

# Cemetery Formation

*An investigation of the effects of:*

Speed of ants and the Number of ants in relation to data set

*On two variants of short term memory in ants*



By Stuart Reid

10026942

[stuartgordonreid@gmail.com](mailto:stuartgordonreid@gmail.com)

Department of Computer Science

Faculty of Engineering, Built Environment and Information Technology

University of Pretoria

19 April 2013

## A. Table of contents

### [A. Table of contents](#)

#### [1. Introduction](#)

#### [2. Background](#)

#### [3. Implementation](#)

##### [3.1 UML Diagram](#)

##### [3.2 Class Descriptions](#)

#### [4. Observations](#)

##### [4.1 Controls](#)

[4.1.1 CLUSTERS - The average total number of clusters formed \(maximize\)](#)

[4.1.2 TOTAL - The average total number of items in clusters \(maximize\)](#)

[4.1.3 INTRA - The average intra cluster distances across all clusters \(minimize\)](#)

[4.1.4 INTER - The average inter cluster distances between all clusters \(maximize\)](#)

##### [4.2 Experiment i - effects of the, Ants : Items, ratio](#)

[4.3.1 Memory buffer - 0](#)

[4.3.1 Memory buffer - 1](#)

[4.3.1 Memory buffer - 10](#)

[4.3.1 Memory buffer - 100](#)

##### [4.3 Experiment ii - effects of the speed of the ants](#)

[4.3.1 Memory buffer - 0](#)

[4.3.1 Memory buffer - 1](#)

[4.3.1 Memory buffer - 10](#)

[4.3.1 Memory buffer - 100](#)

#### [5. Conclusions](#)

### [B. Visual Observations \(Appendix\)](#)

## 1. Introduction

The assignment required an investigation of the effects of the speed of Ants and the number of ants in relation to the data set on the effectiveness of the use of two variations of short term memory used in cemetery formation ant algorithms. The number of ants in relation to the data set are hereafter referred to as 'the tuning parameters of the clustering model used by the ants.

A high level overview of my approach to this investigation involved:

1. Implementing a generic ant algorithm with no memory followed by the two variations of the ant algorithm that use short term memory
2. Defining a set of controls to test the performance of the algorithm against
3. Setting up and conducting a number of experiments that would show the effects of the tuning parameters on the performance of the ant algorithms using short term memory
4. Observing the performance of each algorithm during the experiments and
5. Formulating a set of conclusions supported by those observations.

The implementation of the algorithm was done in Java using full Object Orientation. The implementation included a generic ant algorithm, an ant algorithm where newly picked up items are compared directly with the items in the ant's memory and an ant algorithm where newly picked up items are compared with items around the location of an item in memory. The implementation also included the following controls: number of clusters formed, average size of clusters formed, a measure of intra-cluster distances and a measure of inter-cluster distances.

Experiments conducted were done for the following memory buffer sizes: 0, 1, 10 and 100. For each memory buffer size the performance of ant algorithm variant one and ant algorithm variant two were tested with ant to item ratio of 1:3, 1:6, 1:9, 1:12, 1:15, 1:18 and 1:21 as well as for ant speeds of 1, 2, 3 and 4. Each experiment was simulated 75 times each in order to achieve statistical relevance and the results detailed below are the averages of the controls across all 75 simulations. The controls included a measure of the number of clusters, the total number of items contained in clusters, the average intra-cluster distances and the average inter-cluster distances.

## 2. Background

Extract taken from assignment brief:

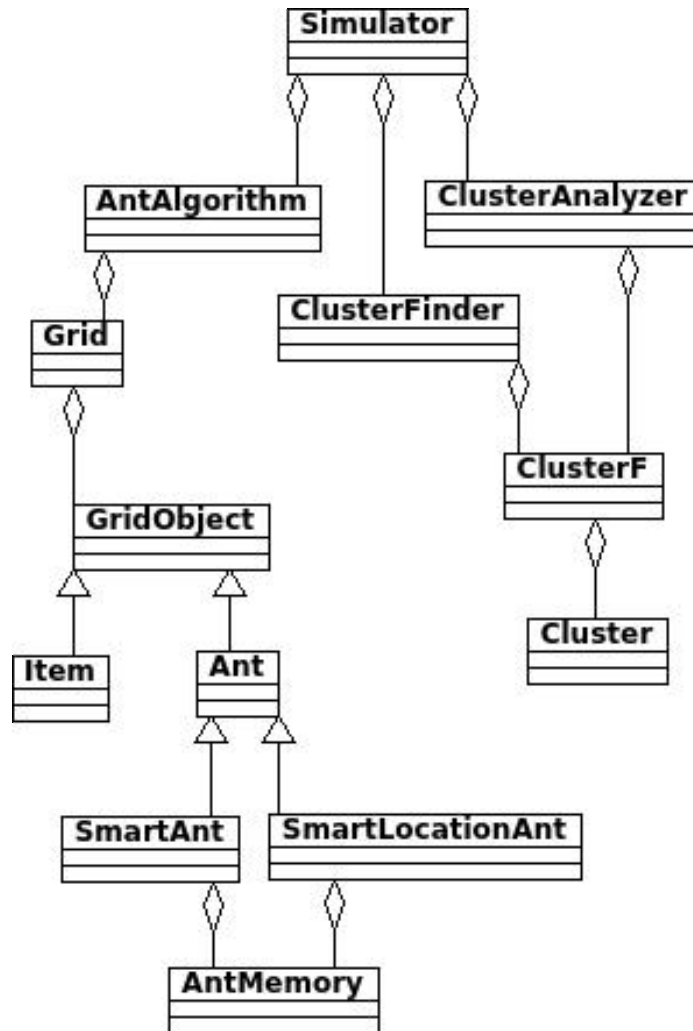
“ *Short-term memory was shown by Lumer and Faieta to affect the quality of clustering with their model of cemetery formation. However, both the speed of ants and the number of ants in relation to the number of data items will impact the accuracy of an ant's memory. In other words, an ant's memory of dropped items becomes stale sooner when there are many ants moving many items as well as when there are fast ants present and likely to move items soon after those items have been dropped by other ants.*

*There are two variations of short-term memory that can be used. In both variations, ants track the locations of a limited number of items that they have previously dropped. In the first variation, a newly picked up item is compared directly with the individual items in the ant's memory. In the second variation, the newly picked up item is compared with items around the location of an item in memory.*”

### 3. Implementation

#### 3.1 UML Diagram

The implementation of this assignment was done in Java using full Object Orientation. Below is a class diagram of the implementation. This is followed by a list of classes with descriptions.






*Note: this class diagram is just a shell and for brevity does not provide the inner details of each class. A full copy of the source code is publicly available at:*

<https://github.com/StuartGordonReid/Ant.git>

### 3.2 Class Descriptions

Class Name	Description
Simulator	This class was used to encapsulate the experiments. For each experiment 75 separate simulations were executed the results of which were averaged out and used for observations in this report.
AntAlgorithm	This class contained the logic for the Ant Algorithm including the initialization of the 2D grid, the placement of Ants and Items onto that grid and the higher level functioning of the Ant Algorithm.
ClusterFinder	This class contains an algorithm that uses a variation of depth first search to locate all of the clusters of items within the 2D grid.
ClusterF	This is the product of the ClusterFinder algorithm. It encapsulates in the form of an iterable list each one of the clusters found on the 2D grid
Cluster	This class encapsulates a cluster. It is essentially an iterable linked list of items that are adjacent to one another on the 2D grid. A cluster can contain any number of items greater than some specified minimum cluster size.
ClusterAnalyzer	This class will analyze the set of clusters found on the 2D grid and extract the intra cluster distances of those clusters (the density of the placement of items) and the average inter cluster distance between each one of the clusters.
Grid	This class encapsulates a 2D array of generic Grid Objects. It also contains methods for printing out the grid.
GridObject	The GridObject class encapsulates the shared components of any item currently on the grid. Specializations of this class include Ants and Items. It also contains the coordinate logic used throughout the ant algorithm.

