A Model-based Approach to Learning from Attention-focusing Failures

Michael Freed, Gregg Collins
The Institute for the Learning Sciences
Northwestern University

Abstract

In this paper we present a theory of how machines can learn from attention focusing failures. Our method requires that learning mechanisms have available a detailed model of decision-making mechanisms they are to modify; it is therefore central to this research to develop and present such a model. The portions of our developing model presented below concern those parts of a decision-making apparatus that should be approximately the same from one agent to another. Though learning mechanisms would have to be sensitive to both the idiosyncratic and agent-invariant elements of an adaptable decision architecture, we have concentrated on the invariant elements, which provide the most general constraints on learning.

Introduction

An agent in a complex environment can generally afford to consider only a tiny fraction of the information that is available to it before deciding on an action; therefore, such an agent must have some mechanism for focusing its attention on the information most likely to prove useful. The mechanisms of attention focusing are thus a critical part of the agent's cognitive apparatus, and serious failures may result when attention is focused improperly. For example, a driver may fail to react to a stop sign even though the sign is within his visual field, he understands the meaning of the sign and he wishes in general to stop at stop signs. In other words, although the high-level mechanisms for reacting to stop-signs were in place, and the raw data necessary to recognize this particular stop sign were available, the driver's attention-focusing mechanisms failed to recognize that this data should be processed further. This is the problem of attention focusing.

Attention focusing failures typically arise because of unanticipated interactions between cognitive tasks, each of which is placing demands on an agent's attentional resources. For instance, a driver might miss a stop sign because he is distracted by a commotion on the sidewalk. The task of determining what is happening at the side of the road draws off some of the perceptual and cognitive resources that would normally be committed to driving the car, thus causing the failure. In order to prevent a recurrence of this type of failure, the agent must determine how the competing tasks caused it to make the mistake, and modify its attention-focusing mechanisms so that they can successfully deal with similar task conflicts in the future.

In this paper we present a theory of how machines can learn from attention focusing failures.

Learning from Failure

Our approach is based on the paradigm of *failure-driven learning*, in which the agent relies on the observed failure of specific, monitored expectations to signal an opportunity to learn [Sussman, 1975, Schank, 1982, Hammond, 1989, Birnbaum *et al.*, 1990]. In particular, when an agent expects a plan to achieve its goal, and this expectation is violated, one response is to attempt to determine what aspect of the agent's decision making machinery was responsible for the faulty plan, and how that aspect can be modified to avoid the recurrence of such failures.[1]

When a plan unexpectedly fails, the first step in learning is to fault a modifiable component of the agent's decision-making machinery. One approach to identifying the faulty component is *model-based reasoning* [Davis *et al.*, 1982]. Just as in model-based systems whose task is to identify the underlying faults in devices like electronic circuits, the agent will use a model of *itself* to facilitate the reasoning process that leads to a diagnosis of the cause of failure.

The ability to diagnose planning failures in general depends critically on having a decision-making model that is rich enough to account for a wide variety of

[1] We have previously presented [Collins *et al.*, 1991] a model of the learning process involving three stages – reconstruction (of the causal history of the failure), diagnosis and repair.

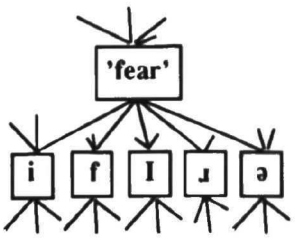

Figure 1: Part of a component network

possible errors. The construction of such self-models is thus a vital part of any attempt to construct agent-models capable of failure-driven learning. Previous attempts to construct such models have for the most part depended on a number of highly domain-specific assumptions [Collins *et al.*, 1992]. Here we describe aspects of what is intended to be a general model of attention-focusing and decision-making in an intelligent agent, and show in detail how this model can be applied to handle a complex example.

Component Architecture

A model of decision-making that is to be used to support learning must represent the decision-making system in enough detail to enable precise[2] characterizations of a failure; this makes it possible to implicate a relatively small portion of the decision-making mechanism, which in turn makes the task of considering how that portion might be changed manageable.

In our model, decision-making is carried out by a network of semi-autonomous components, each responsible for executing some specialized subtask. For instance, a component responsible for detecting a specific word whenever it is uttered near the agent might be connected to components that detect individual phonetic elements, each present in one or more pronunciations of the word (see figure 1). The information flow between these components is bidirectional. The word-detector receives information useful for inferring whether the word has been uttered, and also provides expectations to phoneme detectors which may be used to simplify the processing of sensor data. We will consider such a component in greater detail.

