) of a species survive to pass on to the next generation. The origins of Evolutionary Computation can be traced back as far as the late 1950's and early 60's with works by Bremermann [Bremermann, 1962] and Friedberg [Friedberg, 1958]. However the field didn't really begin to expand until the late 70's and has continued expanding rapidly since. Evolutionary computation uses reproduction, mutation, and recombination and also some sort of positive feedback to achieve a 'survival of the fittest', natural selection process. The major application of evolutionary computation is in areas of optimisation [Back, Hammel and Schwefel, 1997] where a number of different (usually randomly selected) solutions or sub-solutions are put through the evolutionary process with the fittest (most optimal) surviving. This can also lead to previously unthought of solutions when mutation and recombination are applied.

Artificial Intelligence is a response to the problem of creating 'smarter' computers and the obvious natural solution, human intelligence. The original proposal for AI is credited to Alan Turing, specifically in his (unpublished) paper, 'Intelligent Machinery' [Turing, 1948]. In this paper, Turing develops the requirements and a measurement of intelligent machinery. That Turing's measurement has been argued to be faulty by Searle's [Searle, 1980] (in)famous Chinese Room Argument, followed by counter claims including those by

Hasser [Hauser, 1993; Hauser, 1997]. Regardless of the debate over AI definitions, the continuing development in the field of AI is testament to the possibilities that exist by applying concepts derived from natural processes to computer science problems.

In some computing applications certain properties are especially important, such as: fault-tolerance, distributed processing and variable data. Observations of termites: nest building; ants: clustering their larvae; and the insects ability to achieve this without individual knowledge beyond their immediate environment. This leads to the concept of Swarm Intelligence wherein a population of individuals, all, with only minimal local knowledge being able to solve complicated problems. Swarm Intelligence, is another prime example of a sub-category of natural computation, it is the process of solving complex problems with a number of individuals, each of whom only has simple behaviour.

2.2 Swam Intelligence

Swarm Intelligence uses emergent computing, that is when a complex solution emerges from a combination of simple behaviour by simple individuals. This means there is no overseeing, no guiding force controlling and ensuring the complex solution. Despite this the solution created is not a random fluctuation but a method of nest building, combining resources, food foraging, defence and more for social insects. This is clearly not just random chance but a balance critical to their survival.

Swarm Intelligence has been applied to many different computer science problems with the differing behaviour of social insect species providing more insight into potential applications. Applications such as factory-scheduling [Doty and Aken, 1993], telecommunication and network routing optimisation [Caro and Dorigo, 1997], sorting and categorisation [Monmarche, 1999; Handl and Meyer, 2002] and shortest path algorithms have been derived from observation of natural solutions. Artificial ant colonies have also been used as a method of determining good solutions to the classic Travelling Salesman Problem (TSP) [Dorigo and Gambardella, 1997] and the Quadratic Assignment Problem (QAP) [Gambardella, Taillard and Dorigo, 1997]. This is achieved by specific specialisations to the general concepts of swarm intelligence and can involve additional processing with different algorithms.

The idea of an Ant Sort algorithm (see section 2.3) is specifically derived from research into the *Pheidole pallidula* [Deneubourg et al., 1991], *Lasius niger* [Chretien, 1996a] and *Messor sancta* [Chretien, 1996b] species of ant. These species sort larvae and/or corpses to form clusters. This clustering is a result of autocatalytic feedback, occurring because the more of a type of item in a region, the higher the probability of an ant dropping the same item in that region. For comprehension (see Figure 2.1) this can be reduced to the overly simplified notion that an ant wanders at random, finds an item and picks it up. Then the

ant wanders at random carrying the item until the ant comes across another item then the ant drops the carried item and again wanders off at random.

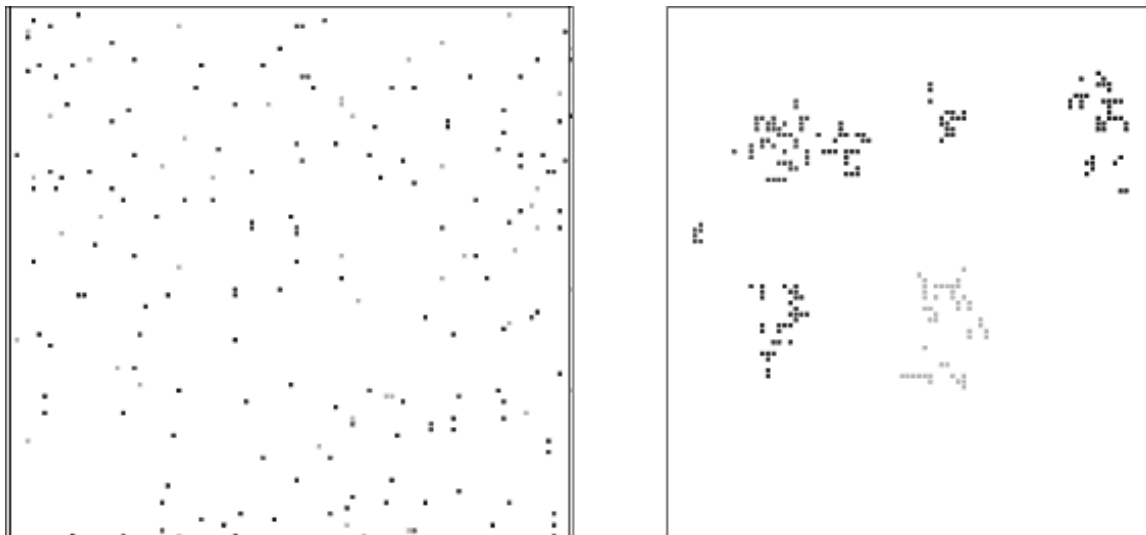


Figure 2.1: On the left, unsorted randomly located objects. On the right, partially clustered objects.

2.3 Ant Sort Algorithm

An early study in using the logic of ecological Ant Sorting for automated clustering problems was by Deneubourg et al [Deneubourg et al., 1991]. Deneubourg et al used a population of randomly moving artificial ants to simulate the experimental results seen with real ants (*Messor sancta*) clustering their corpses. Two related algorithms were proposed as models for the observed experimental behaviour, of chief importance, the item pickup and drop probability mechanism. From this study the model which, least accurately modelled the real ants was the most applicable to automated clustering problems in computer science. Lumer and Faieta [Lumer and Faieta, 1994] advanced the work (by Deneubourg et al), modifying the algorithm to include the ability to sort multiple types, in order to apply the ant sorting concept to exploratory data analysis.

The advantages of swarm intelligence or ant based systems are well documented [Ramos, Muge and Pina, 2002; Dorigo and Caro, 1999; Li, Martinoli and Abu-Mostafa, 2002; Fukuda, Funato, Sekiyama and Arai, 1998] and include fault-tolerance, scalability, distributed processing, modularity, effectiveness on unquantified data [Parpinelli, Lopes and Freitas, 2002], robotic applications [Li et al., 2002; Fukuda et al., 1998] and the simplicity of individual agents. Disadvantages are: suboptimal performance, random behaviour and trapped locally-optimal results. Suboptimal performance is especially seen in fixed domain problems, such as TSP where less flexible solutions are able to perform better [Ramos and Merelo, 2002]. However, in order to achieve these increases, in performance, generally costs one or more of the above listed advantages. For example, to achieve a closer to optimal performance in document classification, the K-means and ISODATA classification algorithms both require prior knowledge of the number of categories [Monmarche, 1999].

The *Macrotermes* termites use stigmergy [Grasse, 1984] to build their nests, stigmergy is a phenomenon whereby self-organisation is achieved through indirect interactions. An individual makes a change to the environment and a second individual responds (possibly later) to the modified environment, this is a function of ant pheromones. Some social insects use positive and negative feedback via stigmergy, including some species of ant. Positive feedback is a process of indirect recruitment a particular state of the environment causes an insect to begin to dig, for example. This insect's modification of the environment indirectly causes another insect to dig, and another and so on. However, at a certain threshold level there is saturation and the environment achieves a state causing negative feedback that is stopping further insects from beginning to dig.

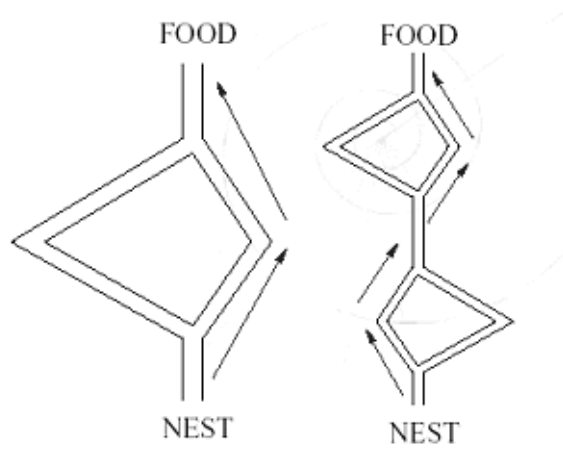


Figure 2.2: Pheromone trails used to determine the shortest path to the food source [Perez-Urbe, n.d.]

Real ants use stigmergy through pheromone trails and are able to determine the shortest path through a simple graph [Deneubourg et al., 1991; Dorigo and Caro, 1999; Goss, Beckers, Deneubourg, Aron and Pasteels, 1990]. This was shown in an experiment where an ant nest was placed with two bridges of differing lengths connected to the same food source (for example, see Figure 2.2). To illustrate this experiment, a simple description: if there are two ants and each selects a different bridge to traverse then the ant that gets to the food and returns (to the nest), first will have taken the shorter route. Now when the first ant again reaches a choice about the two trails it will sense the pheromone on the shorter bridge and be more likely to return the same way again. The second ant returns and makes a choice discovering an equal amount of pheromone on each trail and makes a selection. When this process is repeated the shorter trail builds up pheromone faster than the longer trail therefore making it more and more likely for an individual ant to select the shorter path, this illustrates the concept of positive feedback. In addition to this pheromone is constantly evaporating, thus when the ant on the longer route returns, over a longer time, more of this trail's pheromone will have evaporated.

Use of pheromones for artificial ants based on concepts from real ants [Goss et al., 1990] has been investigated. This allows ant pheromone trails to provide some sort of 'system memory', however the rules for depositing pheromones for each ant remains simple. Typically a constant amount of pheromone, η , is deposited by each artificial ant in each location as it moves through in each step. A constant amount, k , evaporates for each iteration of the algorithm.

Ant sorting uses a bottom-up approach and the concept is that in a swarm, simple ants can create overall colony behaviour that is complex. However, general research using artificial ants tends to resolve issues of complexity by adding complexity to the ants' behaviour to solve specific problem domains. Examples are: Lumer and Faieta [Lumer and Faieta, 1994] suggested using ant memory to improve performance, Handl and Meyer [Handl and Meyer, 2002] used pre-processing to organize data and Monmarche [Monmarche, 1999] alternated Ant Sort and another classification algorithm (K-means) to achieve more accurate classifications overall.

2.3.1 Technical Description of Recent Works

Recently, in 2002, Ramos et al [Ramos et al., 2002] and Ramos and Merelo [Ramos and Merelo, 2002] have, extended upon Deneubourg et al [Deneubourg et al., 1991] and Lumer and Faieta's [Lumer and Faieta, 1994] work, devising an ant clustering system (ACLUSTER). ACLUSTER avoids the additional complexity of: short-term memory, multiple ant types that move at different speeds (making the ant population inhomogeneous) and pre-processing. ACLUSTER instead uses the effects of stigmergy through pheromone trails to achieve unsupervised clustering. ACLUSTER has been applied to clustering both textual

documents (Spanish newspaper articles) [Ramos and Merelo, 2002] and digital image retrieval from databases [Ramos et al., 2002]. This work provides encouraging results: using a simpler and more distributed algorithmic approach Ramos and Merelo reached similar clustering results over the same number of iterations as those in Lumer and Faietas original results.

The existing research most closely correlating to textual document sorting via Antare Ramos and Merelo [Ramos and Merelo, 2002], Handl and Meyer [Handl and Meyer, 2002] and a minor work by Hoe et al [Hoe, Lai and Tai, 2002].

First Steps

The basis for current research in the Ant Sort algorithm is the original study by Deneubourg et al and the following crucial work by Lumer and Faieta. Deneubourg et al established equations to simulate the pickup and drop item probability as observed in ant clustering behaviour [Chretien, 1996b]. These equations measure the chance of an ant picking up or dropping an item at its current location:

$$P_p = \left(\frac{k_1}{k_1 + f} \right)^2 \quad (2.1)$$

Deneubourg et al's item pickup probability

$$P_d = \left(\frac{f}{k_2 + f} \right)^2 \quad (2.2)$$

Deneubourg et al's item drop probability

Where f , according to Deneubourg et al, is the number of items (N), that can be found in the short-term memory of length (T), and the maximum items in memory from any given time T is x then $f = N/Tx$. That is the number of items in memory divided by the maximum number of items potentially in memory. k_1 and k_2 are threshold constants such that when $f \ll k_1$ the probability of pickup is high when there are few items in the short-term memory of the ant, when $f \gg k_1$ there is a low pickup probability with many items. The inverse is the case for item dropping. Deneubourg et al used values of $k_1 = 0.1$ and $k_2 = 0.3$.

