The Evolution of Spacing Effects in Autonomous Agents

Prepared by James Markovitch 9-21-94
Dun & Bradstreet, Search Technologies

Abstract

This paper discusses research into whether the memories of adaptive autonomous agents can be made to spontaneously evolve spacing effects. Experiments involving human memory have shown that learning trials massed closely in time elicit slower learning than the equivalent number trials spaced apart in time. These "spacing effects" have been observed across a wide array of conditions. The experimental results detailed here show that such effects can be made to evolve spontaneously in autonomous agents. The results also suggest that the greater learning difficulty humans experience from closely spaced trials may not be the result of a defect of biology, but rather may be a consequence of a need to give only the appropriate weight to each learning experience.

1 Introduction

The purpose here is to simulate the evolution of memory and by doing so to discover how human memory is organized, and why it is organized the way it is.

The specific topic under consideration will be the evolution of human learning. The method employed will require that autonomous agents with memories move, perceive, reproduce and evolve in a computer simulated environment. This technique is suggested by the work of Landauer (1975) and Dawkins (1976); and has been more recently used by MacLennan (1992) and also Ackley and Littman (1992).

After this evolutionary process has had an opportunity to act, the memories of the evolved population will be "dissected" to discover what if anything they have in common with human memories, and what light, if any, they may shed on human cognitive organization and development.

2 Overview

2.1 The Rationale for this Simulation of the Evolution of Memory

Let us start with the following 3 X 3 grid, populated by numbers ranging from 1 to 4:

C	1	1	2
B	3	4	3
A	1	2	3
	a	**b**	**c**

The upper and lower case letters will serve as coordinates.

If an agent starts at **Aa**, it might make the following trip:

Aa -> **Bb** -> **Bc** -> **Cb**,

in which case its memory will have "seen" in turn 1, 4, 3, and 1.

In keeping score of what it has seen it might create the following memory array:

Memory:

Object 1: Seen: 2 time(s)

Object 2: Seen: 0 time(s)

Object 3: Seen: 1 time(s)

Object 4: Seen: 1 time(s)

If on a subsequent trip through the grid it ambiguously "saw" that cell **Ca** contained an odd number, the agent could then consult this memory and make a guess that 1 rather than 3 is the odd number that it saw, on the grounds that within its experience 1 has occurred twice and 3 has occurred only once. The agent could then move to this cell and verify the correctness (or incorrectness) of its guess.

In saying that an agent ambiguously sees something, we mean that the agent sees something about the object (perhaps that it is odd, or that its lower two bits are 11, or that it is yellow, etc.), but it does not have enough information to positively identify it. In what follows the act of resolving this ambiguous perception through guessing will be referred to as *disambiguation*.

Naturally agents that can resolve ambiguities more accurately than others should have an edge in surviving. This follows because some ambiguities are best resolved at a distance: for instance, the distinction between food and a predator.

Accordingly, a computer program might simulate the evolution of human memory by populating a grid with numbers (where these numbers represent food, predators, objects, etc.), and then letting a group of agents move about this grid, with the computer program all the while keeping a running tally of which agents perform more effectively in resolving ambiguously seen numbers. The computer program could then see to it that agents that do a better job of disambiguating are more likely to reproduce and more likely to survive, and that top scoring agents are more likely to breed with other top scorers than with low scorers. In this way a generic test environment could be exploited to evaluate the plausibility of a wide range of theories of human memory.

The above is a rough overview of the rationale and justification behind the simulation described by this paper. Naturally, there are many possible starting points for exploiting the methodology described above and many unresolved issues of detail regarding the implementation. There are in particular three key issues of implementation:

* What property of memory should be investigated?

* In what ways should agents differ in their disambiguation strategies?

* If the agents are to reproduce, how should their strategies be represented genetically?

In order to keep things simple, I have chosen to answer these questions as follows:

* The subject of investigation will be the evolution of spacing effects in memory. This effect is well known, and relatively easy to simulate. (See below for a description.)