Item	The Item class is a specialization of the GridObject class. An Item object can be picked up and dropped by ants that pass over it.									
Ant	<div>The Ant class is a specialization of the GridObject class. An Ant object can move randomly around the board. An Ant object is capable of moving diagonally, horizontally and vertically:</div> <table><tr><td>MOVEMENT 8, TOP LEFT</td><td>MOVEMENT 1, TOP</td><td>MOVEMENT 5, TOP RIGHT</td></tr><tr><td>MOVEMENT 4, LEFT</td><td></td><td>MOVEMENT 3, RIGHT</td></tr><tr><td>MOVEMENT 7, BOTTOM LEFT</td><td>MOVEMENT 2, BOTTOM</td><td>MOVEMENT 6, BOTTOM RIGHT</td></tr></table>	MOVEMENT 8, TOP LEFT	MOVEMENT 1, TOP	MOVEMENT 5, TOP RIGHT	MOVEMENT 4, LEFT		MOVEMENT 3, RIGHT	MOVEMENT 7, BOTTOM LEFT	MOVEMENT 2, BOTTOM	MOVEMENT 6, BOTTOM RIGHT
MOVEMENT 8, TOP LEFT	MOVEMENT 1, TOP	MOVEMENT 5, TOP RIGHT								
MOVEMENT 4, LEFT		MOVEMENT 3, RIGHT								
MOVEMENT 7, BOTTOM LEFT	MOVEMENT 2, BOTTOM	MOVEMENT 6, BOTTOM RIGHT								
SmartAnt	The SmartAnt class contains the implementation for the first variant of the Ant that makes use of short term memory. In this class newly picked up items are compared directly with the items in the ant's memory. This memory then slightly biases the decision in the direction of the closest Item in memory.									
SmartLocationAnt	The SmartLocationAnt class contains the implementation for the second variant of the Ant that makes use of short term memory.									
AntMemory	This class contains the logic for the memory buffer used by SmartAnt and SmartLocationAnt objects to store recently encountered Items' coordinate locations. In this class newly picked up items are compared with items around the location of an item in memory.									

## 4. Observations

Observations of the ant algorithm can be done visually through the Grid class. An example of this functionality is included as an appendix to this document. For this experiment I chose to instead use the controls to guide the observation of the two algorithms.

### 4.1 Controls

For the two experiments conducted in this investigation the controls listed below were used to measure the performance of the algorithm. Either an increase in the control would indicate improvement in the performance of the algorithm (a control we want to maximize) or a decrease in the control would indicate an improvement in the performance of the algorithm (a control we want to minimize). Each one of these controls was calculated over a sample of 75 simulations to allow for statistical relevance.

4.1.1 CLUSTERS - The average total number of clusters formed (maximize)

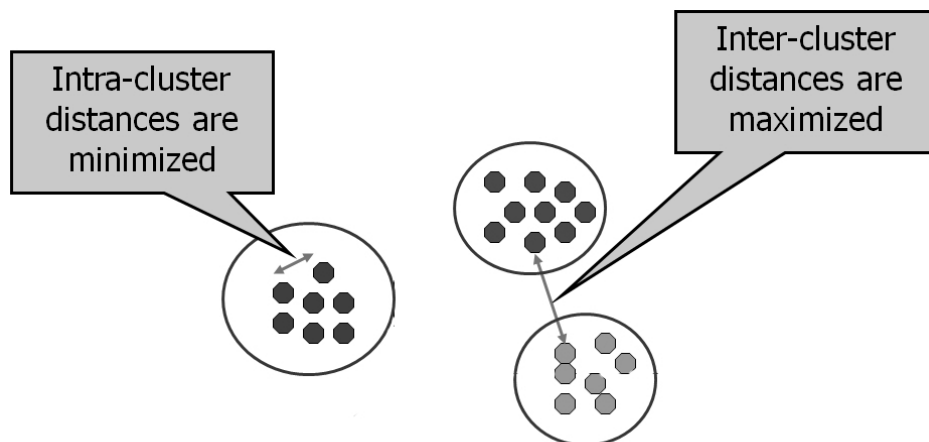
4.1.2 TOTAL - The average total number of items in clusters (maximize)

4.1.3 INTRA - The average intra cluster distances across all clusters (minimize)

This is the sum of the squared distances between all pairs of items in the cluster. The distances referred to here are euclidean distances.