Extending the possibilities: application of ant sorting to exploratory data analysis

Deneubourg et al's work was followed by Lumer and Faieta's model [Lumer and Faieta, 1994] which incorporates a distance or similarity measure between different objects extending the previous model to perform clustering of multiple object types. Lumer and Faieta's

model was created in order to perform exploratory data analysis on databases. Their model requires an equation representing the distance between a specific object and all other objects within a given radius. The following equation assumes that an ant can perceive a radius around its current position (instead of Deneubourg et al's memory-based perception) forming a square area r^2 (radius radius). Letting $d(o_j, o_k)$ be the distance or alternatively the similarity measurement function between objects o_j and o_k and given that R represents the set of all objects with the radius r (excluding empty space), and that o_j is the object at the ants current position.

$$f(o_j) = \max \left(0, \frac{1}{r^2} \sum_{o_k \in R} \left[1 - \frac{d(o_j, o_k)}{\alpha} \right] \right) \quad (2.3)$$

Lumer and Faietas object distance measure

The factor α controls the discrimination level of the function, if α is too high then dissimilar items will be placed together but if α is too low then similar items will not be clustered. According to Handl and Meyer [Handl and Meyer, 2002], the optimal value for α cannot be found without prior knowledge of the data set, unless the value of α is adaptable. $f(o_j)$ is a measure of the distance or similarity between o_j and all other objects within the ants sensing radius. $f(o_j)$ is used instead of Deneubourg et al's fraction measure f , therefore Lumer and Faieta's revised item pickup and drop equations become:

$$P_p(o_j) = \left(\frac{k_1}{k_1 + f(o_j)} \right)^2 \quad (2.4)$$

Lumer and Faieta's item pickup probability

$$\begin{aligned} \text{if } f(o_j) < k_2, \\ P_d &= 2f(o_j) \\ \text{if } f(o_j) \geq k_2 \\ P_d &= 1 \end{aligned} \quad (2.5)$$

Lumer and Faieta's item drop probability

These equations use a similar application of k_1 and k_2 (as in Deneubourg et al's equations), the values used by Lumer and Faieta were $k_1 = 0.1$, $k_2 = 0.15$ and $\alpha = 0.5$. Note that the drop measure is only evaluated in the case of an ant carrying an item when its current position is empty, or the pickup measure when an unladen ants current position contains an object.

Using the sorting and scaling aspects of the Ant Sort algorithm to cluster textual documents

Lumer and Faieta also used: (inhomogeneous) ants with different moving speeds, behavioural switches and short-term memory. The authors also claimed that their algorithm supported cluster analysis and multidimensional scaling, a claim supported by Kuntz and Snyers [Kuntz and Snyers, 1994] and Kuntz, Snyers and Layzell [Kuntz, Snyers and Layzell, 1998], Handl and Meyer also support this claim but suggest that only limited scaling is possible without modification to the algorithm. Using ants with different moving speeds or jumps, should conceptually, result in the faster ants quickly creating coarse clusters and the slower ants refining these clusters. However, Handl and Meyer claim that this modification in isolation to other changes that is having different ants at different speeds doesn't improve the speed of clustering. But that simply larger jumps for all ants (up to 50% of the grid size [Handl and Meyer, 2002]) improves clustering speed.

Handl and Meyer have expanded upon Lumer and Faieta's algorithm by introducing: adaptive scaling, ant jumps, stagnation control and 'eager ants'. Adaptive scaling involved using an initially low value for $\alpha = 0.1$ (see Equation 2.3), and slowly raising this level. When few pickup and drop action occurred over a specified time period the ants' sensitivity to difference was reduced (α increased). This provided an effective method in determining appropriate values for α as it was able to adjust to the properties of the data set currently being clustered. The use of jumping ants changed the nature of the algorithm quite significantly, into: a form of more global sampling with directional bias [Handl and Meyer, 2002], rather than the continuous movement of ants typical of ant sorting algorithms. Stagnation control avoids an ant carrying an outlier item for too long, which causes the ant to be ineffectual, instead after a fixed number of attempts to drop an item; it is dropped regardless of the similarity measure. 'Eager ants' attempts to resolve the issue of an ant's time consuming search for a new item after dropping the previous one. Handl and Meyer resolve this by using a random selection from an index of un-carried items and directing the ant to the item's location. This imposes a directing central control, the index, upon the ants, which seriously undermines its suitability for key application areas of ant sorting algorithms, such as unknown quantity and variable datasets and distributed computing environments.

Re-simplification

The short-term memory used by Lumer and Faieta involves each ant having a precise memory of each of the last c items carried and their location. Thus when a new item is picked up it is compared with this 'sample' in memory, removing the need for the ant to discover an appropriate dropping position. Lumer and Faieta's 'behavioural switches' refers to their solution for the occurrence of locally optimised clustering after a defined period of iterations when ants encounter an unchanging cluster of items, the ants' behaviour switches causing the ants attempt to remove this unchanging cluster.

Ramos and Merelo [Ramos and Merelo, 2002] suggest that these, computational expensive and difficult complex parameter tuning difficulties with the Lumer and Faieta model, can be replaced by using pheromone trails. This pheromone trails can provide a ‘system memory’ for the overall swarm of ants, avoiding the requirement for ants to have any individual memory. Ramos and Merelo use a simple model devised by Chialvo and Millonas [Chalvo and Millonas, 1995] wherein an individual ant can be described by its position and direction and determining the probabilities of an ant moving between any particular pair of position and direction to any other pair of position and direction. Ramos and Merelo determined their pheromone weighting function based on work by Millonas [Millonas, 1992; Millonas, 1994] as follows:

$$W(\sigma) = \left(1 + \frac{\sigma}{\gamma + \sigma}\right)^\beta \quad (2.6)$$

Ramos and Merelo probability of moving to a location with pheromone density σ

Where the value β determines the randomness with which an ant follows a pheromone trail, $1/\gamma$ is an ant’s sensory capacity, essentially allowing an ant’s sense of pheromone to be ‘dulled’ at high concentrations and σ is the pheromone quantity of a given location. The normalised probability of going from location m to location k is given by [Chalvo and Millonas, 1995]:

$$P_{k|m} = \frac{W(\sigma_k) w(\Delta_k)}{\sum_l W(\sigma_l) w(\Delta_l)} \quad (2.7)$$

Lumer and Faieta’s item pickup probability

Where l is the set of locations surrounding m (of which k is one), Ramos and Merelo used values of $\beta = 3.5$ (pheromone following randomness (osmotropotaxic sensitivity [Chalvo and Millonas, 1995]), $\gamma = 0.2$ (inverse of sensory capacity) and used the directional change weighting measures as per Chialvo and Millonas [Chalvo and Millonas, 1995], $w(0) = 1$, $w(1) = \frac{1}{2}$, $w(2) = \frac{1}{4}$, $w(3) = \frac{1}{12}$, $w(4) = \frac{1}{20}$ (*as illustrated in*

2.4 Automated Classification and Semantic Similarity

2.4.1 Classification and Automated Classification

The process of classification can be achieved via two key methods: first, the rule-based approach, for example ‘Is this object blue?’, if so then place in the blue category, otherwise place in the ‘not-blue’ category. The question is then applied to each of the objects in turn, further sub-division can occur simply by adding more questions. Second, the similarity measure approach, each object can be compared against each other, producing a numeric value of their similarity, then if this value is higher (more similar) than a specified threshold (or discrimination level) then the objects are placed together in the same category. Of most interest to Ant Sorting applications is the similarity measure as the rule-based approach is a centrally controlled solution and requires more than local information to reach decisions.

The classification concept can quickly become difficult as: the quantity of objects or classes get large; the objects become complex with many attributes; or, the rules change according to purpose of the classification. Just how variable and relative the process of classifying is can be shown by this example: a music CD and a standard box of tissues are highly dissimilar, a CD has attributes of artist, track names whilst a box of tissues has none of these. However if one looks at classifying objects by weight then they are quite similar in comparison with an adult polar bear. This shows the truly arbitrary nature of classification, but also highlights the importance of the usefulness of the resulting categories.

Automated classification is required to sort large quantities of data, such as is found in when dealing with large databases. Automated clustering is the process, used to classify documents algorithmically; these operate by using measuring similarity between objects or groups of objects [Duda and Hart, 1973]. Automated clustering is often performed upon numeric datasets [Agrawal, Gehrke, Gunopulos and Raghavan, 1998; Bradley, Fayyad and Reina, 1998] but is also applied to categorical datasets, especially in regard to internet resource classifying [Yang, Slattery and Ghani, 2002]. A commonly used clustering algorithm, K-means, has a numerous different derivatives: standard, online, scalable and incremental, it is considered relatively simple and reasonably fast [Ordonez, 2003]. However K-means is centrally controlled and requires knowledge of object quantities to operate. K-means is consistent with the general approach of automated clustering and does not provide a suitable mechanism for achieving such clustering with an ant-based sorting algorithm.

2.4.2 Semantic Similarity Measures

Semantic similarity involves measuring how closely related two words or two sentences are. In order to quantify this similarity measure there needs to be a tool or resource which contains or can calculate by extrapolation the similarity measure between all possible pairs

of words or sentences. The WordNet database provides such a resource. Alternatively, this could be achieved through approximation, as is the case with LSAm Mercury; it uses vectors to represent such relations. Additionally there needs to be suitable methods that are consistent and accurate to actual language use methods of deriving these similarities.

WordNet, QueryData, Similarity

WordNet [Miller, 2003] is a lexical reference system which stores a database of words organised into synonym sets and contains different relations to link these sets. WordNet provides the framework to create lexical chains that is links from one word to another via a series of in-between words. This is measurable and a variety of different measurement techniques have been suggested: Leacock and Chodorow [Leacock and Chodorow, 1998], Jiang and Conrath [Jiang and Conrath, 1997], Resnik [Resnik, 1995], Lin [Lin, 1998], Hirst and St-Onge [Hirst and St-Onge, 1998], Adapted Lesk [Budanitsky and Hirst, 2001]. This list is constrained by those whose measures have been implemented by the WordNet-Similarity-0.07 software [Patwardhan and Pedersen, 2003]. Many of these measures rely upon a corpus of text, meaning that they need to be initialised (so to speak) and this is achieved using a large amount of pre-existing English text (corpus). WordNet-Similarity-0.07 software provides an implementation using WordNet and a perl package WordNet::QueryData [Rennie, 2000] in order to allow a choice between the measurement techniques and provides a function that returns a similarity measure between two words.

Latent Semantic Analysis, LSAm Mercury

Latent Semantic Analysis (LSA) is based upon the assumption that there is an underlying structure or meaning associated with the position of a word in the overall pattern of words in a textual document [Deerwester, Dumais, Furnas, Landauer and Harshman, 1990]. The process LSA uses is to convert the n documents with m words into a matrix and apply the Singular Value Decomposition (SVD) [Zhang and Rudnicky, 2002] method to the distance between the words in the matrix. Zhang and Rudnicky [Zhang and Rudnicky, 2002] have proposed an enhancement which incorporates 3-level knowledge of each words position in sentence, paragraph and document rather than just the document level. Recently, in 2001, Quesada et al [Quesada, Merelo, Castellano, Garcia and Cillero, 2001] developed an online version of LSA called LSAm Mercury, which is a tool allowing users to have an online profile that is modified by LSA.

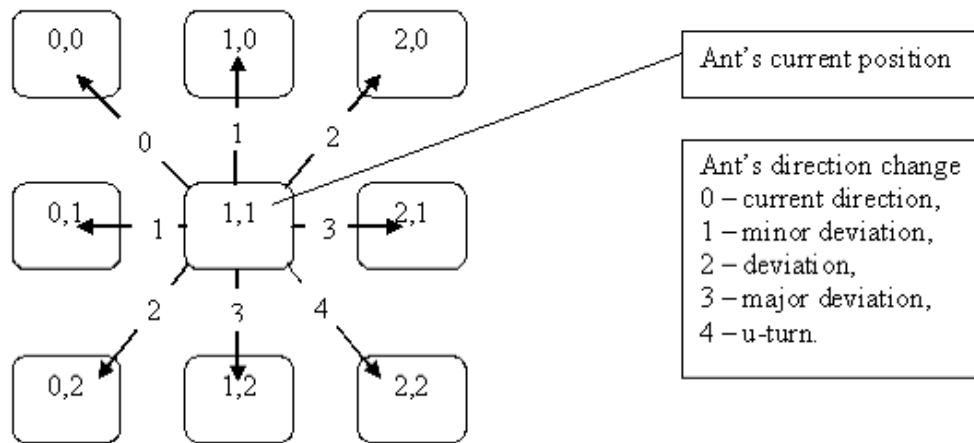


Figure 2.3: Illustration of Chialvo and Millonas directional change weighting measure, ant is located at location 1, 1 and has a current direction (orientation) towards location 0, 0.