* All agents will begin by using no spacing effects. Through subsequent reproduction, agents will have the opportunity to evolve spacing

effects or not as the case may be. In addition creatures may evolve the inverse of spacing effects.

* Each agent's spacing effects will be represented by an array of numbers, and the evolution of this array will be carried out by a genetic algorithm (see Holland 1975; and Goldberg 1989).

The overall objective will be to determine if spacing effects evolved for the purpose of improving the perception (or to be more specific the disambiguation) of human beings.

2.2 What Are Spacing Effects?

Experiments involving human memory have shown that learning trials massed closely in time elicit slower learning than the equivalent number trials spaced apart in time. In addition studies have shown that massed learning trials elicit poorer retention than spaced trials. This effect has been observed across a wide array of conditions. This phenomenon is referred to in the literature as "spacing effects." See Wickelgren (1977) for discussion.

Many theories are possible to explain why this should be so. For example, one might assume that spacing of learning trials allows a period of rehearsal between the trials. Or one might assume that unconscious consolidation of the first trial may take place between the trials. Or that trials separated in time might in some way be "differentially encoded" (that is to say encoded differently) with the consequence that there are more ways of recalling the item learned. And lastly, some studies (e.g. Hintzman, Block and Summers 1973) suggest the effect may be a consequence of a lower level of acquisition of the second occurrence.

Though this programming experiment cannot resolve the above issues, it may shed some light on them. It can shed light on them by answering the question: Do spacing effects help or hinder human disambiguation, or perhaps have no effect at all?

On the one hand, if they help, this might offer an explanation of why the phenomenon of spacing may be so pervasive across such a wide range of learning experiments: Spacing effects may be built into learning at a root level as a means of aiding perception and disambiguation.

On the other hand, if they hinder disambiguation, then this has the makings of a profound puzzle: Why should humans evolve an trait that apparently hinders their ability to survive?

And, lastly, if spacing effects neither help nor hinder disambiguation, then perhaps spacing effects are an experimental quirk, maybe the result of unconscious consolidation between trials, or increased time of rehearsal for the first learning episode.

3 Implementation

3.1 Spacing Effects Arrays

In this experiment autonomous agents randomly wander as a group around a toroidal grid. This rectangular grid has within each of its cells a randomly assigned number that may be thought of as an "object."

All of the agents see these objects as they move from cell to cell in the grid. Each agent has his own memory of what he has seen. And each has his own Spacing Effects Array that enables him to simulate human spacing effects.

For example, the group may observe the following objects:

Time 0: 1

Time 1: 4

Time 2: 3

Time 3: 1

Each agent in the group will register in memory that 1, 4 and 3 have been seen for the first time by setting the following values in Long Term Memory (the number 100000 is an arbitrary number; so long as this number is above zero it does not affect the experiment):

```
Time 0:   LongTermMemory[ 1 ] = 100000;
Time 1:   LongTermMemory[ 4 ] = 100000;
Time 2:   LongTermMemory[ 3 ] = 100000;
```

However, upon seeing 1 for the second time, Long Term Memory will register this by incrementing LongTermMemory[1] by an amount equal to SpacingEffects[3 - 0], which is to say SpacingEffects[3]. In other words, LongTermMemory will be incremented by an amount that depends, firstly, on how long ago it was that 1 was last seen (in this case the two sightings of 1 were three units of time apart); and secondly it depends on what is in element 3 of the Spacing Effects Array for that agent.

Accordingly, the weight given to these later observations will depend on what is in each agent's Spacing Effects Array. Observations close together in time will cause a value from the lower portion of the Spacing Effects Array to be added to Long Term Memory. And similarly, observations spaced widely in time will cause a value from the upper portion of the Spacing Effects Array to be added to Long Term Memory. As each agent explores the grid of objects, each will develop a different Long Term Memory of what he has seen. (Note that each agent does not remember locations, only objects. So it is not possible for a agent to disambiguate by remembering that a 1 once occupied cell **Aa**, for instance.)