[2]The maximum useful specificity depends on what repair strategies are available. In repairing a faulty circuit, it is useful to know which component needs to be replaced, but probably not useful to know how the component is broken. In repairing a faulty decision-maker, it is useful to know enough about a failure to select a repair strategy to apply [Owens, 1990, Freed *et al.*, 1992].

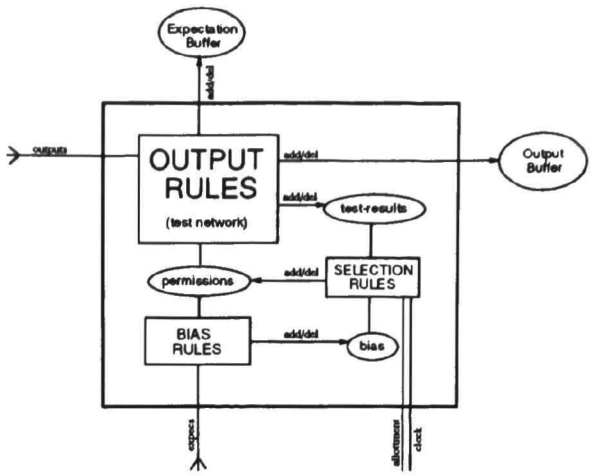

Figure 2: Generic Component Structure

Internal Structure of Components As diagrammed in figure 2, the components of our model incorporate three sets of rules: *output rules*, *bias rules* and *selection rules*. Output rules enable a component to make the specialized inferences that define its function. When it is determined that each condition of an output rule is satisfied a specified value is placed on the component's output buffer, making it available for use by output rules of other components. Conversely, if a condition becomes unsatisfied and the associated value is already on the output buffer, then the value is removed. For example, consider how a component whose function is to identify intervals of sound sensor data corresponding to the spoken word 'fear' should behave as the agent hears the well-known phrase, "We have nothing to fear but fear itself."

Several output rules from a 'fear' detecting component are shown in figure 3. Each specifies that when certain sounds (phonemes) register at specified (relative) intervals in the agent's aural sensor data, a value corresponding to a pronunciation of the word 'fear' should be placed on the output buffer. The word-detector can test whether a phoneme has registered in its sensors by matching the appropriate rule condition against the contents of a phoneme-detector's output buffer.[3]

Each output-rule condition in our example rules is associated with a pair of variable sound parameters

[3]Because our purpose is mainly to enable (and present) a planning architecture which may be easily adapted to deal with attention focusing problems, our treatment of this example takes a substantially idealized of view of speech analysis, dealing with only a few of the many legitimate complications. To the extent possible, we have made our model consistent with current phonological research [Blumstein, 1981].

IF f(e,f) ∧ i(g,h) ∧ r(a,b) ∧ h=a ∧ f=g
THEN *add-to-output-buffer*(midwestUS,e,b)

IF f(e,f) ∧ i(g,h) ∧ I(c,d) ∧ h=c ∧ f=g
THEN *add-to-output-buffer*(boston,e,d)

IF f(e,f) ∧ i(g,h) ∧ ə (i,j) ∧ h>i ∧ h<j ∧ f=g
THEN *add-to-output-buffer*(southUS,e,b)

Figure 3: 'fear'-detector output rules

representing the start and end of the data interval to be tested for the sound. Ordering constraints supplied with the rule can be used to limit the amount of sensor data analysis needed to locate a phoneme. For instance, if an expectation that the word 'fear' will occur has been derived from midsentence recognition of the phrase in which it appears, any information about the location (in the sensor data) of the first phoneme /f/ can be used to constrain search for the second phoneme /i/. Of course, the component which performs the search, the /i/-detector, must somehow learn of this constraint from the 'fear'-detector. We enable this in the following way: every time a parameter of an output-rule condition becomes specified, that specification is placed on the the *expectation buffer* to be used as bias by other components.