Chapter 3

Methods

The key questions of this thesis: Effect of Increasing Complexity, Document Sorting, Internal Sorting, Generic and Iterative Sorting and Optimal Parameters, are addressed through the Anty software (see subsection 3.6.2) developed as part of this thesis and other pre-existing software packages (see section 3.5).

The key tasks involved with evaluating the key objectives of this thesis are listed below and detailed throughout this section:

1. Document Space Mapping
2. Metrics
3. Data Selected
4. Similarity Measure versus Definition of Characteristics
5. Pre-existing Ant Sort Parameters
6. Software/Hardware Used
7. Approach to Complexity

3.1 Document Space Mapping

Random mapping to the 2-dimensional toroidal grid will be used for all cases. This form of document space mapping is used for two reasons: previous Ant Sort experimentation has used this method (ease of comparison); using this method allows physical (robotic) applications of the Ant Sort algorithm to remain feasible. In all cases values are randomly assigned, removing any possible initial positional semantics.

3.2 Metrics

Foremost is the need for the software to produce useful results. To achieve useful results one needs to develop metrics to measure differences between each execution of part of the experiment. Previous works have typically only evaluated the Ant Sort in terms of clustering against turns (how many times all ants move), performance (number of seconds to execute), or empirical measures (ability to perform a particular task). The metrics taken for this thesis are:

1. Density
2. Impurity
3. Piles
4. Seconds

Density, Impurity and Piles are calculated together using a depth-first traversal algorithm to traverse the graph formed by connected items in the 2-Dimensional grid. The number of individual graphs found is the number of 'Piles of items. In each graph the minimum and maximum (x and y) coordinates are recorded and used to calculate the Density of the pile. This is averaged over all piles. Impurity takes the distance (or similarity) measure between items in a single graph/pile giving the minimum average distance between any item and every other item in the group. This is also averaged over all piles. The number of seconds for each execution is calculated, however this is just to provide a guide for the level of complexity variation between different types of item and parameter. This measure does not include any attempt to optimise execution speeds, since the aim of the thesis is to evaluate the achievability of the goals (as stated in chapter ??) and not to investigate practical implementation.

3.3 Data Selected

The differing data types used are:

1. Fixed set size Colour Objects - allows between 1 and 15 different object types to be selected: within each type, objects are considered identical, thus a distance of 0; between 2 different types, objects are considered completely different, thus a distance of 1.
2. Attribute Objects - allows for 1, 10, 100, 1,000 and 10,000 attribute objects: each attribute is an integer between 1 and 1000, the distance between objects is the normalised amount of the sum of differences between each attribute pair

3. Word Objects - each word object represents an English word from the subset of the WordNet database: the distance between words is calculated by the Leacock and Chodorow [Leacock and Chodorow, 1998] measure
4. Document Objects - document objects are formed from the results of an Ant Sort on a set of 200 word objects from the specified text: the document object contains the key word and some distance information from each of the largest clusters (up to 5) and this information is used to calculate a normalised distance between document objects.

The data selected to evaluate the effects of complexity are objects of varying attributes (1, 10, 100, 1,000 and 10,000), with each of these attributes being a randomly generated positive integer less than one thousand. While it may have been of interest to evaluate the complexity effect at levels of millions of attributes, this is beyond the capabilities of the Anty Software package. Colour objects are also used. They have a maximum distance measure between different types. Colour objects are examined in sets of two different types at a time, or fifteen. The textual documents to be used consist of some 5 different works by Shakespeare and Edgar Alan Poe (see 7) for a complete list of works]. Again due to technical limitations of the software and hardware used, the internal document sort is only applied to the first 200 words of each document. These limitations were required to obtain experimental results within the time frame allowed for this paper.

3.4 Similarity Measure versus Definition of Characteristics

This thesis only examines the variations in the Ant Sort algorithm parameters when using a similarity (distance) measure. This work doesn't intend to provide any comparison between the similarity measure approach and the meta-data (defined characteristics) style of approach which relies on specialised or differentiated object data to sort.

The similarity or distance between objects is calculated differently for the different types. The fixed set colour objects are considered as either perfectly similar or completely dissimilar. The distance between attribute objects is the normalised distance between every individual attribute in each object. The word object distance is the normalised value of the WordNet Leacock and Chodorow [Leacock and Chodorow, 1998] similarity measure as described in section 3.6.1. The document object is created from the key words of the clusters formed by the earlier internal sort of the document. The document object distance is calculated as the word similarity measure of the combination of these key words.

3.5 Pre-existing Ant Sort Parameters

Some Ant Sort parameters and functionality from previous research work by Lumer and Faieta [Lumer and Faieta, 1994], Handl and Meyer [Handl and Meyer, 2002] and Ramos et al [Ramos et al., 2002] have been reanalysed and some excluded. This decision was based whether their inclusion would affect the distributed applicability of the algorithm used for this research.

- Incorporated
 - Ant pheromones
 - Object-dependent pheromone trails
- Excluded
 - Ant memory
 - Different ant speeds
 - Inhomogeneous population
 - Direct ant access to object lists (lookup tables)

3.6 Software/Hardware Used

- Software
 - C
 - .NET Framework
 - Visual Studio .NET 2002 (Development Environment)
 - perl
 - WordNet 1.7.1
 - QueryData package
 - Similarity 0.07 package

With no existing software for ant based research it was necessary to develop the Anty Software package in order to examine the overall objectives of the thesis. .NET framework, C# and perl were used to develop the Anty Software package. These languages were selected for development in order to interface with the existing WordNet, QueryData and Similarity packages that allow for word similarity (or distances) to be calculated. The combination of WordNet, QueryData and Similarity packages provides a means of direct semantic comparison between 2 individual words. The option of using Latent Semantic Analysis (LSA) to provide the semantic analysis was dismissed due to lack of availability and the inability for direct semantic comparison.

3.6.1 Comparison of WordNet Similarity measures and justification for the selection

The WordNet-Similarity-0.07 software allows a choice between the semantic measures: Leacock and Chodorow [Leacock and Chodorow, 1998], Jiang and Conrath [Jiang and Conrath, 1997], Resnik [Resnik, 1995], Lin [Lin, 1998], Hirst and St-Onge [Hirst and St-Onge, 1998], Adapted Lesk [Budanitsky and Hirst, 2001], simple edge counts (inverted) or a random measure. The experimentation (not intended to be conclusive) resulted in the Leacock and Chodorow method being selected for use in this thesis. Initially, Jiang and Conrath, Resnik, Lin, and Adapted Lesk measures were removed from consideration as all rely on an existing corpus of text. This limits their applicability to calculating semantic distances between words that exist in the corpus, clearly unsuited to unknown content. The inverted edge measure, although not suffering from the aforementioned limitation, provides a simplified approach, however the distance is not well refined. An examination of the differences between the Leacock and Chodorow, and Hirst and St-Onge measures resulted in the former being evaluated as most suitable, especially in regard to result calculation times.

3.6.2 Anty Software

The Anty Software package has been developed to allow for the following concepts to be examined: (using the format `{minimum, maximum — alternate maximum value}`)

- Batch Processing
- Graphical Processing
- Graphical and XML Output
- Object Quantity - the number of objects assigned to the grid `{100, 9999 — dimension size x dimension size}`
- Ant Quantity - the number of ants used `{1, 500 — object quantity}`
- Ant Sensing Radius - the radius (grid locations) around which the ant can sense items (value is restricted to 1 when pheromones are used) `{1, 5}`
- Ant Step Size - a parameter that allows all ants to move faster (jump) `{1, 100}`
- Chance of Directional Change - the probability of an ant making a change in direction (applicable only when pheromones are not used) `{1, 2000}`
- Dimensions - allows for different sizes of toroidal grids `[50x50], [100x100], [250x250], [500x500]`

- Stagnation Control (Optional) - a control that causes an ant to be increasingly more likely to drop an object as time increases, reducing the likelihood of ants being stuck with a particular item [Handl and Meyer, 2002] ;1, 2000;
- Discrimination (or α) - the level of distinction between compared objects [Lumer and Faieta, 1994; Ramos and Merelo, 2002] ;1, 999; (10^{-3})
- Pickup Probability Constant - a constant value that affects the probability of an ant picking up an object [Lumer and Faieta, 1994; Ramos and Merelo, 2002] ;1, 999; (10^{-3})
- Drop Probability Constant - a constant value that affects the probability of an ant dropping an object [Lumer and Faieta, 1994; Ramos and Merelo, 2002] ;1, 999; (10^{-3})
- Pheromone (Optional) [Ramos and Merelo, 2002]
 - Constant Pheromone Deposit Rate - amount of pheromone an ant leaves in its current location each turn ;1, 99; (10^{-2})
 - Constant Pheromone Evaporation Rate - amount of pheromone that evaporates from all grid positions each turn ;1, 99; (10^{-4})
 - Osmotropotaxic Sensitivity - how closely an ant follows the surrounding pheromone trails (the degree of randomness with which an ant follows trails) ;1, 99; (10^{-1})
 - Inverse Sensory Capacity - the maximum level of pheromone an ant will sense, with higher levels being ignored (a desensitizing effect) ;1, 99; (10^{-2})
 - Object Density-based Additional Pheromone Deposit Rate (Optional) - amount of pheromone deposited, dependent on the density of surrounding objects ;1, 99; (10^{-2})
 - Supports - colour objects, with set sizes between 1-15, attribute objects of 1, 10, 100, 1,000 and 10,000 attributes, word objects and document objects

3.7 Method of Evaluation

3.7.1 Evaluate Increasing Complexity

To evaluate the effect of increasing complexity the values of each parameter will be set within the optimal region, determined by the method described in section 3.7.5. The exact values are specified in the (see chapter 4). Using consistent parameter values over the 1, 10, 100, 1,000 and 10,000 attribute objects (each attribute is a randomly assigned integer value between 1 and 1,000) and recording the associated Density, Impurity, Piles and Seconds levels provides a trend comparison as complexity increases. Additionally, if the results, suggest that parameter optimal regions are affected by the changing complexity then a subset of these values will be further evaluated.

3.7.2 Evaluate Document Sorting

To evaluate the ability of the Ant Sort to cluster similar documents and separate dissimilar ones the document sorting will be carried out on the full set of selected texts (see section 3.3). However this will only provide an empirical result, for example if Shakespeares works tend to be clustered together, one could consider that there is a correlation, but this would be unclear. Thus in addition, the selected texts will be internally sorted, each a total of 10 times. This will create 50 different document objects, to which the Ant Sort will be reapplied. Success or failure will be measured in terms of the level of clustering observed between document objects representing the same actual text. This limited test size, 5 x 10, is due to the time constraints for this paper, the limitations of available hardware (processing time) and the limitations of available software.

3.7.3 Evaluate Internal Sorting

The internal sorting process relies on previous works, WordNet [Miller, 2003], QueryData [Rennie, 2000] and through the Similarity [Patwardhan and Pedersen, 2003] package the work of Leacock and Chodorow [Leacock and Chodorow, 1998]. Past research into the Ant Sort algorithm has shown that it is able to sort items which have a distance (similarity) measure [Lumer and Faieta, 1994]. The ability of the aforementioned packages to produce a distance measure between any two words within the WordNet database (words in the texts but not within the database are removed, prior to sorting) shows that word objects are capable of being sorted using these tools. Taking these considerations into account, the effectiveness of the internal word sort has only been considered from the technical viewpoint, i.e. that clusters are formed, that identical/similar words have short distance measures. The overall evaluation of the practicality of the concept of internally sorting more complex objects by simpler internal components will be dependant on the outcome of the evaluation of document sorting (see chapter 4). The internal sorting will be carried out on the first 200 words of each of the selected texts. The resultant clustering of the internal sort will determine the attributes of the document objects used for the document sort process.

3.7.4 Evaluate Generic and Iterative Sorting

The overall success or failure of the document sorting and the levels of clustering observed will provide the basis for evaluating the applicability of the proposed goals. That is, whether complex objects can be sorted iteratively: first internally by attributes (words); and then externally against peers (documents) using a consistent generic approach.

3.7.5 Evaluate Optimal Parameters

In order to evaluate the general trends and the effects of the parameters, we establish a default set of values:

- Object Quantity - 1000
- Ant Quantity - 50
- Ant Sensing Radius - 1
- Ant Step Size - 1
- Chance of Directional Change - 500
- Dimensions - [50x50]
- Stagnation Control (Optional) - 250
- Discrimination (or α) - 250 (10^{-3})
- Pickup Probability Constant - 300 (10^{-3})
- Drop Probability Constant - 300 (10^{-3})
- Pheromone (Optional)
 - Constant Pheromone Deposit Rate - 7 (10^{-2})
 - Constant Pheromone Evaporation Rate - 15 (10^{-4})
 - Osmotropotaxic Sensitivity - 35 (10^{-1})
 - Inverse Sensory Capacity - 20 (10^{-2})
 - Object Density-based Additional Pheromone Deposit Rate (Optional) - 25 (10^{-2})

These baseline figures have been derived from work by Ramos et al [Ramos et al., 2002; Ramos and Merelo, 2002] regarding the pheromone parameters or have been selected as a simplified value. In the cases of ant sensing radius and ant step size the value of 1 is most simple because they require: the minimum awareness of any ants surroundings and the restriction of movement to only adjacent positions, respectively. Chance of directional change and stagnation control (when in use, no pheromones) are chosen as values that will occur many times over the 10,000 turn execution. The smallest dimension size is used for execution speed, with the base object quantity selected filling two fifths of the grid ($1000/(50*50)$). The number of ants used is one twentieth the number of objects, keeping a reasonable ratio between objects and ants but using enough ants to cause significant clustering within 10,000 turns.

