Heuristics for Inductive Learning

Steven Salzberg
Applied Expert Systems, Inc.
Five Cambridge Center
Cambridge, MA 02142
U.S.A.
(617)492-7322

Abstract

A number of heuristics have been developed which greatly reduce the search space a learning program must consider in its attempt to construct hypotheses about why a failure occurred. These heuristics have been implemented in the HANDICAPPER system [Salzberg 1983, Atkinson & Salzberg 1984], in which they significantly improved predictive ability while demonstrating a remarkable learning curve The *rationalization* process has been developed as a verification system for the hypotheses suggested by the heuristics. Rationalization uses the causal knowledge of the system to ascertain whether or not a hypothesis is reasonable. If the hypothesis is not supported by causal knowledge, it is discarded and another hypothesis must be generated by the heuristics. The resulting learning system, by integrating causal knowledge with heuristic search, has quickly gone from essentially random predictive accuracy to a system which consistently outperforms the experts at predicting events in its problem domain.

1. Introduction: why we need heuristics in inductive learning

Inductive learning systems are often faced with a common problem, which is that of how to search a large space efficiently to find a new rule that will improve their performance [Mitchell 1983, Michalski 1983]. In particular, this paper is concerned with systems that predict events in a given domain, and revise their rules whenever those predictions fail. The problem can be stated succinctly as: given that a set of events S occurred, that the system predicted a result R1, and that the actual result was R2, how can the system revise its own rules so that, in the future, it will predict R2 when it sees S or something similar to S? The reason this rule-revision process is difficult, as has been pointed out a number of times [e.g., Soloway& Riseman 1977, Salzberg 1983], is that the number of potential new rules can be astronomical if the domain is even mildly complex. Any event or combination of events in S could be hypothesized as the cause of the mistaken prediction. If the set of events S includes only ten items, over 1000 possible hypotheses exist, and as S increases, the number of hypotheses increases exponentially (to be precise, if n is the number of events in S, the number of hypotheses is **2 ↑ n-1).**

Part of this work was dona at Yale Univtrsity under tht support of contract F49620-82-K-0010 of tht U.S. Air Force Office of Scientific Rasaarch.

Therefore AI needs to apply its own special techniques to the problem of narrowing down the search space. One of the most well-known and most successful techniques for controlling search is through the use of heuristics [Lenat 1983], and one purpose of this paper is to describe some new, general heuristics that proved to be remarkably useful and efficient in an inductive learning system. These heuristics have been suggested in some instances by psychological research which points out certain heuristics that people commonly use when faced with the task of understanding a complex problem. The task of the learning system, HANDICAPPER, is to predict the winners of thoroughbred horse races. In earlier papers [e.g., Atkinson & Salzberg 1984], this system was described as one which kept track of horses and made predictions based on similarities between new horses and ones in its memory. The system has now been completely re-written and relies on an episodic memory which contains races it has seen previously, rather than horses. Because predictions are based on similarities between races, and because races have a large number of features (about 50) which can contribute to judgments of similarity, the number of possible hypotheses about what rules to change after a failure is much larger than in the earlier system. Therefore the heuristics currently used by HANDICAPPER had to be implemented before the system could perform even tolerably well. As we shall show, the system with its heuristics in place performs much better than that.

The second major point of this paper is the introduction of a new technique, one called *rationalization.* Rationalization is an integral part of the learning algorithm of HANDICAPPER. Its purpose is to provide a check on the hypotheses generated by the heuristics in the system. After a hypothesis is generated, it is passed to the rationalization module which checks the system's causal knowledge (this knowledge is hand-fed to the system) in an attempt to explain why the new hypothesis might be a reasonable correction to the system's developing model of the domain. Rationalization ascertains, via forward inferencing (i.e., forward in HANDICAPPER, but not necessarily in any system), that if the new hypothesis

had been present prior to the incorrect prediction, a correct prediction would have been made, and it also ascertains that this new hypothesis is consistent with what the system already knows. If a hypothesis is rejected by rationalization, then control is passed back to the heuristic search routines, which must then find a new hypothesis. This loop continues until a reasonable explanation of the failure has been found and added to the knowledge base.