Subsequently, the same agents will wander the same grid to test their memories against the environment. Before each move, each agent will "see" the lower bits of the next number to occur. Rather than view just the lowest bit as in the example from the introduction (which would tell merely whether the number is even or odd), the agent instead sees several of the number's bits. Each agent will then consult his Long Term Memory to try to guess the number. In each case any of several numbers from its Long Term Memory will have the requisite bits--so it must select from these that which it regards as most likely. A separate tally of correct predictions is kept for each agent.

Naturally predictions will differ only if the Spacing Effects Arrays used by the agents differ. Initially, all of the Spacing Effects Arrays are randomly initialized to constants--but not the same constant. By way of example, one agent's Spacing Effects Array may be initialized to be an array containing only 34's. Another's might contain only 67's. After the effectiveness of each agent's memory is tested and their correct scores are tallied, the top scorers are sorted to the top. Then the Spacing Effects Arrays of top scorers swap data with each other by the method of the genetic algorithm; and the lower scorers do likewise with each other. There is a randomly assigned *crossover point* that determines where this swap of data begins (see Holland 1975; and Goldberg 1989 for a description of crossover points).

Here is an example of two arrays (A1 and B1) combining by crossover to yield to new arrays (A2 and B2). The original arrays are overwritten and lost by this process.

A1	B1		A2	B2	
34	67		34	67	
34	67		34	67	
34	67		67	34	<-- crossover point
34	67	==>	67	34	
34	67		67	34	
34	67		67	34	

In this way the Spacing Effects Arrays evolve according to simulated natural selection. Notice that array A2 has to some degree evolved spacing effects as it has lower numbers in the bottom of its array; these numbers signify that it will give correspondingly less weight to sightings spaced closely together in time. Array B2 in contrast will give *less* weight to sightings spaced closely together in time.

3.2 Rules for Reproduction

In this simulation the following rules guide evolution:

1. Low scorers die without reproducing.
2. Middle scorers die immediately *after* reproducing.
3. Top scorers reproduce without dying.
4. Top scorers always equal in number low scorers so as to keep the population at the same level.

4 Execution of a Sample Run

In running the program the following variables must be set:

1. How many *generations* will be run?
2. How many *agents* will populate the space?
3. How many *learning steps* will the agent take to learn about the grid?

4. How many *test steps* will it take to test the validity of what it has learned?

5. In how many differently populated grids will the agents be tested in before reproducing? (The idea here is that some grids may by virtue of the arrangement of their numbers be more or less conducive to spacing effects. To see that these effects average out it is therefore necessary that the agent test his Spacing Effects Array against many different grids before reproducing.)

6. How many bits are visible of an ambiguously seen number?

7. How many types of objects populate the grid?

8. What are the dimensions of the grid?

9. What number of low scorers die without reproducing with each generation? This number, which will be called the natural selection number, is the same as the number of top scorers who reproduce without dying.

10. What range of values are possible in the Spacing Effects Array?

For the test run on which this paper is based these values were as follows:

1.	Generations	=	500
2.	Agents	=	100
3.	Learning Steps	=	500
4.	Test Steps	=	2000
5.	Different Grids	=	20
6.	Visible Bits	=	5
7.	Object Types	=	200
8.	Grid Dimensions	=	20 x 20
9.	Natural Selection Number	=	10
10.	Range of Array Constants	=	1 to 10000

5 Results

If spacing effects are an aid to disambiguation one would expect Spacing Effects Arrays to spontaneously evolve that have a gradient that reflects the spacing effects observed by experimental psychologists. In the course of generations these type of agents should dominate the evolutionary landscape.