Each parameter is independently evaluated by incrementing it over its full set of possible values while all other values remain constant. We consider two different types of parameters, those with fewer than 500 possible values and those with greater than 500 values. For each increment of the sub-500 parameters, (namely: ant sensing radius, ant step size and all of the pheromone parameters) the sorting is run for 10,000 turns (equal to the number of times each ant moves). This is reiterated 20 times to accurately calculate an average value at the given parameter level. For other parameters (greater than 500) the Ant Sort is again run for 10,000 turns but this is only repeated twice. While this method is not expected to produce an accurate average value for the given parameter setting it can be expected that the resultant graph of the trends caused by the changing parameter will be consistent. It should be noted that this thesis only seeks to determine optimal regions for each parameter, not specific, precise optimal values.

To avoid determining optimal regions for parameters whose effectiveness is dependant upon the nature of the object being sorted, the above steps are to be applied to each of: 2 colour object set, 15 colour object set, 1 attribute object set, 10 attribute object set and 100 attribute object set.

Chapter 4

Results

4.1 Effect of Increasing Complexity

The effect of increasing complexity uses the four metrics determined for this thesis, chapter 3. The piles (or clusters) and impurity metrics have been, throughout the course of this experiment, the most accurate and independent of the four (see chapter 5 for details). The density has provided so little useful information regarding the clustering of the ant sort that it has generally been disregarded throughout the results.

The seconds measure is used to provide some indication of the performance of the software execution speed. The results not unexpectedly show steadily increasing execution times with as much as 5-8.5 times longer required to execute as the objects increase tenfold in complexity. The addition of pheromone trails adds approximately 3 seconds per iteration (10,000 steps). This is a significant overhead to the process as objects with less than 100 attributes in complexity require less than 1 second per iteration. This overhead increases with complexity, but becomes a smaller percentage (10-15

Impurity levels have tended to slowly increase with object complexity. At low complexity, impurity tends to remain fairly constant or marginally increases see Figure 4.1 regardless of whether pheromone trails are used or not. However as the complexity increases, and this especially notable with the 15 colour object sets, the introduction of pheromone caused a noticeable impact on the trend of the relationship.

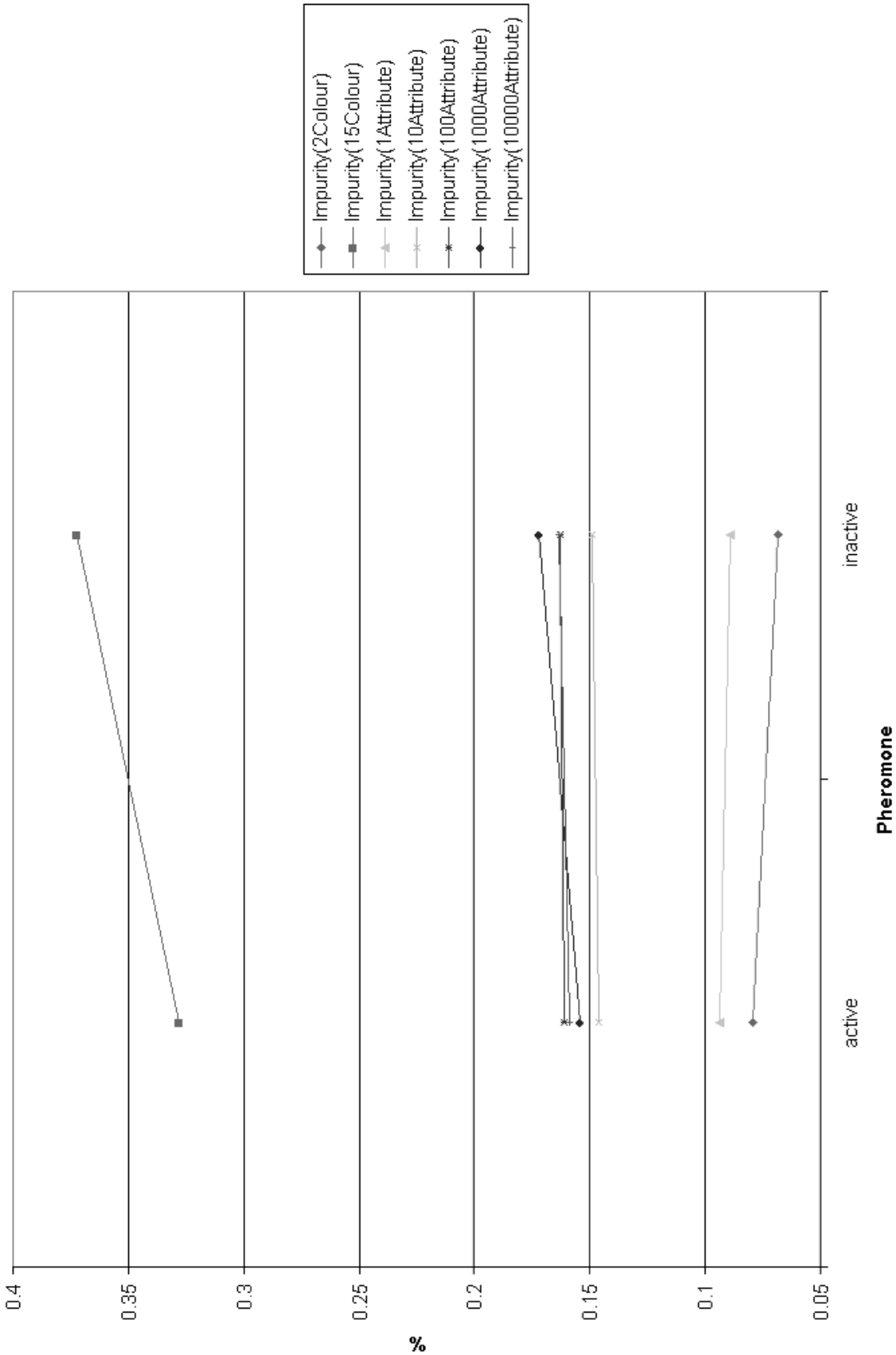


Figure 4.1: Comparison of differing complexities effect on impurity levels

The quantity of piles (or clusters) is the most significant measure of the effectiveness of the Ant Sort. It is most significant of the four metrics because assuming that other metrics are not at extreme ranges then the level of categorisation achieved is dependent on the quantity of piles. An interesting factor here is that increasing object complexity doesn't seem to cause significant difficulty for the ants to cluster, with similar quantities of clusters generated for each complexity in the same number of steps. Also the effect of the pheromone trails can be clearly observed, Figure 4.2 (excepting one outlier) with 2-10

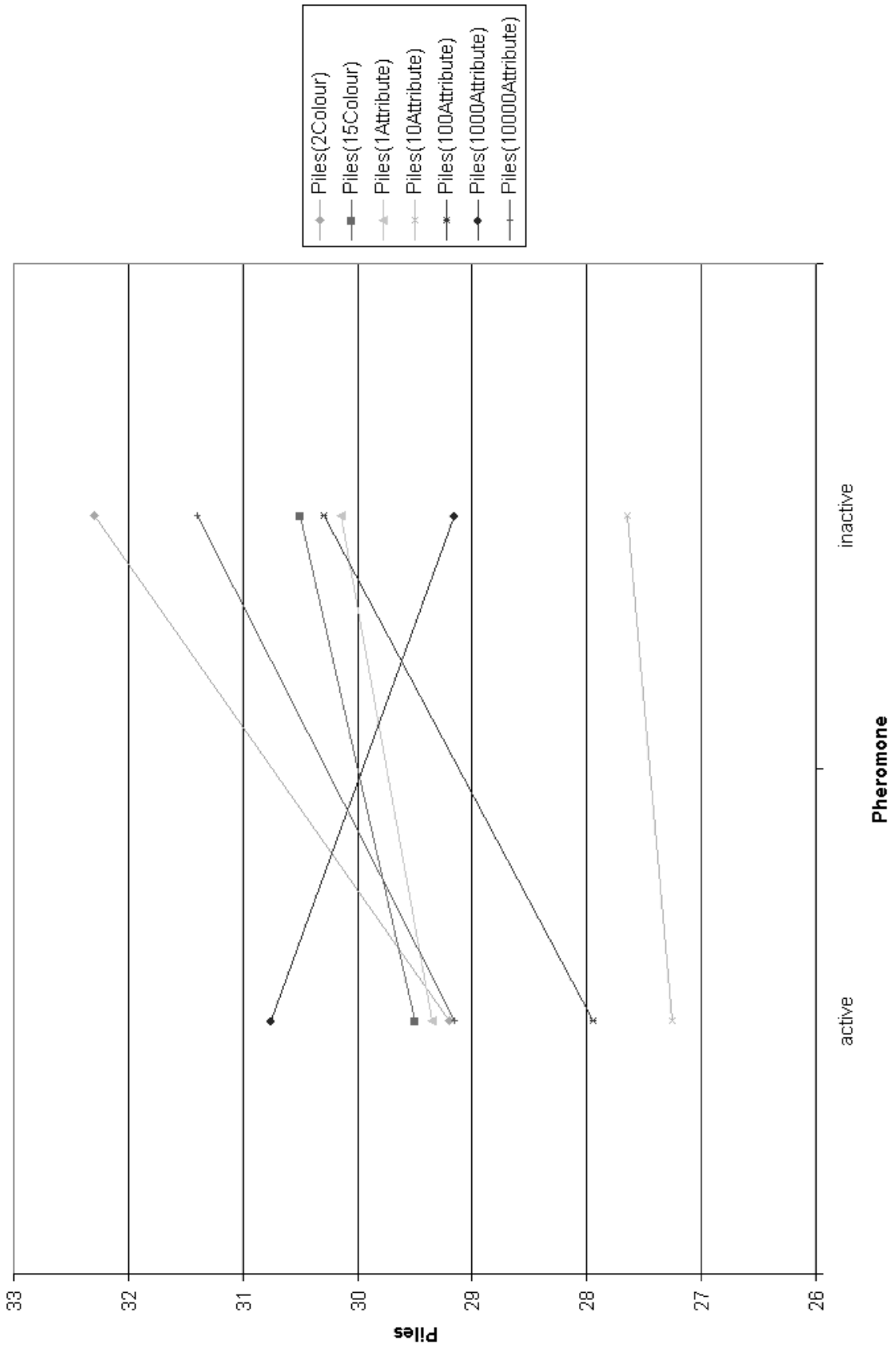


Figure 4.2: Comparison of differing complexities effect on quantities of clusters formed

4.1.1 Optimal Parameters and Values

From the examination (described /refMethods) of each of the ‘Anty’ softwares selected parameters there are a number of factors with have a dramatic effect and others which have a more refined effect on the measured metrics.

The difficulty for each ant to determine an appropriate stopping point is caused by vagaries, for example, is a very fine level of clustering required or is speed more important? These questions are in addition to the problem of determining the overall characteristics of the clustering via purely local information.

The default parameters used in the experimentation (see chapter 3) show that 50,000 iterations are required before the effectiveness of the clustering levels out. This figure is obviously dependent on a number of the default parameters, especially number of ants, grid size and number of objects. Of particular interest is the distinct difference between the high and low complexity objects, with the high complexity causing the impurity level to rise. Whereas over the same period the low complexity objects have a decreasing impurity level see Figure 4.3. The initial clustering of the randomly located objects has, as may be expected a poor level of impurity, and when clustered into smaller piles impurity levels improve. However with more complex objects, see Figure 4.4 as the objects form into fewer clusters, the impurity level rises. This is presumably because the more complex objects are less similar to each other than their simple counterparts.