2. The heuristics

First, let me just list the heuristics developed for HANDICAPPER. A number of the heuristics in the list below have been designed but not implemented, and they are marked N.I. All will be explained in the text that follows.

1. Unusualness
2. Inconsistency
3. Uncertainty
4. Conservatism
5. Strength adjustment
6. Faultiness(N.I)
7. Occam's Razor (N.I.)
8. Ambivalence (N.I.)
9. Proximity

An example application of each heuristic will be given along with its explication, and, wherever appropriate, evidence for the psychological validity of the heuristics will be presented.

2.1 Unusualness

One of the first things that people look for when trying to explain an unexpected event is something *unusual* in the situation. Unusual feature are simply features that do not occur as often as other features within a given context. The following example will help to illustrate. If, while I were sitting in my office, one of the window panes suddenly shattered, I would expect to find a rock or some other dense object that had hit it with considerable force. If no such feature, which my default causal model requires, were present to my senses, I would look for other unusual features. Perhaps one of my office mates had just clapped his hands loudly - if that were the only unusual event just prior to the window's breaking, then I might have to give serious thought to whether or not it could have caused the window to break. If he clapped his hands together another time and *another* window broke, then I would begin to feel strongly that something about his clapping action was breaking the windows. To clarify this example a bit more, let me discuss for a minute what I meant above by the phrase "default causal model." Before using Unusualness to locate a feature in the current scene which might have caused the window to break, I had to consult my knowledge of physical actions to see what actions normally result in broken windows. The body of knowledge which I call a default causal model is undoubtedly very large, but it can be organized so that searching it is very efficient. In this case, I would look at an entity in memory representing the action "glass breaking," and then trace backwards down whatever causal links I had attached to that entity to find possible causes. (The cause might also be found by looking elsewhere in a frame-like structure [Minsky 1975], if the processing model were using such structures.) If none of those causes were present in the current scene, then I would have to turn to my heuristics to find a potential cause. In the example above, I might have considered the possibility of a sonic boom after consulting my default causal model, although I would immediately discard it since I had not heard any loud booms when the window broke. Having failed to find a match between the suggestions of my causal model and the realities of the scene, I would use heuristics to decide that the clapping of someone's hands, which occurred immediately before the window broke, might have been the cause.

Unusualness is determined in HANDICAPPER by keeping statistics on how frequently every feature appears on any horse. In other words, the program keeps track of such things as how often the values of horses change, how often they get new jockeys, and so on. When a prediction fails, and the causal model cannot suggest an explanation, the feature of the predicted winner which is the most unusual is chosen as the first candidate to explain why that horse lost. Likewise, the most unusual feature of the unexpected winner is chosen to explain why it won. The most unusual feature out of a set of feture is the one which occurs least often. If a horse or a race has a particularly unusual feature (according to some threshold), then, that feature will be chosen as a likely cause of the failure. In future races, if that feature appears again in a similar context, it will be monitored closely to see if the earlier hypothesis about it was correct.

2.2 Inconsistency

When a program uses a reasonably large set of feature and rules to make its predictions, as programs in real world domains often do, it is quite possible that it can use rules to support its predictions that are *inconsistent* with one another. In other words, two of the reasons for the same decision may be contradictory in some way, in which case it is possible that neither of the rules should have been used in the first place. A learning program must remember these inconsistencies when they arise and avoid making the same mistakes again. The argument could be made that such contradictions could be noticed beforehand and

avoided. To notice everything of this nature, though, a program would have to check every possible interaction beforehand. With a large feature set, one does not want to have to examine how *all* features might conceivably interact: for one thing, it will often be the case that two features which are inconsistent will never appear together precisely because of their contradictory nature, and thus any time spent worrying about them is wasted. Furthermore, if there are many feature, the amount of wasted work might be prohibitively expensive. It seems better, rather, to let prediction failures guide the emendations to rules and memory.

An example of inconsistency and how it is used in HANDICAPPER can be found in [Salzberg 1983]. In summary, that example describes how a horse with two *good* features lost, because the interaction between those features was bad. The horse in question was (1) dropping down in value, which is good because it means the horse will be competing against easier competition, and (2) recently bought by a new owner, which is usually good because the new owner must have seen something good in the horse to buy it. However, since the owner dropped the horse to a lower class of race soon after buying it, exposing himself to a financial loss, something must have been wrong with the horse, so despite the two good features, the horse lost the race.