The array displayed in Figure 1 is one of those that evolved under the conditions described above. It shows how much weight (y-axis) should be given to observations separated by from 1 to 499 units of time (x-axis). In its main features it is representative of its fellows. Because the test environment allows for 500 learning steps, this Spacing Effects Array must have 499 elements--objects may be seen from 1 to 499 time units apart. The weight varies from 1 to 10000 as this is the range of constants that populated the original set of Spacing Effect Arrays. The presence of spacing effects is manifest in the gradually increasing weights in the first one hundred elements. Figure 2 is more detailed graph of these first 100 array elements.

6 Discussion

Earlier it was proposed that the results of the simulation might shed light on human psychology. It is now time to revisit that issue. The above results suggest that spacing effects may be built into learning at a root level, and may operate by seeing to it that subsequent learning takes place with a lower level of acquisition. This is as suggested by Hintzman, et. al. (1973).

Why should the above spacing effects materialize in this computer simulation? A simple explanation is that memory is trying to build in Long Term Memory a view of what exists *physically*--and in particular, a view of which objects occupy more than just one cell in the grid. To do this it must take into account that two closely spaced sightings of the same object may signify that in fact the same object was seen twice, rather than that two objects were seen occupying two distinct cells. Naturally the more time that separates these sightings the more likely that two distinct objects do exist.

This conclusion is supported by a quirk that emerged in an early version of this program. In allowing for movement by the agent one must decide two issues. The first issue is the width and height of the jump, and the second is whether an agent may jump in place. In the first version of the program, jumping in place was not allowed. It followed that if the same object were seen twice in a row it must occupy different cells. This led to an interesting phenomenon. Spacing Effect Arrays evolved that had large numbers in the two lowest elements of the array, followed by a series of lower numbers. The lowest element is always unused as the time separating two sightings is always nonzero. But the second lowest element comes into play whenever two elements occur in immediate succession. That the evolutionary process placed a large number in this element suggests that it was exploiting the rule that an agent may not jump in place, and that an object seen twice in a row must occupy two distinct cells

Therefore this Spacing Effects Array element appears to be the exception that proves the rule: In this simulation spacing effects serve the purpose of helping memory discern which objects occupy the most cells, as distinct from which objects have been seen the most often.

In order to verify the above conclusion a program was written that kept track how often two sightings of the same object occurred at the same location, and how often not. The results are plotted in Figure 3, where the x-axis displays distance in time, and the y-axis the probability that the second object seen occupies its own distinct cell. The results mirror those in the earlier graphs, which suggests that memory is trying to build a *physical* model of what it has seen.

In the early part of the graph, the further apart two sightings were, the greater the likelihood that two separate cells were involved rather than the same cell twice. Notice however that the probability that two object sightings are from different cells increases only up to about 100 units of time, and thereafter it tails off somewhat. This may be a consequence of the toroidal nature of the grid. The agent may reach the same location by traveling "around the world" so to speak. As the spacing increases beyond 100 such travel becomes more likely. It is significant therefore that in Figure 5, which displays the same computed probabilities for a 200 x 200 grid, the probabilities do not tail off as they did for the 20 x 20 grid.

In general the shape of Figure 3 maps well onto the evolved Spacing Effects Arrays, as may be seen in Figure 4. This suggests that the gradient observed in the genetically developed array is approximating the probability gradient of Figure 3. This clearly supports the conclusion that this simulated memory is trying to build a *physical* model of the grid from its observations.

A few random observations:

1. The ragged look of the upper portion of the evolved array may be accounted for as follows: The lower portion of the Spacing Effects Array may be initially subject to more evolutionary pressure than its upper portion, and so it may evolve more uniformly. Supporting this conclusion are two observations: Computer simulation showed that the lower portion was used far more often than the upper portion. And, the lower portion of the array tended to stabilize throughout the population many generations earlier than the upper portion.

2. Allowing the program to run until the arrays of all agents are identical does not assure that a plausible array will evolve. If reproduction occurs before adequate testing of each agent in the environment the arrays evolved tend to be very "noisy." Specifically, small numbers tend to find their way into the upper portions of the array. In later generations these numbers "breed out" only to a limited degree.

