Simple storage models for immediate memory: a working paper.

Gordon Bower

Paper written while attending SSRC Institute on Computer Simulation of Cognitive Processes, July 1963. The Two-Bank model reported here was programmed in IPL-V and was run in several simulated experiments as the writers's Institute project. The restraints on the model-building enterprise were primarily due to my limited programming skills.

A number of writers in psychology have suggested analogs between the memory storage system of a digital computer and the memory system of a human. This paper grew out of an attempt to fill in the analogy. The computer stores items of information in addressable cells or registers, one item to a cell in the simplest case. We will discuss a few properties of several models which have in common the assumption that there is but a small number of memory cells for storage of incoming items of information. The parameter N will represent the number of cells available in memory; incoming information is stored in these, one item to a cell. The various models represent different schemes for doing the storage.

There are two experimental contexts in which the models will be made to perform. The first is in the Shepard-Teghtsconian experiment (1961) in which there is an indefinitely long string of incoming items; as each item appears, the subject is to judge whether that item is new in the series or whether it has appeared before (gives "old" as response). The response rule for the S-T experiment is always this: given the next item in the string, S searches for it through his N cells of memory; if he

finds it there, he says "old"; if it is not there, he says "new".

The second experimental context for the models is one devised by Yntema (1962), and modified here. As in the S-T set up, S monitors a long string of items, responding old or new. But there are frequent choice tests on which two items are presented, and S is to judge which of the two has occurred more recently in the series. The response rule for this task assumes that the N memory cells are on a partially ordered list; S scans this list for the two items and chooses that item which is found "higher up" in the memory list. If there is a tie, or if neither item is found in memory, then S chooses randomly on the test.

We begin by examining two simple models which are obviously inadequate, but yet which have some useful properties we will want to work into the better models.

A. Random storage: The N cells are strictly ordered on a memory list. To store a new item, we select a cell at random and store there the new item, erasing whatever may have been stored there previously. The probability that any particular item of information in memory (M) at trial n gets obliterated by trial n + 1 is 1/N, since there is random selection of the cell to which the new item on trial n is to be assigned. Let L be a random variable representing the number of trials between the trial on which some item is stored and the trial on which it is erased from memory. The L stands for "lifetime in memory". The probability distribution of L is

$$(1) \qquad \Pr\{L = k\} = \frac{1}{N}\left(1 - \frac{1}{N}\right)^{k-1} \qquad \text{for } k = 1, 2, \ldots$$

The average probability that an item input at trial 0 is still in M at

the beginning of trial n is

$$(2) \qquad p_n = \left(1-\frac{1}{N}\right)^{n-1}$$

The expected life-time of a particle of information in this system is

$$(3) \qquad E(L) = \sum_{k=1}^{\infty} k \Pr\{L = k\} = N$$

And the variance of the life-times is

$$(4) \qquad \mathrm{Var}(L) = E\left[(L - \bar{L})^2\right] = N(N - 1)$$

The undesirable features of this model are several. First, it gives (in (2)) a simple exponential forgetting curve which will not fit the empirical data. Second, it will not respond in a reasonable way to the Yntema task; because of the random storage, the order of items in M has no correspondence to their recency of appearance in the incoming sequence.

B. Push-down storage list: M is a simple push-down list of length N. Incoming items are pushed down on top of M. If all N cells of M are already filled, then inputing a new item to the top of M results in the dropping off of the last item in M. At steady state, then, a new item inserted at the top pops off from the bottom of M the oldest item. This model has the feature that decisions in the Yntema-task regarding the "more recent" item are easily effected since the order of items on M directly reflects their recency. The disadvantages of the model are the following:

(1) Each item is in memory for exactly N trials; there is no variance to the retention.

(2) The temporal discriminations are perfect, unless the more recent of the two items occurred more than N trials ago -- in which case, choices are random.

Mixed models.

The remaining models to be considered involve various mixtures of the two previous notions of random selection and pushing-down. They give predictions that at least are not obviously contradicted by the data known to me.