To be useful in conserving computational resources, expectations generated by other components must be translated into useful constraints on variables. This translation process is performed by the component's *bias rules* which encode dependencies between values likely to be found in the expectation buffers of specified components and parameter values of output rules. Once generated, bias on one parameter (such as the time-location of an /f/) should be propogated to other parameters (such as the time-location of an /i/). Constraint propogation is facilitated by representing output rules in a compiled form such as the rule network depicted in figure 4. The output rules are thus treated as a set of tests to be run on the output buffers of other components; when a certain set of tests (corresponding to the conjuncts of an uncompiled rule) are passed, a give output value *thresholds* and is placed on the component's output buffer. Tests in the network may be run in any order.

The ability of an agent to make efficient use of its attentional resources depends in part on whether individual components make effective use of their share of computational resources. One way

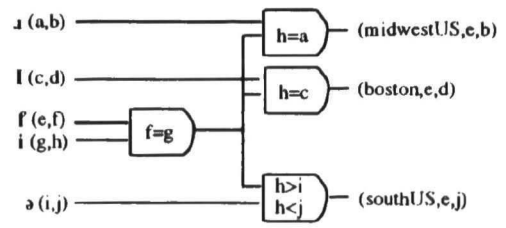

Figure 4: Compiled Output Rules

to manage resource use is for components to be discriminating about which tests in the output rule network to run and which bias rules to execute; those tests and rules that are least likely to help the component carry out its function will be given lower priority are thus least access to the components resources. *Selection rules* determine which tests and rules to run. In our example, a 'fear'-detector's selection rules, having determined the dialect of a speaker, will improve that component efficiency by giving first priority to tests associated with the pronunciation in that dialect and least priority to tests associated exclusively with other pronunciations.

Selection rules must take account of a number of factors, including:

- Whether to run more tests or generate more bias.
- Whether the information produced by a previously run test or bias rule is likely to have become obsolete; selection rules must weigh the need to produce more information against the need to reevaluate the old.
- The number of outputs that may be ruled in or out by running a test; preference should be given to the most discriminating tests.
- Leverage on potential productivity supplied by current test results and bias; preference should be given to tests most likely to cause some output to threshold.
- The expected cost of running a test or bias rule vs. the cost of other tests and bias rules.

Selection rules determine how the component uses its resources by producing a set of *permissions* that enable specified output tests and bias rules to execute using a supplied set of variable constraints.[4] As these rules more effectively take into account the above listed factors, the attentional performance of the component, and thus of the agent as a whole, improves.

[4] Alternately, permissions may supply bias-derived truth-values to be used in place of running a test. This allows components to compensate for noisy or missing data and to trade off some accuracy for speed/efficiency. E.g. when expecting the word 'fear,' the relevant detector may choose to test for only the first and last phoneme, simply assuming the presence of the middle one.

Component failures In carrying out its responsibilities, each component depends on a number of other components to provide valid information. When a component provides faulty information, that failure may propogate and cause a planning failure for the agent as a whole. One way to prevent recurrence of a planning failure is to fix the source of the problem; this requires an analysis of the way in which a particular component failed. As listed below, components can fail in a number of ways; such failures may result in planning errors corresponding to an attention-focusing failure.

- Output rules don't license a valid output or do license an invalid one — in this case, the agent is not knowledgable enough to perform some process no matter how resources are allocated. Attention-focusing failures occur when an invalid output results in an unproductive diversion of computational resources in a client component.
- Bias rules don't license some useful constraint (or truth-value assumption), making it impractical or unproductive to run a critical test.
- Bias rules license an unwarranted constraint or value assumption, causing a critical test not to be run (over certain parameter values).
- Selection rules fail to run a critical test in time — available constraints would have enabled a useful output.
- Selection rules fail to rerun a test in time to retract an obsolete output. The output is used to license an invalid inference (output or expectation) in another component.
- Selection rules fail to run a bias rule in time to make a critical test cost-effective
- Insufficient resources are available to the component to generate a critical output using any good selection strategy.
- A component interacts pathologically with a downline component — e.g. by flooding it with correct but irrelevant information [Freed *et al.*, 1992]

Each of these kinds of failure may be associated with one or more general strategies for modifying the component so as to prevent its recurrence.[5] For instance, a failure of bias rules to license a useful constraint can be dealt with by using explanation-based learning [DeJong and Mooney, 1986] to create a new bias rule. [Krulwich, 1991] describes such a process in detail.