7 Conclusion

Some closing ideas:

* The biological tendency to "learn less" from closely spaced trails may actually be an aid to learning in the sense that this tendency sees to it that no more weight is given to a learning event than is appropriate. Accordingly the greater learning difficulty humans experience from closely spaced trials may not be the result of a defect of biology, but rather may be a consequence of a need to give only the appropriate weight to each learning experience.

* Human spacing effects may result from man's need to distinguish between: *the same object seen twice*, and *two identical but distinct objects each seen once*.

* Because the above environment is a simple standard environment it might be used to test the plausibility of other theories of how human memory anticipates and disambiguates.

* By uncovering the optimal form for a memory that must perceive, learn, disambiguate, anticipate, and evolve in the above environment, it is possible that much can be learned.

* By examining the environment that has shaped human memory through evolution, it should be possible to illuminate the organization of human

memory itself, in much the same way that examining, say, the game of chess, might enable us to guess the internal organization of a chess-playing computer.

Acknowledgements

I would like to acknowledge the assistance of Jian Chen, Greg Farrington, Gary Griner and Tom Thornbury in preparing this paper.

References

1. Ackley, D. H., M. L. Littman, "Interactions Between Learning and Evolution," Artificial Life, vol. X, edited by C. G. Langton, C. Taylor, J. D. Farmer, & Rasmussen, SFI Studies in the Sciences of Complexity, Proc. Vol. X, 487-509. Redwood City, CA: Addison-Wesley, 1992.

2. Dawkins, R. *The Selfish Gene*. Oxford: Oxford University Press, 1976.

3. Goldberg, D. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Reading, MA: Addison-Wesley, 1989.

4. Hintzman, D. L., R. A. Block, and J. J. Summers, "Modality Tags and Memory for Repetitions: Locus of the Spacing Effect," *Journal of Verbal Learning Behavior*, 1973 (12), 229-38.

5. Holland, J. H., *Adaptation in Natural and Artificial Systems*. Ann Arbor: University of Michigan, 1975.

6. Landauer, Thomas K., "Memory Without Organization: Properties of a Model with Random Storage and Undirected Retrieval," *Cognitive Psychology*, 1975 (7), 495-531.

7. MacLennan, Bruce J., "Synthetic Ethology," Artificial Life, vol. X, edited by C. G. Langton, C. Taylor, J. D. Farmer, & Rasmussen, SFI Studies in the Sciences of Complexity, Proc. Vol. X, 631-658. Redwood City, CA: Addison-Wesley, 1992.