C. The two-bank model: Imagine that we partition our N memory cells into two levels or banks, called B0 and B1, which will vary in their vulnerability to getting bumped out of memory on a trial. Let N_0 and N_1 (which sum to N) be the respective number of storage cells in B0 and B1. An item of information is input to B0, the "immediate memory" bank. This is done by randomly selecting one of the N_0 cells in B0 and writing in it the incoming item. The item which formerly occupied this cell in B0 is then either forgotten (lost) with probability Θ or, with probability $1-\Theta$, it moves down to B1 and is stored in a randomly selected cell from the N_1 in B1. In the latter case, the old item formerly in B1 which gets bumped out is forgotten (lost).

We will perform calculations to follow the course of a particular item which is input to B0 on some arbitrary trial, which will be labeled trial 1. Let the states of the Markov chain be 0,1,F corresponding to the item being in B0, B1, or the Forgotten state on some particular trial. Abbreviate $x = \frac{1}{N_0}$ and $s = \frac{1}{N_1}$. The matrix of transition probabilities from one trial to the next for our particle of information is:

(5)

state on trial n \ state on trial n+1	F	0	0
F	F	0	0
1	$s(1-\Theta)$	$1-s(1-\Theta)$	0
0	$x\Theta$	$x(1-\Theta)$	$1-x$

We begin the particle in state 0 on trial 1. Let $W_{i,n}$ represent the probability that it is in state i at the beginning on trial n. The equations for $W_{i,n}$ are

$$W_{0,n} = (1-\chi)^{n-1} \tag{6}$$

$$W_{i,n} = \frac{\chi(1-\Theta)}{\chi-s(1-\Theta)}\left[(1-s+s\Theta)^{n-1} - (1-\chi)^{n-1}\right]$$

$$W_{f,n} = 1 - W_{0,n} - W_{1,n}$$

If the particle is in state 0 or state 1 at the beginning of trial n, then it will be recognized (correctly) as an old item. Hence, the average probability of a correct recognition after n intervening items is

$$p_n = W_{0,n} + W_{1,n} \tag{7}$$

Equation 7 is the sum of two exponential decay functions, and will do a fairly good job at fitting empirical data.

State F is an absorbing state and every particle will end up there eventually. The average life-time of a particle before it enters state F (is forgotten) is given by

$$E(L) = \sum_{n=1}^{\infty} p_n = \frac{1}{\chi} + \frac{1}{s} = N_0 + N_1 = N \tag{8}$$

Thus, the average life expectancy of an item input to memory is N, the number of storage cells, and E(L) is independent of the turn-over rate,

Because of the simplicity of the model, it is possible to obtain the probability distribution of lifetimes, L. It is

$$(9)\quad Pr\{L=m\} = \frac{\theta}{N_0}\left(1-\frac{1}{N_0}\right)^{m-1} + \frac{(1-\theta)^2}{N_1 - N_0(1-\theta)}\left[\left(1-\frac{1}{N_1}+\frac{\theta}{N_1}\right)^{m-1} - \left(1-\frac{1}{N_0}\right)^{m-1}\right]$$

The mean of L is N, as given in Eq. 8. The variance is

$$(10)\quad Var(L) = N_0^2 - N + \frac{(1-2\theta)}{(1-\theta)^2} N_1^2 - 2N_0N_1\frac{\theta}{1-\theta}$$

Although the mean life-time doesn't depend on θ or the partition N_0, N_1 the variance does. The variance increases as $1-\theta$ increases, being at a maximum when $\theta = 0$. When $\theta = 0$, the particle knocked out of B0 goes to B1 without fail. The variance of its life-time is the sum of the variances of its (exponential) times in B0 and B1, viz,

$$Var\left(L \,\middle|\, \theta=0\right) = N_0(N_0 - 1) + N_1(N_1 - 1)$$

and this is largest when the N cells of memory are evenly distributed to B0 and B1 (i.e. $N_0 = N_1 = N/2$).

Analytic solutions can be derived from this model for the Yntema task of judging which of two items occurred more recently. The S searches for the location of each item, finding each in either B0, B1, or in neither (viz., state F). He then compares where the two were found, choosing the one in B0 as more recent than one in B1 or F, and choosing the one in B1 as more recent then the one in F. If both are found in the same state, then he chooses at random. To formalize this, suppose item A is input at trial 0 and item B at trial k. Some other inputs intervene and then the choice test for "more recent" of A or B is given on trial k + n. Let a_0, a_1 and a_F represent the probability that item A is in states 0, 1 and F at trial k + n, and let b_0, b_1, b_F represent similar state probabilities for item B at trial k + n. These probabilities can be calculated from Eq. 6.