4.1.4 INTER - The average inter cluster distances between all clusters (maximize)

This is the sum of the squared distances between all pairs of cluster where the distance between two clusters is calculated as the distance between the closest pair of point belonging to the clusters. This is referred to as chain shaped clusters.





#### 4.2 Experiment i - effects of the, Ants : Items, ratio

In this experiment I investigated the effects of increasing the ratio of ants to items on the two variants of the short term memory ant algorithm. For this experiments the following tuning parameters were chosen empirically and were constant:

1. Grid Size = 35 x 35
2. Items = 122 Items
3. Patch size = 1 x 1
4. Speed of ants = 1
5. Iterations = 200

The observations made during this experiment are tabulated below. The tuning parameters that were varied in producing these observations were the size of the memory buffer in each ant as well as the ratio of items to ants. I used the following values for these:

1. Memory buffer = {0, 1, 10, 100} and
2. Ant : Item's ratio = {1:3, 1:6, 1:9, 1:12, 1:15, 1:18, 1:21}

Additionally for each one of the tables listed on the next few pages colours have been included in the values to show which variant of the short term memory ant algorithm performed better in that particular circumstance.

## 4.3.1 Memory buffer - 0

## Short term memory ant algorithm variant 1

ANT : ITEM	CLUSTERS	TOTAL	INTRA	INTER
1:21	2	17	3.89	486.60
1:18	2	16	4.10	499.89
1:15	3	23	4.00	745.51
1:12	3	24	4.10	699.84
1:9	4	28	4.02	827.01
1:6	5	36	4.34	1,028.75
1:3	7	47	4.09	1,529.95

## Short term memory ant algorithm variant 2

ANT : ITEM	CLUSTERS	TOTAL	INTRA	INTER
1:21	3	18	3.93	559.84
1:18	3	20	4.18	622.64
1:15	3	23	4.00	673.52
1:12	4	26	4.42	759.72
1:9	4	29	4.02	895.91
1:6	6	33	4.41	894.07
1:3	7	47	4.11	1,389.48

## 4.3.1 Memory buffer - 1

Short term memory ant algorithm variant 1

ANT : ITEM	CLUSTERS	TOTAL	INTRA	INTER
1:21	3	19	4.44	547.07
1:18	3	19	4.16	604.32
1:15	3	23	4.41	713.81
1:12	4	26	4.21	733.63
1:9	4	30	4.33	954.69
1:6	6	37	4.27	1,021.32
1:3	7	47	4.27	1,340.24

Short term memory ant algorithm variant 2

ANT : ITEM	CLUSTERS	TOTAL	INTRA	INTER
1:21	3	17	3.90	529.49
1:18	3	20	4.00	748.67
1:15	3	23	4.09	691.51
1:12	4	24	4.16	669.71
1:9	4	28	4.36	855.91
1:6	6	38	4.18	1,060.48
1:3	7	47	4.32	1,354.95

## 4.3.1 Memory buffer - 10

## Short term memory ant algorithm variant 1

ANT : ITEM	CLUSTERS	TOTAL	INTRA	INTER
1:21	3	19	4.04	610.69
1:18	3	20	3.93	640.92
1:15	3	23	4.09	633.80
1:12	3	23	4.20	606.84
1:9	4	28	3.99	902.03
1:6	5	34	4.07	994.80
1:3	7	45	4.20	1,405.36

## Short term memory ant algorithm variant 2

ANT : ITEM	CLUSTERS	TOTAL	INTRA	INTER
1:21	3	18	4.08	526.97
1:18	3	19	4.54	581.44
1:15	3	22	4.27	687.04
1:12	4	25	4.32	734.97
1:9	4	29	4.12	914.67
1:6	5	36	4.19	1,004.79
1:3	7	47	4.22	1,363.11