8. Wickelgren, W. A. *Learning and Memory*, 1977.

Figure 1. Weight Plotted Against Time Between Observations

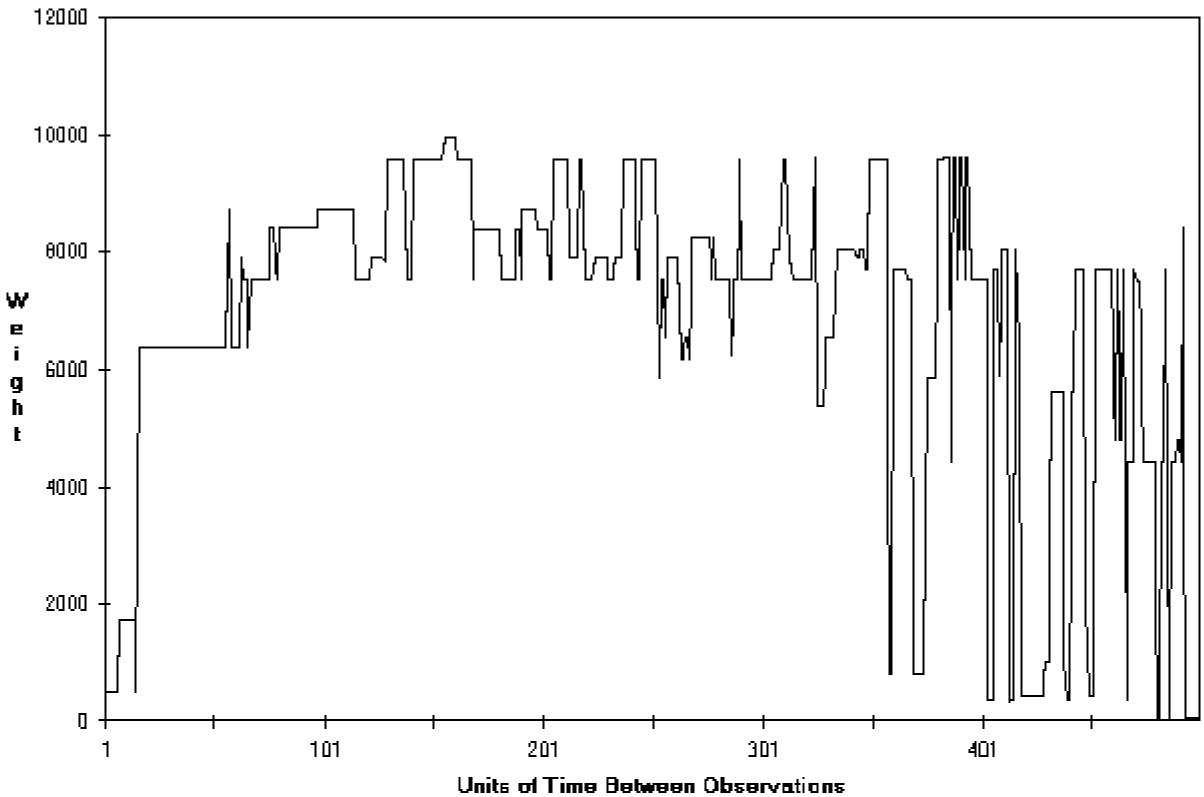