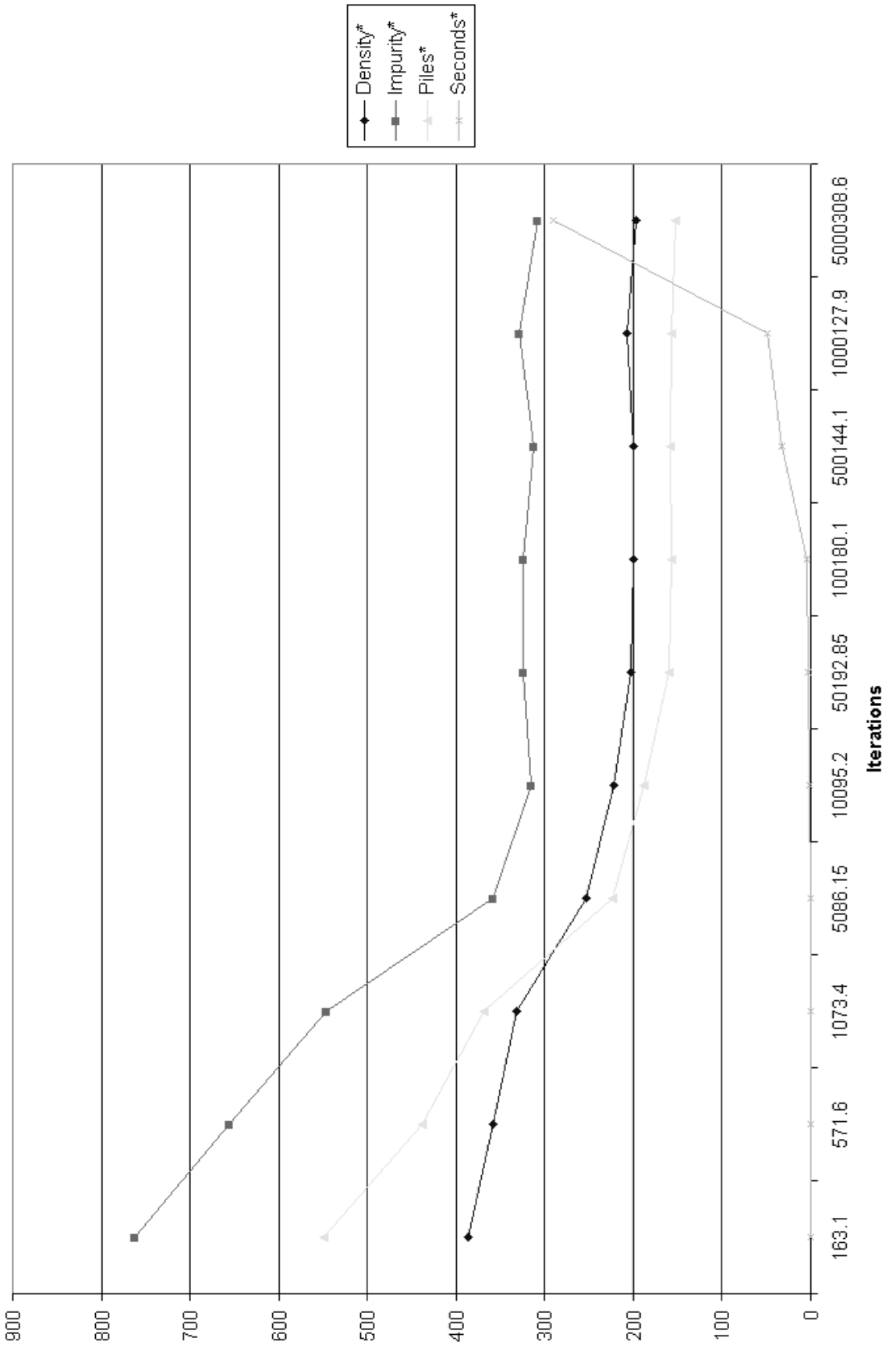


Figure 4.3: Increasing iterations using default for other parameters effect on quantity of clusters (piles), seconds, impurity and density (2-Colour objects). Y-axis values are arbitrary values in order to show the trends of all metrics together

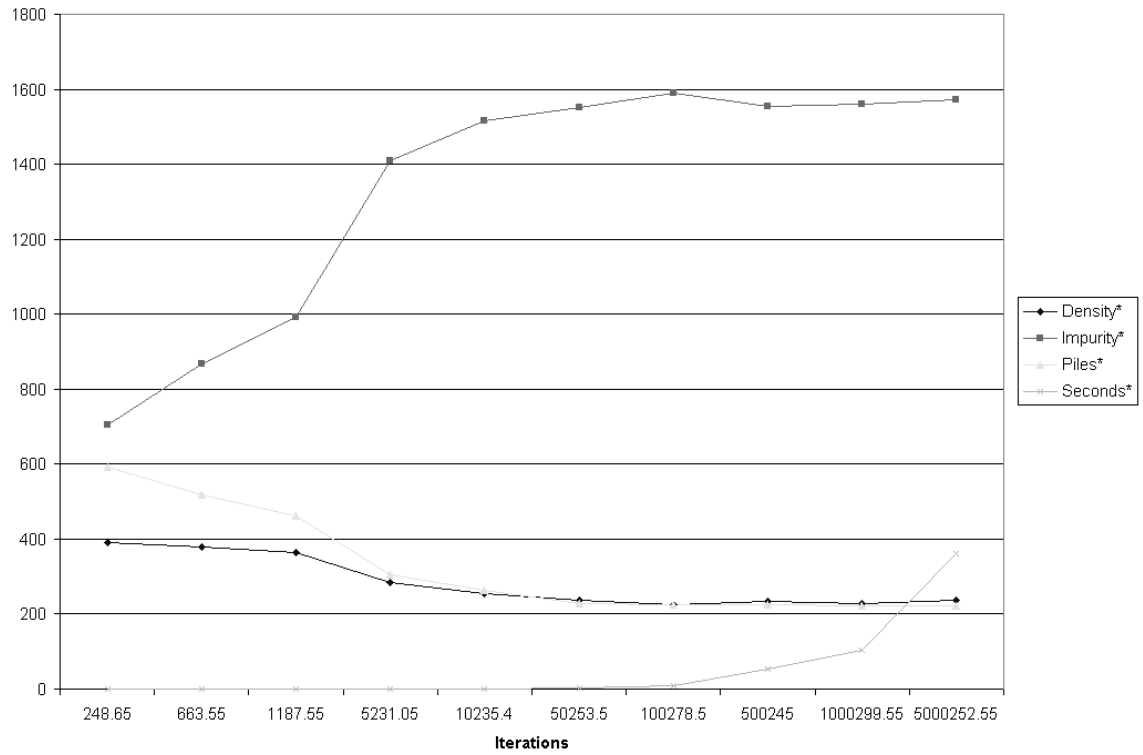


Figure 4.4: Increasing iterations using default for other parameters effect on quantity of clusters (piles), seconds, impurity and density (10 Attribute objects). Y-axis values are arbitrary values in order to show the trends of all metrics together

The perhaps unsurprising effect of the ratio of objects to grid size, Figure 4.5 shows that a significant amount of 'empty grid space is required for effective sorting (within identical time periods). Here we can see that as the number of piles increases, the density decreases, as does the impurity level. Impurity and density minimums and the cluster maximum occur roughly simultaneously at around 550-600 objects or between about 20-25

Figure 4.5: Increasing quantity of objects on a fixed grid size effect on quantity of clusters (2 Colour objects)

Another interesting factor is the number of ants, particular as a ratio to objects. The results show in Figure 4.6 a distinct downwards trend in the number of clusters (after execution) beginning to level out as the number of ants approaches a 1:2 object ratio.

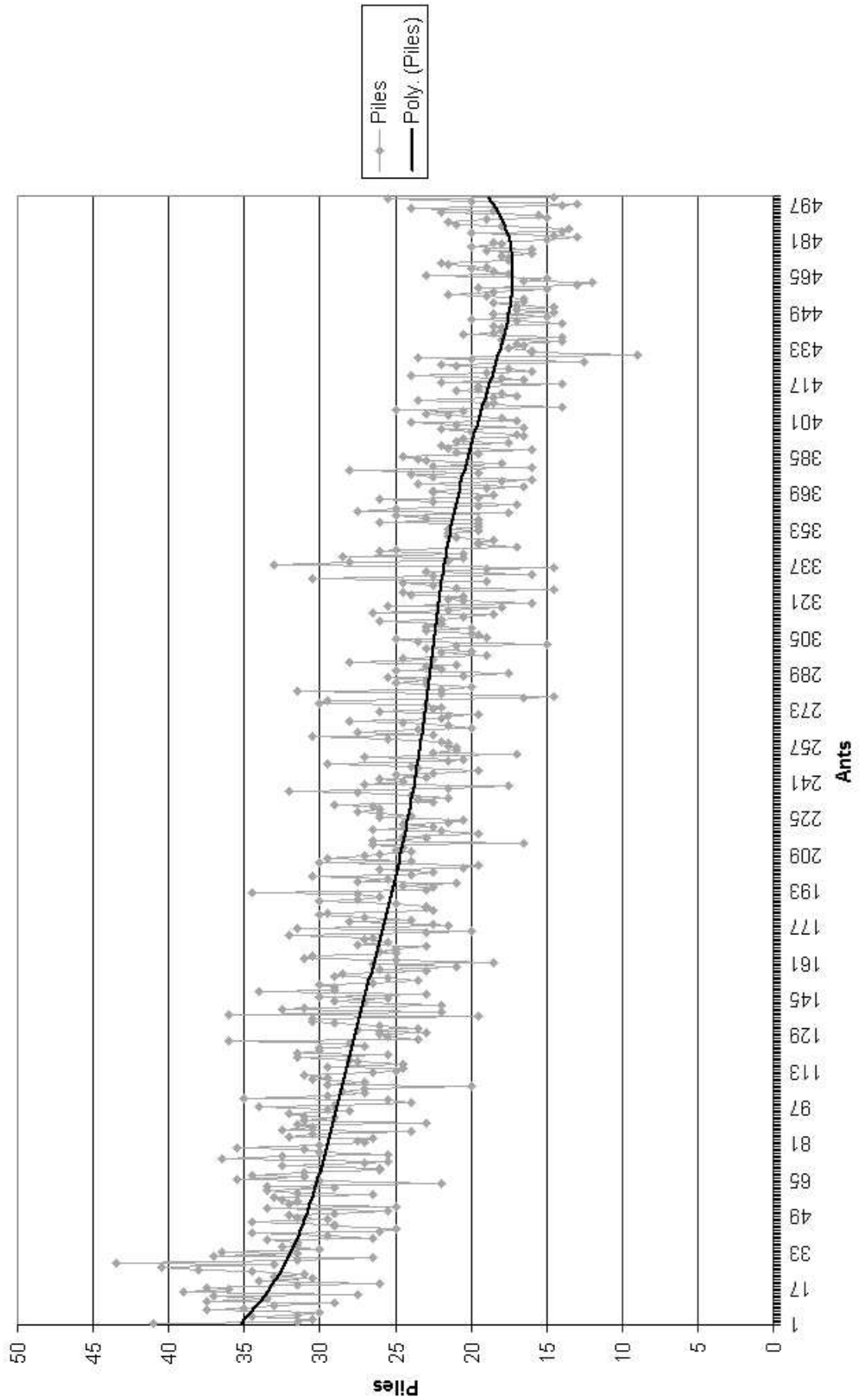


Figure 4.6: Increasing quantity of ants effect on quantity of clusters (2 Colour objects)

The sensing distance of each ant, that is the distance over which an ant senses (and thus compares) surrounding objects, shows that fewest remaining clusters occur when the sensing distance is minimal.

The increase in step size for all ants has an insignificant effect on the metrics taken. This correlates with the suggestions of Handl and Meyer [Handl and Meyer, 2002] refuting the results of Lumer and Faieta [?], however variably increasing step sizes with inhomogeneous populations has not been investigated.

The stagnation control levels (limiting how long an ant searches before dropping an object) show a rapid decrease in the impurity measure (see Figure 4.7) until around the 250-350 turns region. However the impurity level does continue to decrease throughout as the turns given before an ant will drop an object increase. This suggests that any level of stagnation control causes some coarseness in the clusters, in spite of the benefit of the stagnation control freeing up ants earlier. Stagnation control may be more effective given a smaller ant to object ratio and/or over shorter execution steps.