## 4.3.1 Memory buffer - 100

## Short term memory ant algorithm variant 1

ANT : ITEM	CLUSTERS	TOTAL	INTRA	INTER
1:21	3	17	3.72	525.40
1:18	3	19	4.16	668.04
1:15	3	22	4.32	589.59
1:12	4	25	4.43	724.87
1:9	4	28	4.02	843.40
1:6	5	33	4.03	991.64
1:3	7	47	4.24	1,330.93

## Short term memory ant algorithm variant 2

ANT : ITEM	CLUSTERS	TOTAL	INTRA	INTER
1:21	3	19	4.34	575.32
1:18	2	17	4.01	490.61
1:15	3	23	4.14	635.27
1:12	3	24	4.12	730.95
1:9	4	27	4.24	833.47
1:6	6	37	4.11	1,139.39
1:3	6	45	4.27	1,435.35

### 4.3 Experiment ii - effects of the speed of the ants

In this experiment I investigated the effects of increasing speed of the ants on the two variants of the short term memory ant algorithm. For this experiments the following tuning parameters were chosen empirically and were constant:

1. Grid Size = 35 x 35
2. Items = 122 Items
3. Patch size = 1 x 1
4. Iterations = 200
5. Ant:Items ratio = 1:10

The observations made during this experiment are tabulated below. The tuning parameters that were varied in producing these observations were the size of the memory buffer in each ant as well as the speed of the ants. I used the following values for these:

1. Memory buffer = {0, 1, 10, 100} and
2. Ant Speed = {1, 2, 3, 4}

Additionally for each one of the tables listed on the next few pages colours have been included in the values to show which variant of the short term memory ant algorithm performed better in that particular circumstance.

## 4.3.1 Memory buffer - 0

Short term memory ant algorithm variant 1

SPEED	CLUSTERS	TOTAL	INTRA	INTER
1	4	26	4.16	885.69
2	6	40	4.35	1,075.71
3	7	46	4.32	1,549.04
4	8	55	4.35	1,622.59

Short term memory ant algorithm variant 2

SPEED	CLUSTERS	TOTAL	INTRA	INTER
1	4	26	4.20	749.01
2	6	39	4.12	1,193.65
3	7	49	4.41	1,442.23
4	8	55	4.25	1,541.69

## 4.3.1 Memory buffer - 1

Short term memory ant algorithm variant 1

SPEED	CLUSTERS	TOTAL	INTRA	INTER
1	4	29	4.30	770.97
2	6	39	4.11	1,149.93
3	7	49	4.24	1,458.87
4	7	53	4.33	1,601.85

Short term memory ant algorithm variant 2

SPEED	CLUSTERS	TOTAL	INTRA	INTER
1	4	27	4.12	854.21
2	6	39	4.33	1,297.72
3	7	47	4.30	1,365.13
4	7	53	4.25	1,561.59

## 4.3.1 Memory buffer - 10

Short term memory ant algorithm variant 1

SPEED	CLUSTERS	TOTAL	INTRA	INTER
1	4	28	4.15	849.29
2	6	40	4.18	1,158.05
3	7	47	4.31	1,316.64
4	8	55	4.34	1,589.32

Short term memory ant algorithm variant 2

SPEED	CLUSTERS	TOTAL	INTRA	INTER
1	5	31	4.04	956.08
2	6	39	4.34	1,182.23
3	7	49	4.39	1,440.93
4	8	55	4.30	1,722.77

## 4.3.1 Memory buffer - 100

Short term memory ant algorithm variant 1

SPEED	CLUSTERS	TOTAL	INTRA	INTER
1	4	27	4.20	816.64
2	6	40	4.11	1,236.60
3	6	45	4.18	1,317.60
4	8	56	4.33	1,740.41

Short term memory ant algorithm variant 2

SPEED	CLUSTERS	TOTAL	INTRA	INTER
1	4	28	4.41	829.25
2	6	43	4.42	1,298.76
3	7	50	4.61	1,302.48
4	7	52	4.55	1,496.56



## 5. Conclusions

After conducting many simulations and observing the behaviour of the short term memory ant algorithms I can quite comfortably conclude two things about the effects of the ratio of ants to data items, and the speed of ants on the performance of these algorithms:

1. A higher ratio of ants to data items for all memory sizes tested will more likely than not lead to an improvement in the performance of the algorithm in terms of:
  - a. The average inter cluster distances
  - b. The average number of items in clusters and
  - c. The average number of clusters.
2. A higher speed for the ants for all memory sizes tested will also more likely than not lead to an improvement in the performance of the algorithm in terms of:
  - a. The average inter cluster distances
  - b. The average number of items in clusters and
  - c. The average number of clusters.

And based on the empirical data and my observations I can also conclude that:

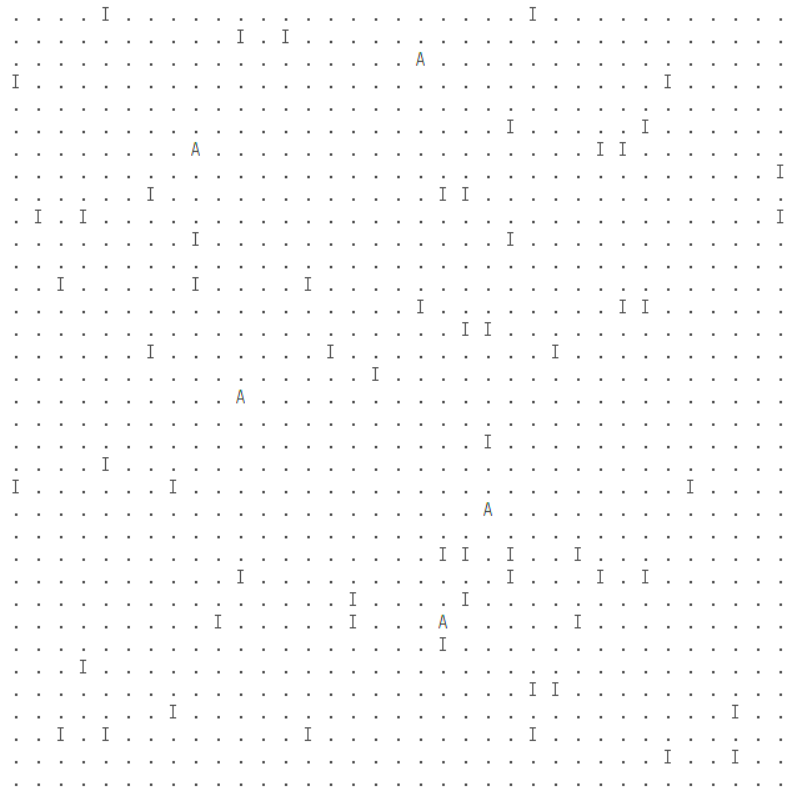
1. The range of inter cluster distances is relatively stable and does not vary wildly for any memory size or for either algorithm
2. Neither variant of the short term memory ant algorithm stands out as being consistently better than the other across all of the experiments and simulations conducted, however, in certain instances the performance of one of the variants did outdo the other one

Additionally based on the process of implementing these algorithms I am inclined to say that the value of adding short term memory to the ants in the algorithm does not outweigh the algorithmic costs of doing so. That is to say that the small improvement in the quality of clustering performed by ants with short term memories could have been achieved quicker and cheaper by increasing the number of iterations the simulations run for.