Figure 2.   Weight Plotted Against Time Between Observations

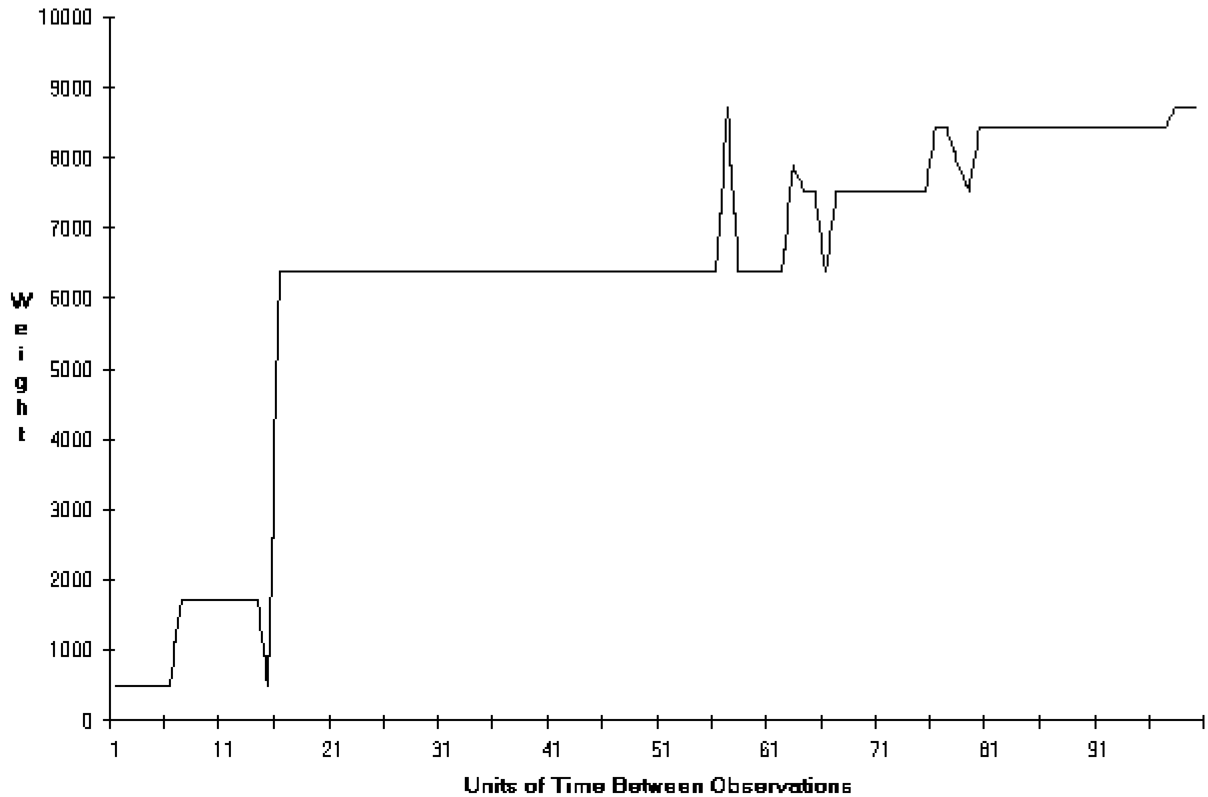

Figure 3. Computationally Observed Probabilities that an Object Seen Occurred in Two Distinct Cells