Since each item can be in one of 3 places, there are 9 possible outcomes of the pair comparison. These are shown in Table 1.

Table 1

state of A	state of B	Probability of pair	Probability of saying "B more recent"
0	0	a_0b_0	.50
0	1	a_0b_1	0
0	F	a_0b_F	0
1	0	a_1b_0	1.00
1	1	a_1b_1	.50
1	F	a_1b_F	0
F	0	a_Fb_0	1.00
F	1	a_Fb_1	1.00
F	F	a_Fb_F	.50

From this table, it may be seen that the probability that S says B is the more recent is

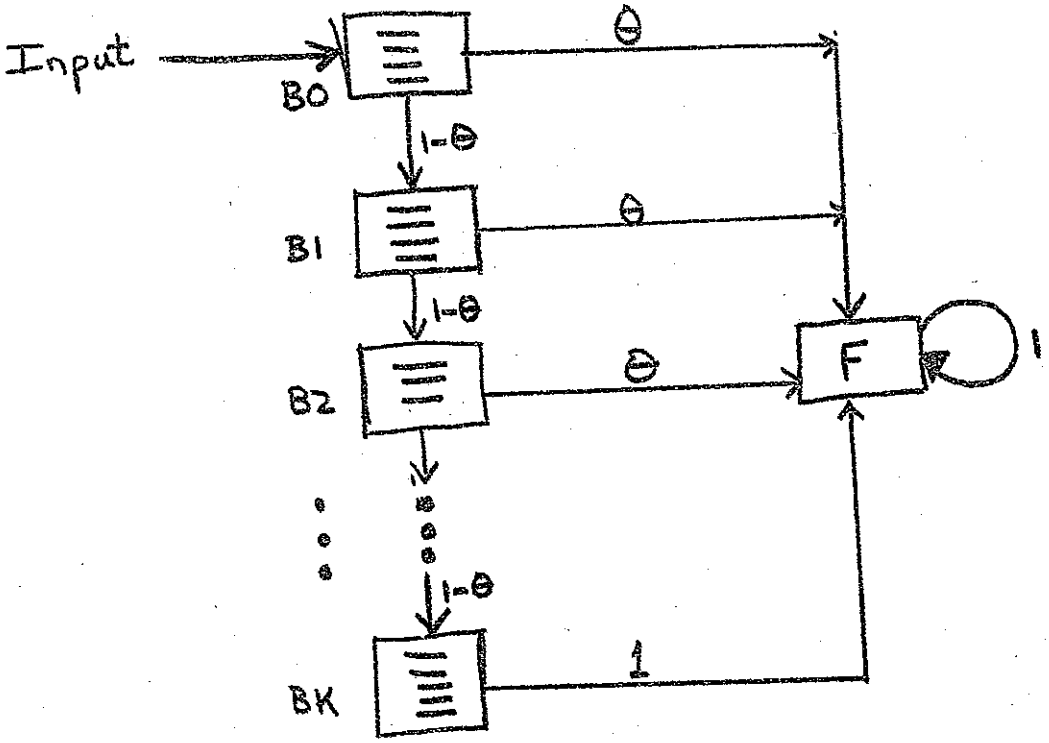
$$P(B) = b_0(1-a_0) + b_1a_F + .5(a_0b_0+a_1b_1+a_Fb_F). \tag{11}$$

Recall that A was input at trial 1, B input at trial k, and the test given at trial k + n. Eq. 11 says that P(B) increases with k; the greater the separation in time between A and B, the more likely that S will judge B as more recent. Also, for fixed k, P(B) decreases as n increases; the farther back in time that both events occurred, the more likely their order will be confused. In particular, as $n \to \infty$, $P(B) \to .50$ since both items are likely to have been forgotten. Trial-and-error plots of Eq. 11 yielded

curves (for k and n) that resembled the data reported by Yntema.

D. <u>The (k + 1) - Bank Model</u>: Remove the restriction on only two storage banks in the former model. Partition the N memory cells into k + 1 banks B0, B1,..., Bk with corresponding numbers of cells N_0, N_1, ..., N_k which sum to N. As before, an incoming item is placed in a randomly selected cell of B0. The item bumped from B0 can drop out of memory (probability θ) or can skip down to the next lower level, B1, and insert itself there, bumping out a random item from B1. The item bumped from B1 can be lost (probability θ) or can drop down to B2 and bump out a random item there, and so on. The process is diagrammed in Fig. 1. The boxes in Fig. 1 represent banks of storage cells (lines inside the boxes).