[5]Failure categories have been used as indices to repair strategies in other domains including strategic planning [Collins, 1989] and device repair [Goel, 1991]; see also [Owens, 1990]

An Example

After riding the Tokyo subways for several weeks, one of us became inured to the novelty and decided to read a novel while waiting for his stop. Despite regular public address (PA) announcements containing the name of the next station, the rider missed his stop.

How might the rider explain his failure? A variety of common sense explanations seem plausible. For instance:

Story 1: I never heard my station announced. It probably was announced, but it didn't get my attention.

Story 2: I remember hearing my station announced, but somehow this didn't strike me as important. I must have been too absorbed in my book.

Story 3: I wasn't sure whether I heard my station, so I decided to check the station-sign once it became visible out the window. I then became absorbed in my book and forgot to do this.

In each failure story, the rider failed to get off the train because some easily obtained information was not taken into account. Such failures, which can be described as failures to "pay attention," can each be traced to an underlying failure in the model of sound-interpretation mechanisms of the agent's decision-making apparatus. We will now consider several of the components in this model and the accounts they provide for these stories of failure.

Interpreting Perceptual Information

Our model of perceptual interpretation assumes a hierarchy of *detectors* that process the data in stages. For example, a vocalization-detector, in our model of aural interpretation, may provide information to a b-detector, which may in turn inform a detector responsible for identifying the sound sequence ba when it occurs. This detector, along with several others, will inform a component responsible for detecting the sound of the station called 'Takadanobaba'.[6] This process is graphically depicted in figure 5.

The agent's aural interpretation machinery must be constructed so that contextual knowledge can be brought to bear on the interpretation task. For example, the fact that the agent is trying to hear a name spoken by a Japanese person and broadcast over an intercom should produce some expectations

[6]Such a model appears to be consistent with phonological research indicating that humans employ specialized detectors in interpreting human speech. See, e.g., [Eimas and Corbit, 1973], which presents evidence for *vocalization* detectors that allow listeners to discriminate vocalized phonemes such as /b/ from non-vocalized phonemes such as /p/.

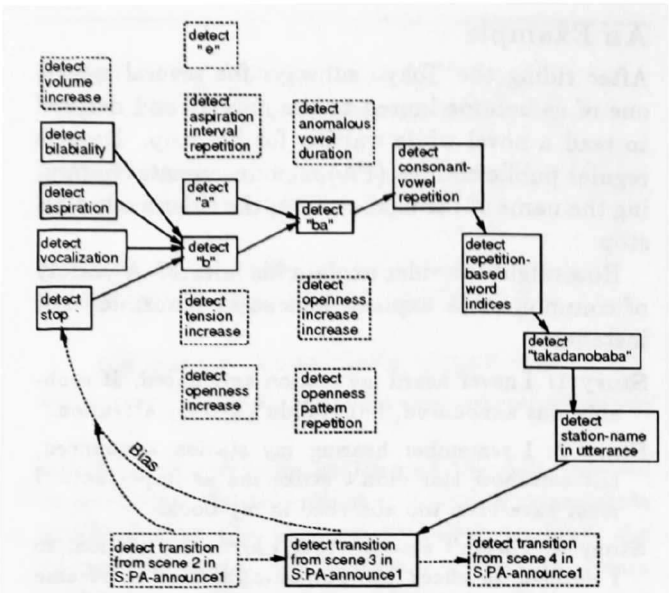

Figure 5: Aural interpretation components

about how the announcement will sound, which should in turn be employed to bias components of the interpretation process. Since the listening agent expects to hear Japanese, the a-detector might conform its analysis to expectations about Japanese vowel sounds, e.g. that unlike English vowel sounds these persist for an approximately equal duration.[7] For such facts to influence the behavior of the a-detector, they must be translated into meaningful advice about how to best process sensor data, for example, "only look for /a/ sounds having durations within [some specific range of times]." As previously discussed, this translation task is the responsibility of the detector's bias rules.[8]

We will characterize the bias on a detector as the expectation that target stimuli will fall within a certain range on a certain dimension, for example that the duration of a Japanese /a/ will fall within a certain range. In effect, the bias determines the range over which the detector must search, meaning that the narrower the range of the bias, the less costly it will be to run the detector. On the other hand, the narrower the range of the bias, the more likely the detector is to miss examples of the stimuli it is meant to detect. There is thus a tradeoff between the expense of running a detector and its hit rate.