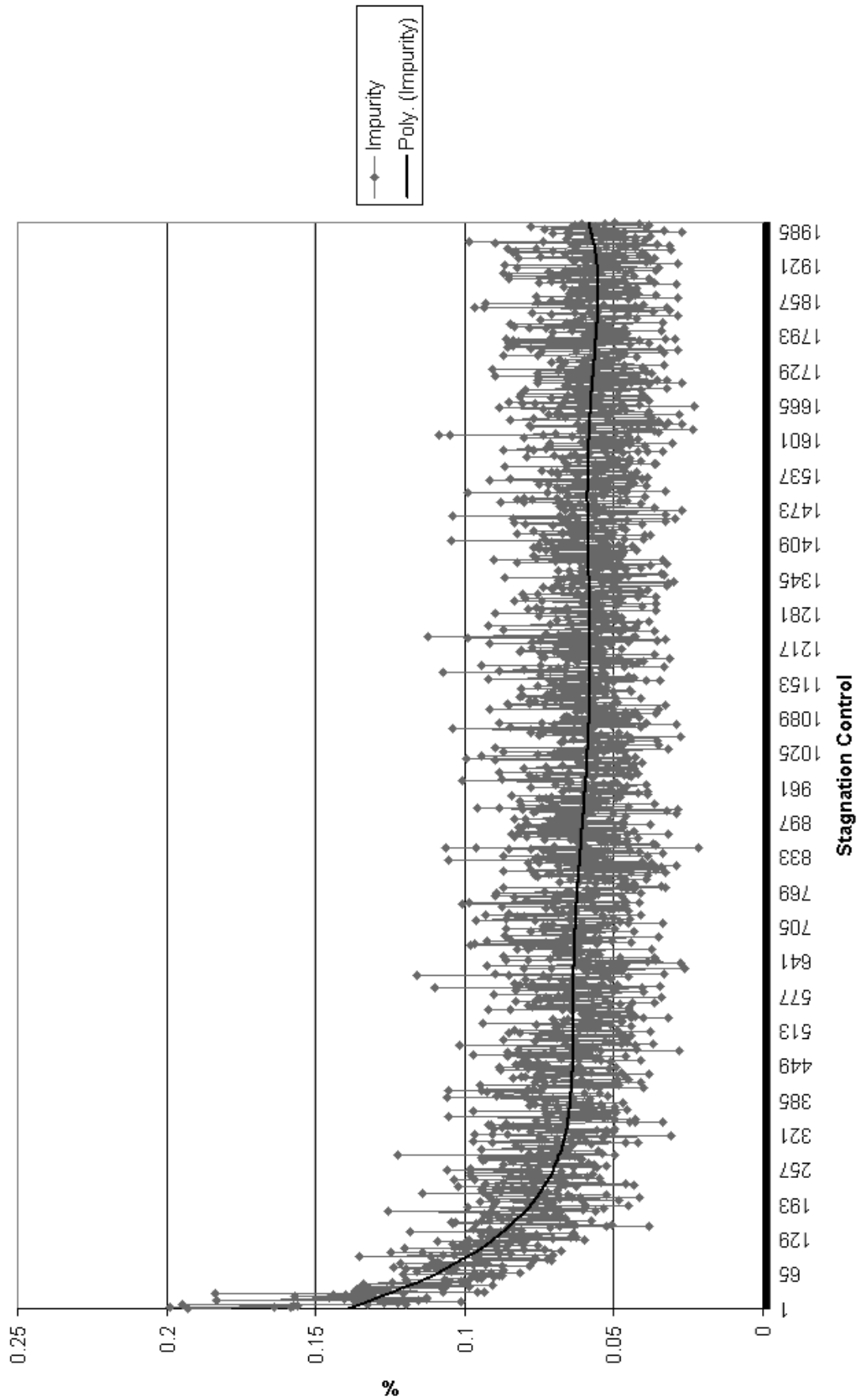


Figure 4.7: Increasing stagnation control effect on the impurity levels of clusters (2 Colour objects)

Directional change frequency is a parameter that is used instead of pheromone trails to provide each ant with some directional consistency. The larger values for this parameter cause each ant to continue in straight lines horizontally or vertically across the grid for long periods. These results show an initial improvement in Figure 4.8 when directional change frequency is used. The number of clusters decreases for values of directional change frequency between 50-150, but this parameter becomes inefficient beyond this. Interestingly this trend reduces as object complexity is increased (see Figure /refDirectional-Change-100Attribute-Figure). Instead, a minimal variation is observed. The effect of directional change frequency would also be more pronounced if only a handful of ants were used rather than the 50 used in these cases. This would occur because few ants would be unable to cover the grid effectively without often changing direction.

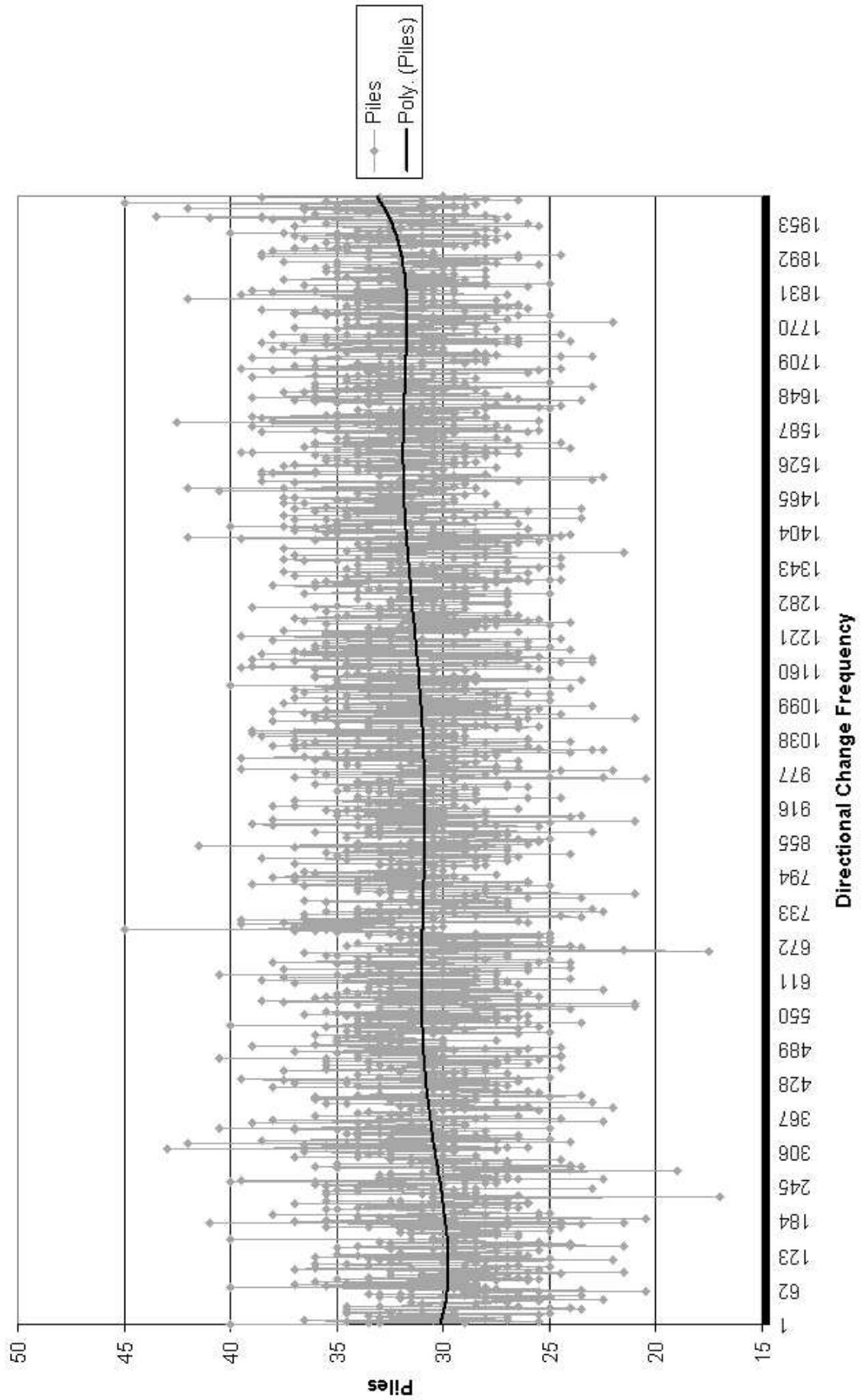


Figure 4.8: Increasing directional change frequency effect on the quantity of clusters (2 Colour objects)

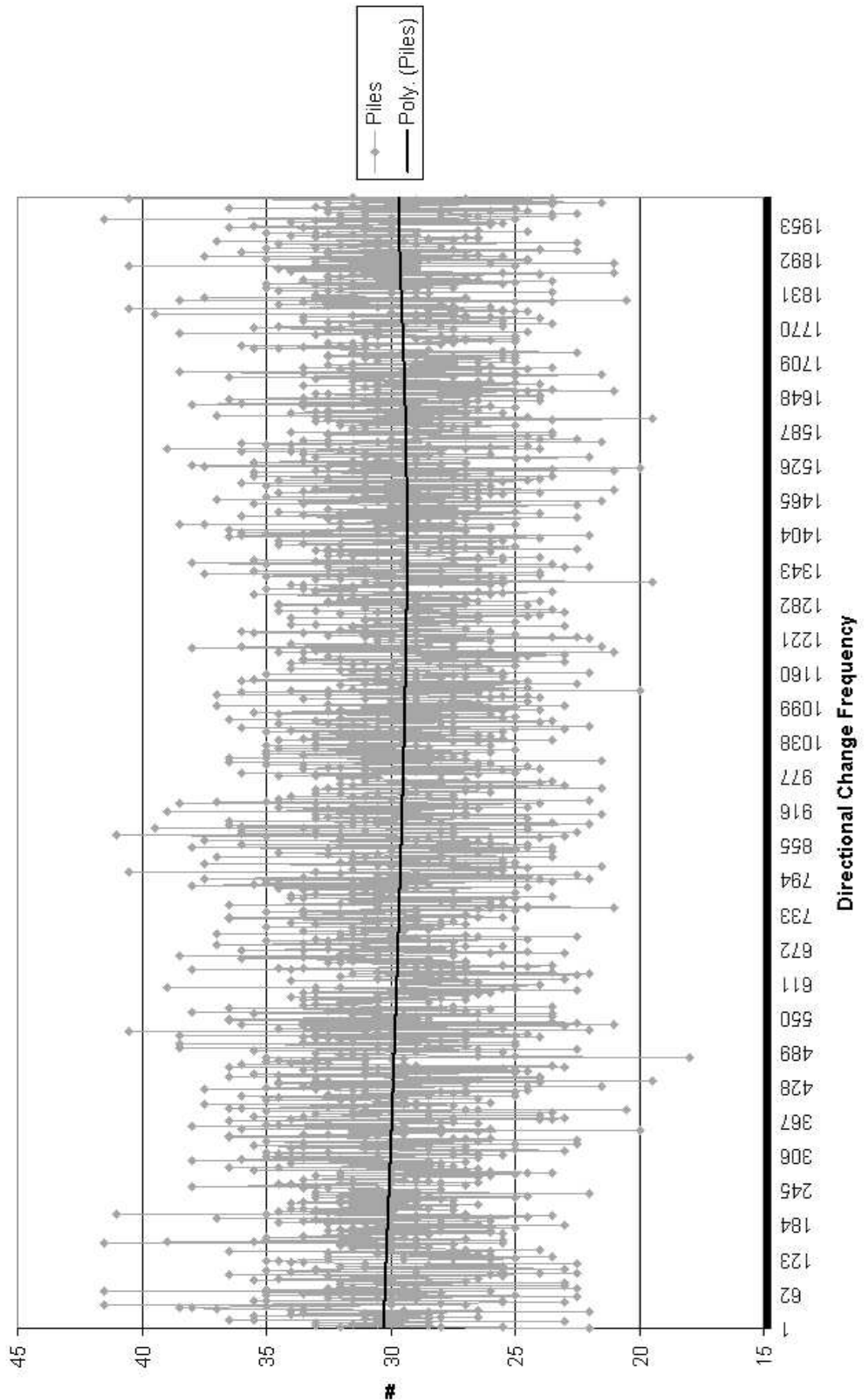


Figure 4.9: Increasing directional change frequency effect on the quantity of clusters (100 Attribute objects)

A key method to retain the distributed capabilities and non-communication approach of the ant sort whilst providing improved performance through environmental conditions is provided by pheromone trails in nature, and the artificial version in these experiments. The effect of pheromone is significant as shown by Figure 4.10 (between 2-10

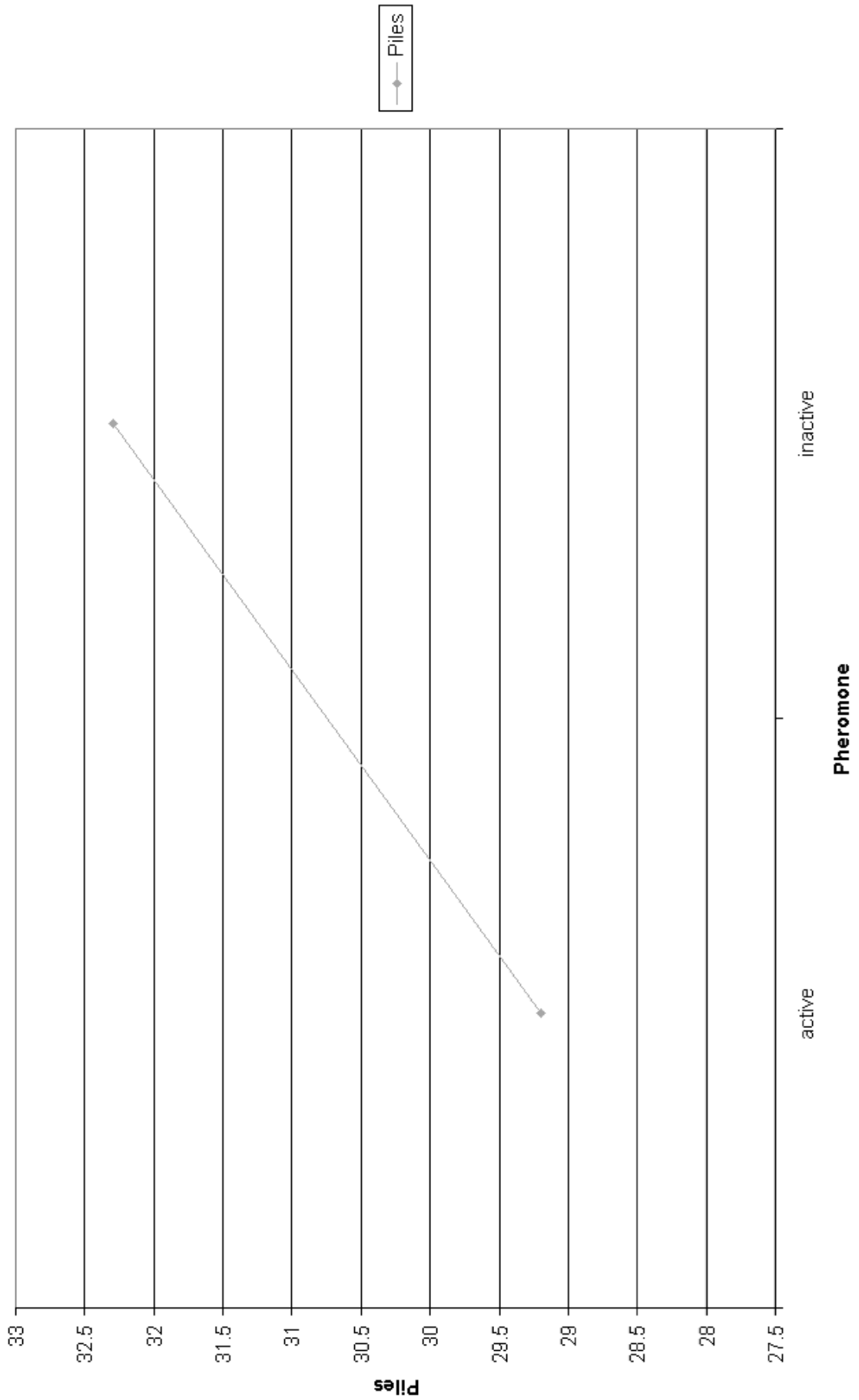


Figure 4.10: The effect of incorporating pheromone trails on the quantity of clusters (2 Colour objects)