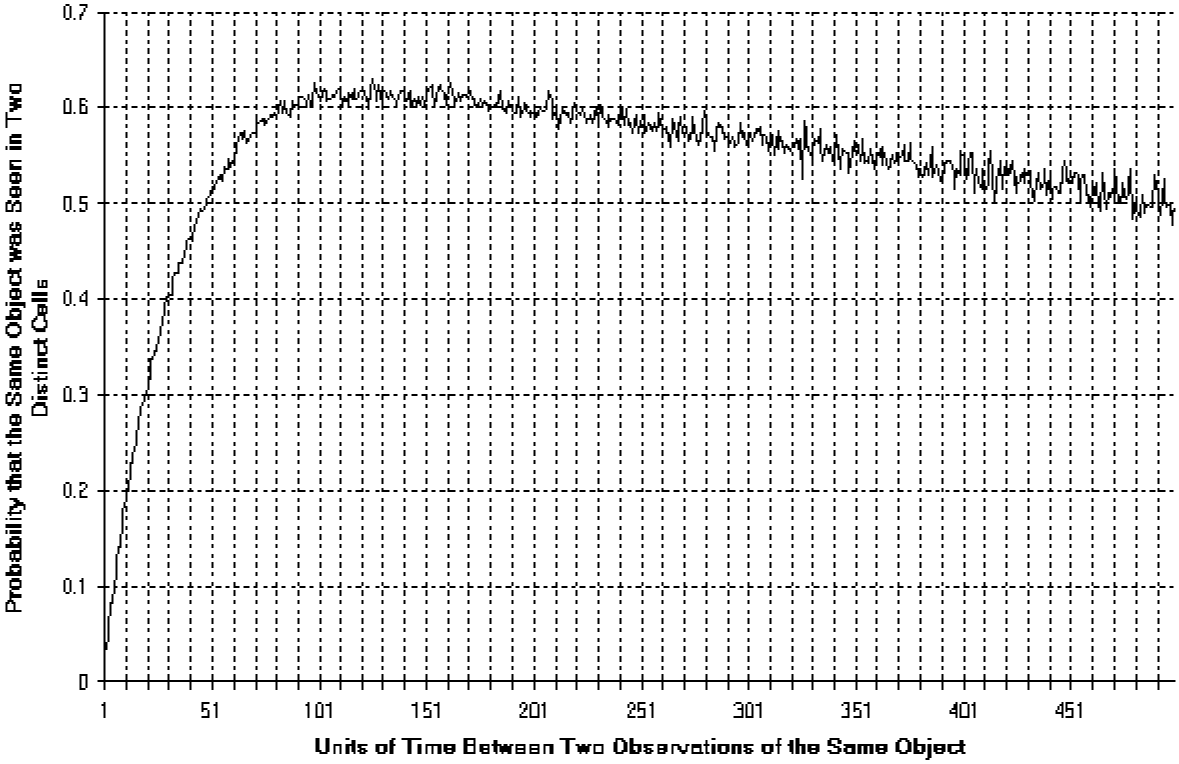

Figure 4. Weight (Series 1) and Probability (Series 2) Plotted Against Time Between Observations

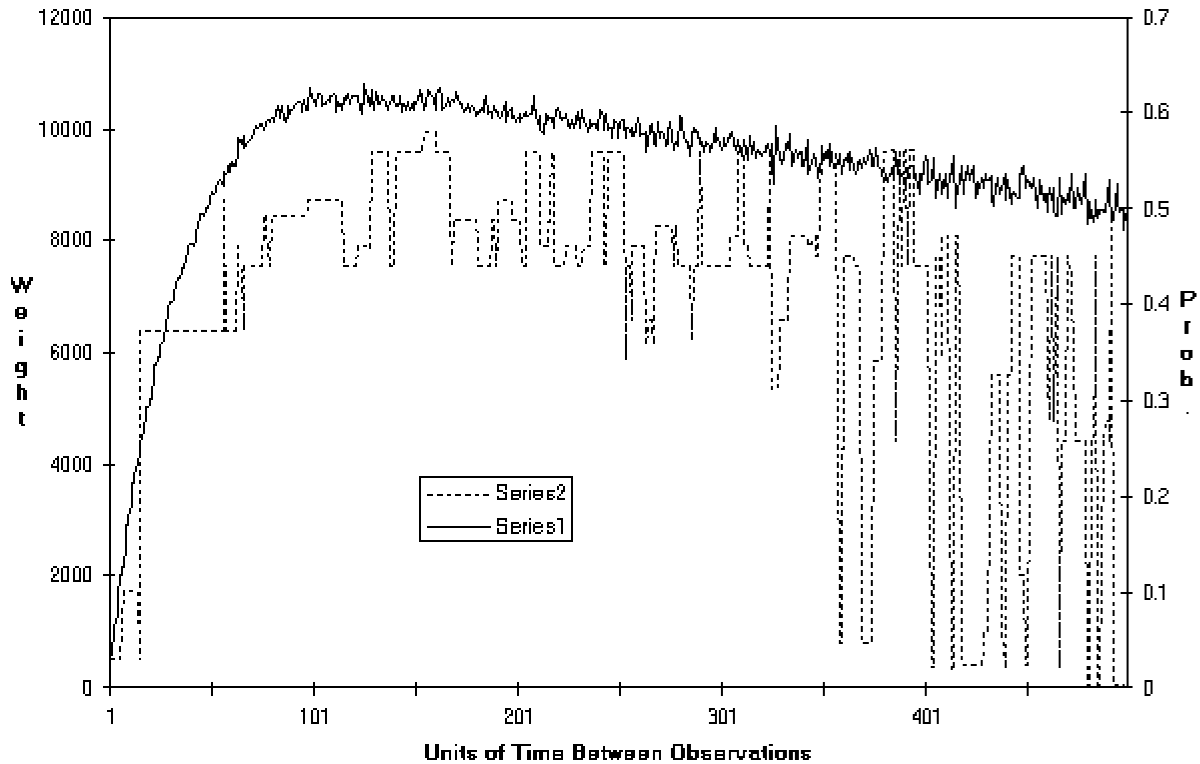

Figure 5. Computationally Observed Probabilities that an Object Seen Occurred in Two Distinct Cells; Series 1: 200 x 200 Grid; Series 2: 20 x 20 Grid