If an instance of the detector's target concept falls outside of the range of its bias, the detector will miss that instance. This failure may lead to the failure of a higher-level detector that uses this detector as input. For example, if the detector for the /a/ sound fails to trigger during the announcement of the Takanobaba station, this may cause a failure to detect the word 'Takadanobaba', which may in turn cause a failure to detect the fact that the agent is at Takadanobaba station. In this way, the failure of any particular detector may propagate arbitrarily far up the network. The system thus has an obvious motivation to bias its detectors as broadly as possible. However, this is balanced by the fact that the broader the bias, the more resources consumed by the detector. For example, in attempting to detect the word 'Takadanobaba', the agent might bias its phoneme detectors so broadly that it is in effect carefully attending to every vocalization that is made during its train ride (and perhaps other sounds as well). While this will make it very likely that the agent will detect the name of its station, it will also draw so heavily on the agent's cognitive resources that it will be very difficult for the agent to perform other tasks such as reading a book.

Let us consider the stories of our example in light of this discussion. After the agent enters the train and finds a seat, it is expected that a station announcement in Japanese will be heard. Biasing mechanisms identify these factors as relevant, then translate them into detector-specific advice on what to look for in the way of (e.g.) minimum phoneme duration, dynamic range and maximum pitch fluctuation. The detectors are then given a certain share of the cognitive resources of the agent, which is severely limited, because the agent is attempting at the same time to read a book. The combined effects of the inaccuracy of the original bias and narrowing of that bias due to resource limitations causes the relevant detectors to miss the critical station announcement and ultimately causes the failure of the plan.

The agent's interpretation of the failure will depend on where failures occurred in the component network. For example, suppose that the agent's failure to exit the train was caused by the lack of a bias rule in the a-detector that would have enabled the detector to translate the expectation that the PA announcement would be spoken in Japanese into suitable constraints on its search of sensor data. The lack of constraint leaves the detector unable to cope with the diminished resources it has available due to

[7] Japanese is a syllable-timed language.

[8] High level expectations concerning (e.g.) which language an agent should listen for may be derived from schemas, or scripts [Schank and Abelson, 1975], that the agent is tracking. E.g. A script describing a typical ride on a Japanese train will include a scene in which a PA announcement, spoken in Japanese, conveys the name of the next station at which the train will stop. See [Hammond *et al.*, 1990] for a discussion of how schemas may be used as a source of expectations in a hierarchical agent architecture.

the agent's reading task, causing it to fail to detect an instance of its target concept and, thus, unable to provide knowledge of this occurrence to the higher-level **ba**-detector. This causes the **ba**-detector to fail, which in turn causes other failures including, ultimately, a failure to recognize that the agent has reached its destination and should therefore exit the train. In this instance, since the initial component failure prevented the 'Takadanobaba'-detector from carrying out its responsibilities, the agent experiences the failure as described in story 1: *I didn't hear my destination announced. It probably was announced but it didn't get my attention.* If instead, the intial failure occurred in between detecting the sound of the station name and detecting the imminent need to exit the train (the schema transition), the agent would have experienced the failure as described in story 2: *I remember hearing my station announced, but somehow this didn't strike me as important. I must have been too absorbed in my book.*[9]

The agent can respond to its failure to exit the train at the right time by trying to learn how not to repeat its mistake. In the case of the failing a-detector, this means learning a new bias rule which relates an expectation that Japanese will be spoken to an appropriate set of constraints on its output rule tests. The learning process proceeds in several steps. First, diagnostic mechanisms compare the actual behavior of components to their ideal, intended behavior as described in the model. When a component is found which may be considered the source of the problem (see [Birnbaum *et al.*, 1990] and [Owens, 1990]), a repair strategy is indexed based on the way in which the component failed [Freed *et al.*, 1992]. Finally, explanation-based learning mechanisms employing a model of the component they are to modify are invoked to implement the indexed repair strategy [Krulwich, 1991].

Acknowledgements: We thank Larry Birnbaum, Will Fitzgerald, Cheryl Jacques, Bruce Krulwich and Greg Siegle for many useful discussions. This was supported in part by the Office of Naval Research under contract N00014-89-J-3217, and in part by the Defense Advanced Research Projects Agency, monitored by the Air Force Office of Scientific Research under contract F49620-88-C-0058. Thi Institute for the Learning Sciences was established in 1989 with the support of Andersen Consulting, part of the Arthur Andersen Worldwide Organization. The Institute receives additional support from Ameritech, an Institute Sponsor, and from IBM.