Fig. 1



A new item inserted into B0 causes a variable amount of reshuffling of the memory before the storage operation is completed. If θ were zero, then the input particle would cause a chain reaction in which one particle from each Bn is shifted to Bn+1, and a particle from Bk is lost into F. The storage operation continues each trial until some one item is lost to F, which then stops the chain reaction.

The diagram in Fig. 1 can be represented as a markov chain with states 0,1,...,k, and F. If an item is sitting in state i (bank Bi), on the next trial it can be moved to state i+1 or to state F_j or stay in state i. The transition probabilities for one-trial steps from state i are:

(12)
$$P_{i,j} = \begin{cases} (1-\theta)^i \theta / N_i & \text{for } j = F \\ (1-\theta)^{i+1} / N_i & \text{for } j = i+1 \text{ and } i = 0,1,\ldots,K-1 \\ 1 - \frac{(1-\theta)^i}{N_i} & \text{for } j = i \end{cases}$$

These transition probabilities arise directly from consideration of Fig. 1. The probability that some item in Bi will be bumped out on a trial is $(1-\theta)^i$; the likelihood that any particular item from Bi gets bumped is then $\frac{1}{N_i}$. With probability θ, the bumped item goes to F; with probability $1-\theta$, the bumped item goes to the next state, i+1. In case i=k (i.e., item is in Bk at start of trial), if the item is bumped, with probability $(1-\theta)^k/N_k$, then it is forgotten with probability 1.

The theorem to be proven about this system is that the expected life-time of an item in memory is N trials, and is independent of θ and the number of partitions (banks). We obtain the total expected life-time of an item by summing its expected life-times in each of the states 0,1,..,k. Call these l_i. The probability that an item entered into B0 ever gets

into state i sometime later is $(1-\Theta)^i$. Given that the particle has entered state i, the probability that it will be there k trials is given by

$$\Pr\left\{l_i=k \,\middle|\, i \text{ entered}\right\} = \frac{(1-\Theta)^i}{N_i}\left[1-\frac{(1-\Theta)^i}{N_i}\right]^{k-1} \tag{13}$$

The conditional mean life-time in state i is

$$E(l_i \mid i \text{ entered}) = \frac{N_i}{(1-\Theta)^i} \tag{14}$$

The unconditional expected trials in state i is given by the conditional mean in Eq. 14 times the probability that i is entered:

$$E(l_i) = (1-\Theta)^i\left(\frac{N_i}{(1-\Theta)^i}\right) = N_i \tag{15}$$

The total expected life-time is

$$E(L) = \sum_{i=0}^{K} E(l_i) = N_0 + N_1 + \ldots + N_k = N. \tag{16}$$

This completes the derivation. The result states that the expected life-time is independent of Θ and the number and divisions of banks of storage cells. The variance of L will depend on Θ and the partition, as it did in Eq. 10.

Further mathematical work on this system is cumbersome. The equation to solve for the $W_{i,n}$, the probability that our particle is in state i at the beginning of trial n, is

$$W_{i,n+1} = W_{i,n}\quad(1-\alpha_i) + \alpha_{i-1}W_{i-1,n} \tag{17}$$

where $\alpha_i = (1-\Theta)^i/N_i$. To date, the solution to Eq. 17 has not been worked