Unfortunately the various measurements taken of the pheromone parameters, namely deposit, evaporation, osmotropotaxic sensitivity, sensory capacity and the effect of basing the amount of pheromone to deposit on surrounding objects, had too little effect as individual components to be worth further analysis.

4.2 Generic and Iterative Sorting

4.2.1 Internal Sorting

From the internal or word sort, 2 words are selected from each of the ‘top’ 5 clusters. Note that significant further experimentation is required to make these arbitrarily selected extraction methods effective in extracting the intended near unique defining characteristics of the word sort.

Some of the results of the word sort are encouraging, suggesting that there are some reasons for further experimentation. Shown below are some of the top cluster words from a series of different Ant Sort’s run on the ‘Angel of the Odd, The’ text by Poe (chapter 7 for full detail) () represents the number of occurrences:

1. not(5), strongly(5), by(3), november(1), then(2), great(2), singular(2), dinner(1), but(4), away(5)
2. is, a(5), just(4), all(1), but(4), strongly(5), up(3), confess(1), unusually(4), miscellaneous(2)
3. a(5), alone(2), important(2), lafitte(2), up(3), small(1), least(2), chilly(2), a(5), hearty(1)
4. up(3), not(5), an(2), dyspeptic(1), unusually(4), about(4), through(2), just(4), pilgrimage(3), numerous(1)
5. pilgrimage(3), just(4), somewhat(3), criticize(2), chilly(2), alone(2), away(5), unusually(4), some(4), not(5)
6. strongly(5), then(2), a(5), some(4), about(4), strange(4), carefully(1), puff(1), but(4), some(4)
7. extravaganza(1), a(5), criticize(2), two(1), about(4), great(2), important(2), strange(4), strongly(5), least(2)
8. unusually(4), some(4), was(1), by(3), by(3), away(5), somewhat(3), lafitte(2), not(5), happy(2)
9. through(2), away(5), attacked, somewhat(3), dessert, an(2), strange(4), miscellaneous(2), about(4), strongly(5)

10. singular(2), but(4), follows(1), happy(2), just(4), pilgrimage(3), strange(4), away(5), when(1), not(5)

These word lists do show a number of frequently appearing words, especially when some of the more meaningless words are removed (see common words, chapter 5).

4.2.2 Document Sorting

An example of the resultant clusters of the document sort is shown below. Very little correlation can be seen, mostly due to the aforementioned issues with the internal sorting and the extraction of useful information from that phase.

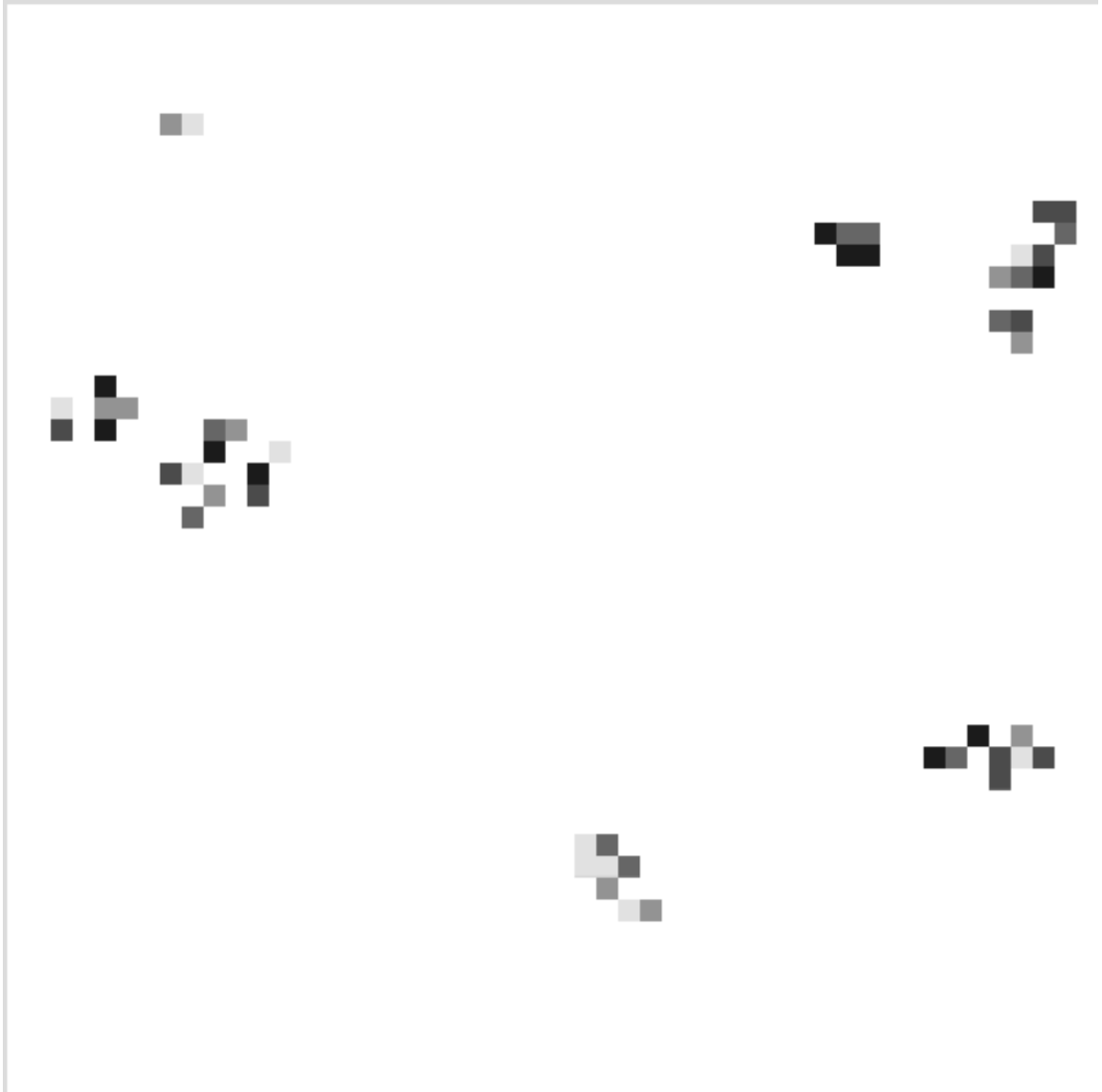


Figure 4.11: The partially accurate clustering of 50 different derived document objects; each shade represents 1 of 5 actual texts

We can see that some of the document objects (1 actual document for each colour, with each object one of 10 variations of each document) do tend to be located adjacent to one another. However, it can be seen that there is also significant impurity in all of the major clusters.

Chapter 5

Discussion/Conclusions

The results of this thesis provide a basis for more extensive research into this area and include substantial work on the development of the Ant Sort metrics program, 'Anty'. This may give rise to an opportunity for further development to underwrite more research into this area.

5.1 Effect of Increasing Complexity and Optimal Parameters and Values

The results section (see chapter 4) shows a number of interesting factors of the Ant Sorts response to increasingly complex objects. These factors include:

- Stagnation Control
- Clusters Formed
- Impurity Increase
- Pheromone Trails

These results suggest that the benefit of the stagnation control, that is the earlier 'freeing of an ant from a long carried object, is actually less than the 'cost of not continuing efforts to find an appropriate location for the ant.

The results show that the number of clusters formed by the Ant Sort algorithm in a given number of 'turns remains consistent in spite of increasing complexity. The cause of this is most probably that Ant Sort will operate effectively given that there are appropriate similarity measures, indifferent to the changes in complexity.

The impurity metric increase occurs with increasing complexity, however this is likely to be in response to the increasing variability of more complex objects. This same factor is also likely to be responsible for the interesting opposite trends of the simple objects to the more complex as observed in Figure 4.3 and Figure 4.4.

Pheromone trails provide a useful method to improve performance. The results from this analysis support those of previous works (especially Ramos et al [Ramos and Merelo, 2002; Ramos et al., 2002]). The results show improved clustering of between 2-10

5.2 Generic and Iterative Sort

Overall this work has provided a clear direction for a new possible method to be used by the Ant Sort on complex objects. Also an admittedly limited study of the idea has been tested providing uncertain results, leading to the suggestions for future research, chapter 6. The following two subsections detail the conclusions drawn from the results of the document and word sorting as an attempt to demonstrate the generic and iterative concept.

5.2.1 Internal Sort

The results of the internal sort have limited relevance, being almost purely a step in the progression to the overall document sort. However the results of the admittedly small-scale experimentation and the ‘top 5 cluster word lists (see 7) do seem to show some very limited correlation. This suggests that with further refinement, especially the removal of common words (‘I, ‘about, ‘is, ‘and, etc.) and improved word cluster to document object data extraction, better results could be achieved for the generic and iterative Ant Sort approach.

5.2.2 Document Sort

The very minor correlation between documents and clusters, shown for the results of the overall document sort, chapter 4, does cast some doubt on the viability of the generic and iterative concept. Due to the limited scope of the experimentation performed as part of this thesis, a more final determination of the success or failure of the concept will require further research.

There are numerous alternative methods that could be used to extend and modify the evaluation of the document sort. However this thesis has attempted to use the document sort in order to validate the possibility of a generic and iterative approach to the Ant Sort algorithm. Therefore we have not looked at the effects of: larger (or smaller) than the 200 word samples from each text; experimentation over significantly larger datasets; use of

different value and mechanisms to extract the most relevant data from the internal word sort; and removal of common words. These are noted weaknesses of the existing work and should be considered as possible extensions for further research, on the basis of the concepts opened up in this thesis.

Chapter 6

Future Work

A key purpose that motivated this research was to attempt to approach the Ant Sort algorithms application to complex object sorting from a different direction. This has not led to a convincing conclusion (see chapter 5) either way, leaving further experimentation necessary to resolve these questions.

This different direction has by no means been explored in depth and is merely an attempt to authenticate the possibility of the generic and iterative approach. Some factors and additional questions related to the document sort that still require (further) examination are listed here:

- Variation of the sample size used from each text. Does a larger sample reduce the impact of individual words? Does it distinguish individual documents more clearly?
- Examination of the effect of the parameters of the Ant Sort (especially the number of turns to sort) on the final clustering.
- Inclusion of context sensitive data within the word sort or attempting to cluster similar sentences. Latent Semantic Analysis (LSA) techniques would be one possibility.
- Further tests of the existing work over much larger sets of texts.
- Using different information and mechanisms to convert the results of the clustered word objects into document objects. Much improved extraction of relevant data could be achieved.
- Extending the removal of ‘common words that detracted from the relevance of results, for example: ‘I, ‘at, ‘by, ‘an etc.
- Using known categorised documents for evaluating the Ant Sorts conformance.

This list of possible future work recognises the limitations of this presented work, especially in regard to larger scale (larger pool of texts) sorting scenarios. The field of semantic similarity should be used more extensively in conjunction with future research into this area.