[9]The experience of story 3 arises from a failure in the process of organizing a reponse to the detection of the imminent scene transition which involves components of the gaze control apparatus. Unfortunately, we do not have the space to describe this portion of our agent model.

References

[Birnbaum *et al.*, 1990] L. Birnbaum, G. Collins, M. Freed, and B. Krulwich. Model-based diagnosis of planning failures. In *Proceedings of the Eighth National Conference on Artificial Intelligence*, pages 318–323, Boston, MA, 1990.

[Blumstein, 1981] Stevens K.N. Blumstein, S.E. Phonetic factors and acoustic invariance in speech. *Cognition*, (10):25–32, 1981.

[Collins *et al.*, 1991] G. Collins, L. Birnbaum, B. Krulwich, and M. Freed. Plan debugging in an intentional system. In *Proceedings of the Twelfth International Joint Conference on Artificial Intelligence*, pages 353–358, Sydney, Australia, 1991.

[Collins *et al.*, 1992] G. Collins, L. Birnbaum, B. Krulwich, and M. Freed. The role of self-models in learning to plan. Technical Report 24, Northwestern University, The Institute for the Learning Sciences, 1992. Also to appear in *Machine Learning: Induction, analogy, and discovery*, A. Meyrowitz and S. Chipman, eds., 1992.

[Collins, 1989] G. Collins. Plan adaptation: A transformational approach. In K. Hammond, editor, *Proceedings of a Workshop on Case-Based Reasoning*, Palo Alto, 1989. Defense Advanced Research Projects Agency, Morgan Kaufmann, Inc.

[Davis *et al.*, 1982] R. Davis, H. Shrobe, W. Hamscher, K. Wieckert, M. Shirley, and S. Polit. Diagnosis based on description of structure and function. In *Proc. AAAI-82*, pages 137–142, Pittsburgh, Pa., August 1982.

[DeJong and Mooney, 1986] G. DeJong and R. Mooney. Explanation-based learning: An alternative view. *Machine Learning*, 1:145–176, January 1986.

[Eimas and Corbit, 1973] P.D. Eimas and J.D. Corbit. Selective adaptation of linguistic feature detectors. *Cognitive Psychology*, 4:99–109, 1973.

[Freed *et al.*, 1992] M. Freed, B. Krulwich, L. Birnbaum, and G. Collins. Reasoning about performance intentions. In *Proceedings of the Fourteenth Annual Conference of the Cognitive Science Society*, pages 242–247, Bloomington, IN, 1992.

[Goel, 1991] A. Goel. A model-based approach to case adaptation. In *Proceedings of the Thirteenth Annual Conference of The Cognitive Science Society*, pages 143–148, The University of Chicago, July 1991.

[Hammond *et al.*, 1990] K Hammond, T. Converse, and Martin C. Integrating planning and acting in a case-based framework. In *Proceedings of the Tenth International Joint Conference on Artificial Intelligence*, pages 292–297, 1990.

[Hammond, 1989] K. Hammond. *Case-based planning: Viewing planning as a memory task.* Academic Press, San Diego, CA, 1989.

[Krulwich, 1991] B. Krulwich. Determining what to learn in a multi-component planning system. In *Proceedings of the Thirteenth Annual Conference of The Cognitive Science Society*, pages 102–107, Chicago, IL, 1991.

[Minsky, 1986] Marvin Minsky. *The Society of Mind.* Simon and Schuster, New York, 1986.

[Owens, 1990] C. Owens. Representing abstract plan failures. In Proceedings of the Twelfth Annual Conference of the Cognitive Science Society, pages 277–284, Cambridge, MA, July 1990.

[Schank and Abelson, 1975] R. C. Schank and R. P. Abelson. *Scripts, Plans, Goals, and Understanding.* Lawrence Erlbaum Associates, Hillsdale, NJ, 1975.

[Schank, 1982] R.C. Schank. *Dynamic Memory.* Cambridge University Press, Cambridge, England, 1982.

[Sussman, 1975] G.J. Sussman. *A Computer Model of Skill Acquisition.* American Elsevier, New York, 1975. based on Phd thesis, MIT, Cambridge, MA, 1973.