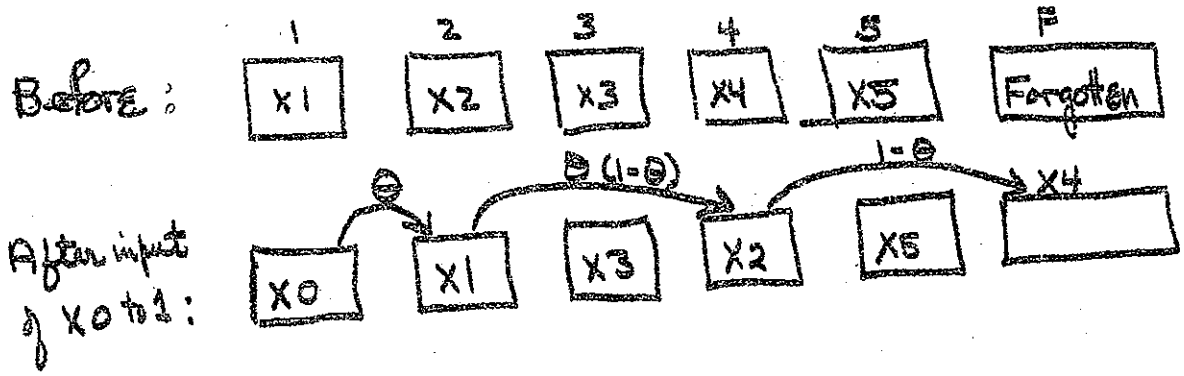
out, but it will involve sums of exponential decay terms. The mean retention curve will be the sum of the W's (excluding W_F), and thus will also be sums of exponential decay terms.

It is clear that this model, with more "temporal" divisions (banks), will perform better on the Yntema-task than will the preceding model with only two banks. It's also clear that analytic solutions can't be had until the equations for $W_{i,n}$ are solved.

E. The Leap-Frog model:

The N memory cells are arranged in a linear list. New items are input to cell 1 on the list, knocking out the old item in cell 1. This old item jumps down the list a geometrically distributed number of spaces, and possibly leaps clear off the end of the list (and is forgotten). If it lands somewhere on the list, it kicks out the item currently sitting in that cell, and this new item takes a jump down the remainder of the list. The storage operation on the trial ceases when some kicked item jumps off the end of the list. The scheme is illustrated in Fig. 2 before and after inputting the symbol X0 to a five-cell storage system.

Fig. 2



The states of the system are 1, .. , N, and F corresponding to an item's position on the list. If an item in cell i is bumped, the probability it goes to cell j directly is

$$P_{ij} = \begin{cases} \theta(1-\theta)^{j-i-1} & \text{for } i+1 \le j \le N \\ (1-\theta)^{N-i} & \text{for } j = F \\ 0 & \text{otherwise} \end{cases} \tag{18}$$

We want to calculate the expected life-time in memory of an item introduced into cell 1 on trial 1. As before, we calculate the prbability that the particle ever enters cell i and multiply this times the conditional mean trials in cell i given that i was entered; finally, these are summed over the N cells (states).

Let p_j be the probability that an item entered into cell 1 eventually lands in cell j at sometime later. Of course, $p_1 = 1$, since each item is entered into state 1. The general rule for p_j is

$$p_j = \sum_{i=1}^{j-1} p_i P_{ij} \tag{19}$$

The solution to Eq. 19 is $p_1 = 1$, and $p_j = \theta$ for $j \ge 1$. To prove this, make these substitutions and employ Eq. 18 for p_{ij}:

$$p_j = P_{1j} + \sum_{i=2}^{j-1} \theta \cdot \theta(1-\theta)^{j-i-1} \tag{20}$$

$$= \theta(1-\theta)^{j-2} + \theta\left[1-(1-\theta)^{j-2}\right] = \theta$$

Thus, an item introduced into cell 1 is equally likely to eventually appear in any of the other N-1 cells of memory.

Suppose an item enters state j. We wish to know how long on the average it will remain there. Let h_j be the probability that the item in

cell j is hit and bumped out (and farther down the list) on any given trial. Its average life-time in cell j will be $1/h_j$. Of course, $h_1 = 1$ since the first item is always pushed-out by the incoming message. The other h's are determined as in Eq. 19, viz.,

$$h_j = \sum_{i=1}^{j-1} h_i P_{ij} = \begin{cases} 1 & \text{if } j=1 \\ 0 & 1<j\leq N \end{cases} \tag{21}$$

For example, the item in the fourth position can be hit in the following ways when the first item is bumped by the incoming signal: 1 to 4, 1 to 2 to 4, 1 to 3 to 4, 1 to 2 to 3 to 4. The first "path" has probability $h_1 p_{14}$, the second $h_1 p_{12} p_{24} = h_2 p_{24}$, and the third and fourth pathes together have probability $(h_1 p_{13} + h_1 p_{12} p_{23}) p_{34} = h_3 p_{34}$. Equation 21 tells us that items in any given storage cell on the list are equally likely to be disturbed by the incoming information.