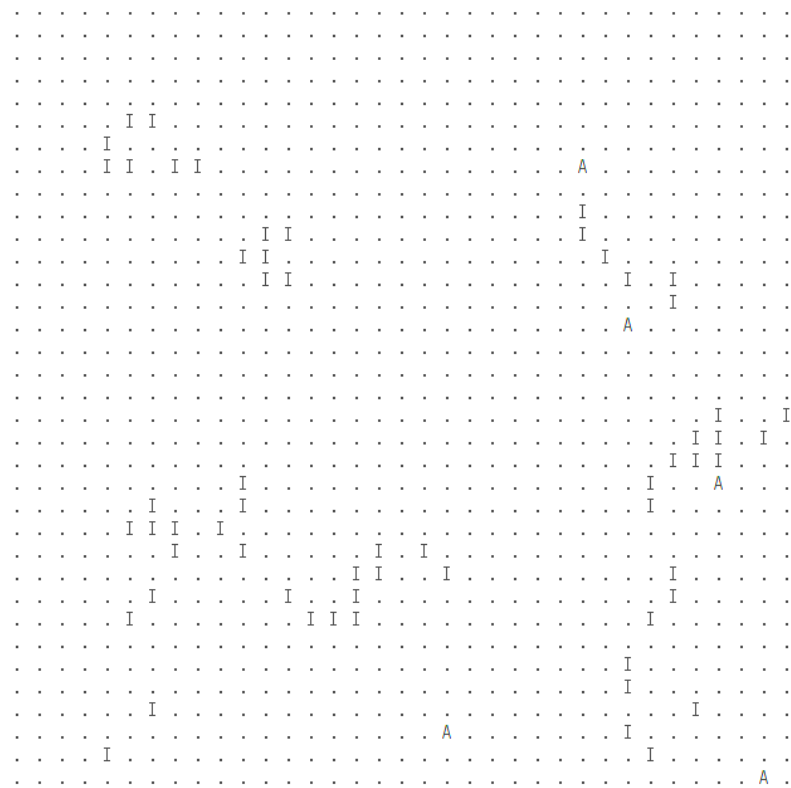
My concluding remark is that the value of cemetery formation ant algorithms comes from their simplicity. The complexity of their behaviour is emergent and attempts at expanding the algorithmic model through the extension of the ants capabilities will probably detract from the overall algorithmic simplicity and, in so doing, decrease the real value of the algorithm.

## B. Visual Observations (Appendix)

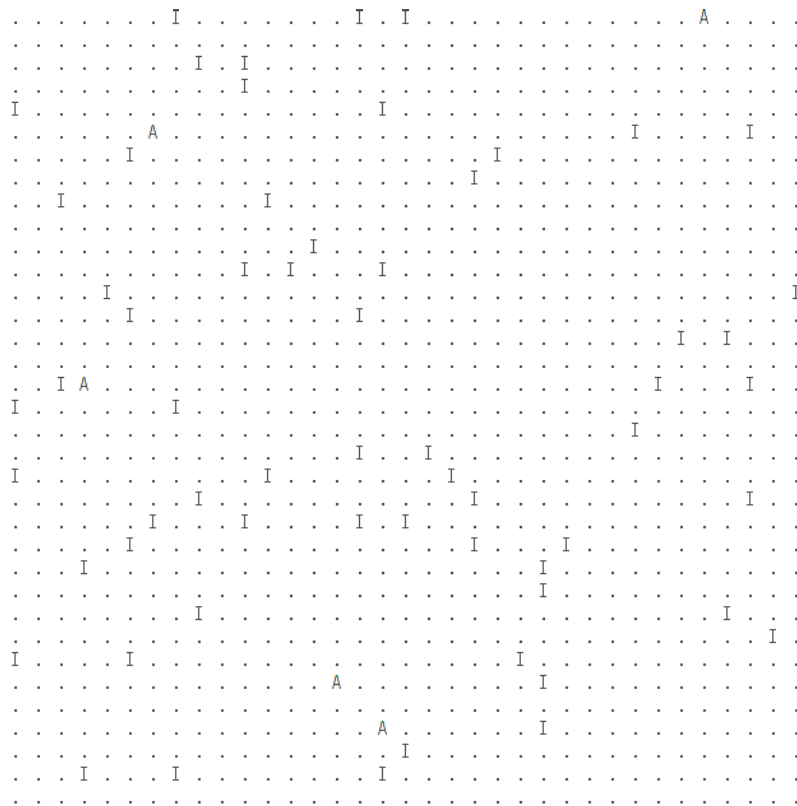
Visual clustering simple ant algorithm before



Visual clustering simple ant algorithm after



### Short term memory ant algorithm before



### Short term memory ant algorithm after