Chapter 7

Appendix - Text Documents Used

1. Shakespeare
 - (a) Hamlet
2. Poe
 - (a) Angel Of The Odd, The
 - (b) Assignment, The
 - (c) Murders In The Rue Morgue, The
 - (d) Mystery Of Marie Roget, The

Chapter 8

Appendix - Extended Word Clustering Results

1. Hamlet

- (a) once,all,barn,most,not,art,friends,much,westward,most
- (b) dane,barn,all,when,two,who,primus,me,most,westward
- (c) twice,where,holla,made,a,thou,but,two,not,prima
- (d) me,nay,thou,where,not,minutes,most,not,thou,who
- (e) carefully,together,all,not,therefore,not,not,warlike,awhile,westward
- (f) so,all,dane,not,prima,primus,two,not,o,me
- (g) two,me,together,two,warlike,tis,harrows,barn,carefully,therefore
- (h) mar,now,not,not,not,carefully,sometimes,once,together,not
- (i) not,honest,when,apparition,thou,primus,same,sometimes,me,along
- (j) by,fortified,there,sometimes,same,where,whos,me,honest,illuminate

2. Angel Of The Odd, The

- (a) not,strongly,by,november,then,great,singular,dinner,but,away
- (b) is,a,just,all,but,strongly,up,confess,unusually,miscellaneous
- (c) a,alone,important,lafitte,up,small,least,chilly,a,hearty
- (d) up,not,an,dyspeptic,unusually,about,through,just,pilgrimage,numerous
- (e) pilgrimage,just,somewhat,criticise,chilly,alone,away,unusually,some,not
- (f) strongly,then,a,some,about,strange,carefully,puff,but,some
- (g) extravaganza,a,criticise,two,about,great,important,strange,strongly,least
- (h) unusually,some,was,by,by,away,somewhat,lafitte,not,happy

- (i) through,away,attacked,somewhat,desert,an,strange,miscellaneous,about,strongly
- (j) singular,but,follows,happy,just,pilgrimage,strange,away,when,not

3. Assignation, The

- (a) me,away,as,on,how,a,midnight,yet,magnificent,once
- (b) ducal,on,venice,met,brilliance,henry,away,other,who,silent
- (c) not,but,who,own,thou,ducal,sounded,other,yet,how
- (d) gloom,wife,mysterious,away,once,wide,but,but,unusual,valley
- (e) how,a,great,away,by,before,who,silent,unusual,not
- (f) great,unusual,who,but,thou,magnificent,once,again,away,yet
- (g) who,but,palladian,magnificent,away,everlasting,other,unusual,hour,by
- (h) imagination,thou,fourth,wide,other,but,away,or,mysterious,magnificent
- (i) sophist,unusual,or,other,multitude,person,king,great,palladian,yet
- (j) not,again,vale,archway,behold,but,beneath,away,who,unusual

4. Murders In The Rue Morgue, The

- (a) most,random,much,possibly,unjustly,exults,appreciate,retrograde,on,but
- (b) sir,activity,such,strong,assumed,not,not,peculiar,highest,each
- (c) by,yet,but,not,other,inordinately,unjustly,sang,higher,beyond
- (d) liveliest,trivial,very,analytical,strong,but,not,such,by,each
- (e) by,in,higher,merely,on,all,discoursed,faculty,very,when
- (f) mental,but,strong,by,such,by,although,mathematical,not,decidedly
- (g) more,possibly,exhibiting,women,about,analytical,analysis,narrative,trivial,not
- (h) physical,by,simply,peculiar,greatly,inordinately,therefore,most,susceptible,but
- (i) possibly,has,unjustly,mental,only,most,highest,especially,higher,when
- (j) highest,usefully,analyst,in,does,more,but,about,very,most

5. Mystery Of Marie Roget, The

- (a) der,marvellous,thoroughly,when,roget,recognized,by,new,very,occasionally
- (b) der,intelligible,mathematical,about,extraordinary,mental,remarkable,such,by,public
- (c) never,depict,der,such,extraordinary,about,seldom,ago,new,intelligible
- (d) thrilling,coincide,generally,by,by,entitled,extraordinary,stified,vague,der
- (e) readers,ideal,das,generally,late,vague,yet,der,equally,such
- (f) by,such,now,instead,equally,das,reformation,dass,calmest,such
- (g) marvellous,extraordinary,der,ago,calmest,yet,seldom,rigidly,der,vague

- (h) an,all,stifled,seemingly,occasionally,not,coincidences,real,mathematical,unable
- (i) in,mathematical,who,called,depict,marvellous,when,instead,most,equally
- (j) most,seldom,in,purely,full,so,vague,sentiments,seems,new

References

- Agrawal, R., Gehrke, J., Gunopulos, D. and Raghavan, P. [1998]. Automatic subspace clustering of high dimensional data for data mining applications, *ACM SIGMOD Conference*, pp. 94–105.
- Back, T., Hammel, U. and Schwefel, H. P. [1997]. Evolutionary computation: Comments on the history and current state, *IEEE Transactions on Evolutionary Computation*, Vol. 1, pp. 3–17.
- Bradley, P., Fayyad, U. and Reina, C. [1998]. Scaling clustering algorithms to large databases, *ACM KDD Conference*.
- Bremermann, H. J. [1962]. Optimization through evolution and recombination, in M. C. Y. et al. (ed.), *Self-Organizing Systems*, Spartan, Washington, DC.
- Budanitsky, A. and Hirst, G. [2001]. Semantic distance in wordnet: An experimental, application-oriented evaluation of five measures, *Workshop on WordNet and Other Lexical Resources, Second meeting of the North American Chapter of the Association for Computational Linguistics*, Pittsburgh, PA, USA.
- Caro, G. D. and Dorigo, M. [1997]. Antnet: A mobile agents approach to adaptive routing, *Technical report*, IRIDIA, Universit Libre de Bruxelles.
- Chalvo, D. R. and Millonas, M. M. [1995]. How swarms build cognitive maps, in L. Steels (ed.), *The Biology and Technology of Intelligent Autonomous Agents.*, Vol. (144) of *NATO ASI Series*.
- Chretien, L. [1996a]. PhD thesis, Universit Libre de Bruxelles.
- Chretien, L. [1996b]. *Organisation Spatiale du Materiel Provenant de lexcavation du nid chez Messor Barbarus et des Cadavres douvrieries chez Lasius niger (Hymenopterae: Formicidae)*, PhD thesis, Universite Libre de Bruxelles.
- Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K. and Harshman, R. [1990]. Indexing by latent semantic analysis, *Journal of the American Society for Information Science* **41(6)**.

- Deneubourg, J. L., Goss, S., Franks, N., Sendova-Franks, A., Detrain, C. and Chretien, L. [1991]. The dynamic of collective sorting robot-like ants and ant-like robots, *in* J. A. Meyer and S. W. Wilson (eds), *SAB 90 - 1st Conference On Simulation of Adaptive Behavior: From Animals to Animats*, MIT Press.
- Dorigo, M. and Caro, G. D. [1999]. The ant colony optimization meta-heuristic, *in* D. Corne, M. Dorigo and F. Glover (eds), *New Ideas in Optimization*, McGraw-Hill, London, UK.
- Dorigo, M. and Gambardella, L. M. [1997]. Ant colonies for the traveling salesman problem, *Biosystems* **43**.
- Doty, K. L. and Aken, R. E. V. [1993]. Swarm robot materials handling paradigm for a manufacturing workcell, *Proceedings of the 1993 IEEE International Conference on Robotics and Automation*, Vol. 1, IEEE Computer Society Press, Atlanta, Georgia, USA.
- Duda, R. and Hart, P. [1973]. Pattern classification and scene analysis, J. Wiley and Sons.
- Friedberg, R. M. [1958]. A learning machine: Part i, *IBM J.* **2**.
- Fukuda, T., Funato, D., Sekiyama, K. and Arai, F. [1998]. Evaluation on flexibility of swarm intelligent system, *Proceedings of the 1998 IEEE International Conference on Robotics and Automation*, IEEE Press, Leuven, Belgium.
- Gambardella, L. M., Taillard, E. D. and Dorigo, M. [1997]. Ant colonies for qap, *Technical report*, IDSIA, Lugano, Switzerland.
- Goss, S., Beckers, R., Deneubourg, J. L., Aron, S. and Pasteels, J. M. [1990]. How trail laying and trail following can solve foraging problems for ant colonies, *Behavioural Mechanisms of Food Selection* **G 20**.
- Grasse, P.-P. [1984]. Termitologia, *Fondation des Societes*, Tome II, Masson, Paris.
- Handl, J. and Meyer, B. [2002]. Improved ant-based clustering and sorting in a document retrieval interface, *PPSN VII, LNCS* **2439**.
- Hauser, L. [1993]. *Searle's Chinese Box: The Chinese Room Argument and Artificial Intelligence*, PhD thesis, Michigan State University.
- Hauser, L. [1997]. Searle's chinese box: Debunking the chinese room argument, *Minds and Machines* **7**.
- Hirst, G. and St-Onge, D. [1998]. Lexical chains as representations of context for the detection and correction of malapropisms, *Fellbaum* .
- Hoe, K. M., Lai, W. and Tai, T. S. Y. [2002]. Homogeneous ants for web document similarity modeling and categorization, *Ant Algorithms 2002*.
URL: <http://www.informatik.uni-trier.de/ley/db/conf/antalg/antalg2002.html>

- Jiang, J. and Conrath, D. [1997]. Semantic similarity based on corpus statistics and lexical taxonomy, *Proceedings of International Conference on Research in Computational Linguistics*, Taiwan.
- Kuntz, P. and Syners, D. [1994]. Emergent colonization and graph partitioning, *3rd International Conference on Simulation of Adaptive Behaviour: From Animals to Animats 3*, MIT Press.
- Kuntz, P., Syners, D. and Layzell, P. [1998]. A stochastic heuristic for visualising graph clusters in a bi-dimensional space prior to partitioning, *Journal of Heuristics* .
- LCNC [n.d].
URL: <http://www.lcnc.nl>
- Leacock, C. and Chodorow, M. [1998]. Combining local context and wordnet similarity for word sense identification, *Fellbaum* .
- Li, L., Martinoli, A. and Abu-Mostafa, Y. S. [2002]. Emergent specialization in swarm systems, *Intelligent Data Engineering and Automated Learning* .
- Lin, D. [1998]. An information-theoretic definition of similarity, *Proceedings of the 15th International Conference on Machine Learning*.
- Lumer, E. D. and Faieta, B. [1994]. Diversity and adaptation in populations of clustering ants, in D. Cliff, P. Husbands, J. Meyer and S. Wilson (eds), *From Animals to Animats 3, Proceedings of the 3rd International Conference on the Simulation of Adaptive Behavior*, The MIT press/Bradford Books.
- Miller, G. A. [2003]. Wordnet 1.7.1.
URL: <http://www.cogsci.princeton.edu/wn/index.shtml>
- Millonas, M. M. [1992]. A connectionist-type model of self-organised foraging and emergent behaviour in ant swarms, *Journal of Theoretical Biology* **159**.
- Millonas, M. M. [1994]. Swarms, phase transitions, and collective intelligence, *Artificial Life III XVII*.
- Monmarche, N. [1999]. On data clustering with artificial ants, *Data Mining with Evolutionary Algorithms: Research Directions* .
- Ordonez, C. [2003]. Clustering binary data streams with k-means, *ACM DMKD 2003*.
- Parpinelli, R. S., Lopes, H. S. and Freitas, A. A. [2002]. An ant colony algorithm for classification rule discovery, in H. A. Abbass, R. A. Sarker and C. S. Newton (eds), *Data Mining: a Heuristic Approach*, Idea Group Publishing.
- Patwardhan, S. and Pedersen, T. [2003]. Wordnet::similarity, version 0.05.
URL: <http://search.cpan.org/src/SID/WordNet-Similarity=0.05>

- Perez-Uribe, A. [n.d.]. Collective and swarm intelligence in natural and artificial systems.
URL: http://www.geocities.com/fastiland/swarm_course.pdf.gz
- Quesada, J. F., Merelo, J. J., Castellano, J. G., Garcia, M. and Cillero, M. [2001]. Personalization as disambiguation: Lsamercury, a information filtering engine based on latent semantic analysis.
- Ramos, V. and Merelo, J. J. [2002]. Self-organized stigmergic document maps: Environment as a mechanism for context learning.
- Ramos, V., Muge, F. and Pina, P. [2002]. Self-organized data and image retrieval as a consequence of inter-dynamic synergistic relationships in artificial ant colonies, Vol. 87, IOS Press.
- Rennie, J. [2000]. Wordnet::querydata, a perl module for accessing the wordnet database.
URL: <http://www.ai.mit.edu/~jrennie/WordNet/WordNet-QueryData-1.28.readme>
- Resnik, P. [1995]. Using information content to evaluate semantic similarity.
- Searle, J. [1980]. Minds, brains, and programs, *Behavioral and Brain Sciences* **3**.
- Turing, A. M. [1948]. Intelligent machinery: A report, *Technical report*.
URL: http://www.alanturing.net/turing_archive/archive/index/aceindex.htm
- Yang, Y., Slattery, S. and Ghani, R. [2002]. A study of approaches to hypertext categorization, *Journal of Intelligent Information Systems* **18**.
- Zhang, R. and Rudnicky, A. I. [2002]. Improve latent semantic analysis based language model by integrating multiple level knowledge, *Proceedings of ICSLP*.
URL: <http://www.speech.cs.cmu.edu/Communicator/papers/lsalm.pdf>