The total expected life-time of the particle of information is

$$E(L) = \sum_{i=1}^{N} p_i / h_i = N \tag{22}$$

Again we find that the expected life-time of items in the system is just the total number of storage cells available. This is an invariant feature of all the models regarding the storing process that has been considered in this note.

As before, for the general case of N storage bins, it is difficult to solve the equations for the state probabilities (the $W_{i,n}$'s). It would appear that the average retention curve will be positively decelerated according to this model; the longer an item has been in memory, the more likely it is that it will be forgotten in the next unit of time.

This happens because, with longer times in storage, it is increasingly more likely that the particle is near the bottom of the memory list where it is easily forgotten if it is hit. Some computations for the simple case of N = 3 shows the average retention probability is

$$(23)\qquad r_n = \begin{cases} 1 & \text{for } n = 1 \\ \theta(2-\theta)(1-\theta)^{n-2} + \dfrac{\theta^2}{(1-\theta)}(n-2)(1-\theta)^{n-2}, & \text{for } n \geq 2 \end{cases}$$

A graph of Eq. 23 shows r_n to decrease slowly at first, and then with increasingly larger decrements. This is an undesirable feature since no empirical retention curves I have seen have this shape. The next Leap-Frog model avoids this undesirable feature.

F. The Modified Leap-Frog Model;

In the former model, forgetting occurred because a "hit" item leaped beyond the end of the memory list; items at the top of the list were thus less likely to be forgotten per trial. We modify that rule as follows: whenever an item is hit from above, there is probability α that it goes directly into state F; with probability $1-\alpha$ it goes through the leap-frog process outlined before. If in the latter process, it leaps beyond the end of the list, it also will be forgotten on that trial.

For this model, then, the probability that an item hit in cell i will jump down and kick out the item in cell j is

$$(24)\qquad p_{i,j} = \begin{cases} (1-\alpha)\,\theta\,(1-\theta)^{j-1-i} & \text{for } 2 \leq j \leq N \\ \alpha + (1-\alpha)(1-\theta)^{N-i} & \text{for } j = F \end{cases}$$

For this process, the unconditional probability that a cell will be hit per

trial is smaller the lower down in the list it is. The relation is

$$h_j = \sum_{i=1}^{j-1} h_i \, p_{ij} \tag{25}$$

Writing out the expressions for h_j and for h_{j+1}, the following recursion is seen to obtain:

$$h_{j+1} = (1-\theta\alpha)\, h_j \tag{26}$$

Thus, the solution to Eq. 26 with the special boundary conditions is

$$h_j = \begin{cases} 1 & \text{for } j=1 \\ (1-\alpha)\,\theta\,(1-\theta)^{j-2} & \text{for } 2 \le j \le N \end{cases} \tag{27}$$

As before, the expected life-time of an item is cell j is 1 trial because the probability of ever entering cell j is identical to the probability that cell j is hit per trial (analagous to Eqs. 20, 21 and 22). Thus, the total expected life-time in memory is N trials for this system. The mean retention probability will be a weighted sum of exponential decay curves. New items in memory are likely to be forgotten; the older an item is in memory, the less likely it is to be hit and forgotten.

G. Miscellaneous considerations:

These models have assumed that the incoming items represent a single chunk of information, so that a single item can be stored as a unit, one to a storage cell. This condition might be approximated by using familiar words with human subjects. It is known that items containing several chunks of information (k-digit numbers, consonant k-grams) are forgotten more readily the larger is k. A simple way to handle this fact in the storage models

is to assume that several storage cells are required to store an item containing several chunks of information; an item in memory would then consist of a list of storage cells containing its information. This modification would degrade the memory of our systems to a degree directly correlated with the number of storage cells used per item. If a word uses one storage cell and a 3-digit number uses three cells, then the average life-time in memory of numbers would be one-third of that for the words.

Storing a k-chunk item as a list of cells raises the possibility that some of the information has been destroyed (or changed) when the item receives a later recognition test. The matching test by which the system says "old" or "new" to a current item would have to be modified to deal with and decide what to do with partial matches. The test might consist of trying to match each subobject (i.e., digit) of the current item to the stored subobjects of items in memory. A simple decision rule is for the system to adopt a criterion: if the number of subobject matches exceeds this criterion, it says "old"; otherwise, it says "new". The criterion selected supposedly would be influenced by the same variables that affect the criterion in standard detection tasks. (cf. Egan, 1958; Swets, Tanner, and Birdsall, 1961). Thus, if S were to rate the current item on a scale from 1 ("definitely new ") to 7 ("definitely old"), higher ratings would be output the greater the number of subobject matches. ROC curves may then be obtained by plotting for each j on the scale the probability of a rating of j or higher given that the item is old vs. the probability of a similar rating given that it is new (cf. Egan, 1958). Monte Carlos will have to be run on the systems outlined here to see whether

they generate the type of ROC curves found by Egan. The criterion selected should also be a function of S's knowledge of what proportion of the items in the series are new. Thus, if 80% of the items are new (which means that many that are input are not tested later), then S should set a high subobject-match criterion, saying "new" unless he is almost certain that the current item is old. By varying the proportion of new items over several series, ROC curves may be obtained. (Gallanter, personal communication, 1962).

A fact demonstrated by Shepard and Teghtsoonian is that the false alarm probability (saying old to a new item) increases to an asymptate as the experiment continues. This increase would be expected from all the present models because the number of match tests that must be performed (to decide "new") increases over trials. To illustrate, consider the simple random storage model. All of its N cells begin the experiment either empty or filled with irrelevant material that is passed over by the match test. Let C_n be the average number of cells that are filled with relevant experimental items at the beginning of trial n. Then the law for the growth of C_n over trials is

$$C_n = N\left[1 - \left(1 - \frac{1}{N}\right)^{n-1}\right] \tag{28}$$

The matching testor compares the input item only with the C_n relevant items stored in memory. Suppose that there is some small probability, E, that the match test misfires (and says "positive") for each item in memory it tries to match to a new item. Then the probability of a false alarm on trial n is

$$f_n = 1 - (1 - E)^{C_n} \tag{29}$$

This specifies a curve that grows (over trials) more or less exponentially to an asymptote of $1 - (1-E)^N$. This idea is readily generalized to the case where subobjects of an item are stored on a list occupying several cells. In that case, E would be a function of the number of positive matches of subobjects between the current item and a degraded item in memory.

The storage models have been discussed in the context of experiments on continuous, immediate memory in which individual items are presented only once or twice. But suppose the task is to be modified so that the same set of L items is repeated indefinitely. Intuition tells us that Ss eventually would recognize all L items as being old ones. Our models will not do nearly so well as the humans on this repeated list unless the number of memory cells far exceeds the number of items to be stored perfectly.

This conceptual experiment led to considering systems having very large memories. In fact, let the number of storage cells, N, be practically infinite. With this additional assumption, the various models differ markedly in their predictions. The push-down list, the random storage model, and the leap-frog model would imply perfect retention of each input item if N were infinite. The k-bank model can be extended either by fixing k and letting the size of the banks become infinitely large or by fixing the bank size, N_i, and letting k, the number of banks, become infinitely large. The first extension delivers the unreasonable prediction of perfect retention for each input. The second extension gives more plausible predictions; the average probability of forgetting a singly presented item follows approximately the same time course as before. It differs in that there is some small probability

that a singly presented item will never be forgotten; hence, the expected lifetime of an item in memory is infinite. The system will also display eventual perfect retention of a set of L items repeated sufficiently often, since eventually copies of all items get stored in higher banks from which there is negligible probability that they will be knocked out on a trial.

The modified-leap-frog model also stands up well in its predictions as N becomes indefinitely large. As with the k-bank model (of which it is a special case), singly presented items may be forgotten as before, yet sufficient repetition will insure their almost perfect storage.

References

Egan, J. P. Recognition memory and the operating characteristic. Tech. report June 15, 1958, Hearing and Communication Lab, Indiana University.

Sheperd, R. N. & Teghtsoonian, Martha. Retention of information under conditions approaching a steady state. J. exp. Psychol., 1961, 62, 302-309.

Swets, J. A. Tanner, W. P., Jr., and Birdsall, T. G. Decision processes in perception. Psychol. Rev., 1961, 68, 301-340.

Yntema, D. Recall as a form of data-processing. Paper read at AAAS meetings, 1962.