2.3 Uncertainty

If you have to decide which of many features was responsible for a prediction failure, and you think your reasoning was sound in using each of those features, then another heuristic available for choosing one is the *uncertainty* heuristic. The Uncertainty heuristic says to choose the feature you know the least about and assign responsibility for the failure to that feature. The amount you know about each feature depends, for example, on how many times you have seen it. The rationale for this rule is that if you do not know very much about a feature - if you know very little about what it causes, what causes it, what other features usually appear with it, etc. - then it is more likely that it caused something unexpected (i.e., the prediction failure) than some other feature with which you are more familiar. The Unusualness heuristic is based on a similar principle.

To illustrate, let's take an example. Suppose the domain is the stock market, and the task is to predict the movement of stock in a high technology company. Suppose further that the stock was predicted to increase 10 points, and that in fact it only increased 5 points. Among the many features considered relevant to the prediction were: (1) the announcement of a new product by the company, (2) the election of a new, more conservative President of the U.S., and (3) a steadily increasing market for the products the company had been producing. Of these three factors, (1) and (3) might be quite familiar, occurring with regularity in stock market analysis and having fairly consistent effects. On the other hand, (2) only happens very infrequently: new Presidents are only elected once every four or eight years, and only half of those (on the average) are more conservative than their predecessors. So although the business community, and the stock market experts in particular, might believe that a more conservative President should boost prices in high technology stock (perhaps because he will give more money to the defense industry), it is quite possible that the opposite will occur, or that, as in this example, the effect will not be as large as expected. The reason is that we were simply less certain about the effects of a new conservative President on stock prices because we were basing our knowledge on very few experiences.

Features single out by the Uncertainty heuristic, then, are chosen precisely because they have not been observed many times. It is possible for HANDICAPPER, after a feature has been assigned credit for a failure by the Uncertainty heuristic, to observe that feature many more times and no longer be uncertain about it, but the first few times a feature is observed one cannot be certain whether or not its effects will recur consistently.

2.4 Conservatism

The *Conservatism* heuristic says to choose the new hypothesis which requires the smallest changes to the domain model. This heuristic partly explains why Ptolemaic astronomers preferred building epicycles to adopting Copemican theory. For one thing, it is usually simpler and easier to make a modification to an old theory rather than start over with a new one, and often that is the correct approach. However, when too many modifications have already been made, the old theory starts to be so cumbersome that throwing it out makes explanations simpler, and that is the point where Occam's Razor, another heuristic, should apply. On the other hand, when you have a reasonably simple theory, you usually want to modifiy it slightly rather than start from scratch. As was pointed out to me, this heuristic is equivalent to "trusting the known" while some of the others are more like "distrusting the unknown." Look at the Copernican model again, for example. After it was first suggested that the planets revolve in circles around the sun, evidence accumulated through the observations of Tycho Brahe that their motion was not circular. The correct solution, drawing upon the Conservatism heuristic, was to retain the heliocentric model but to posit elliptical, rather than circular, orbits for the planets. That solution, in fact, was suggested, by Johannes Kepler, and future astronomers used the modified model.

This heuristic has its basis in psychological data on biases [e.g., Nisbett & Ross 1980], which show that people tend to cling to beliefs once they have formed them. Much of the psychological data is designed to show that people cling to biases inappropriately, but as the example just given shows, the Conservatism heuristic can be useful. In HANDICAPPER this heuristic is implemented as an "unwillingness" to throw out old hypotheses unless no simple modification is workable. For example, a horse may be predicted to win because its speed rating has increased for the last two races and it placed second its last time out. Suppose the horse loses: the program could throw out its old beliefs about speed rating and recent performances, or it could modify those beliefs by making them weaker, or it could find entirely new reasons why the horse lost. The last is the preferred hypothesis formation technique, but when nothing new can be found, the program simply reduces the strength of the rules concerning speed ratings and recent performances. Unless something happens which indicates that those rules were completely off base in the first place, the Conservatism heuristic produces the most reasonable new hypotheses. A final note: over-reliance on this heuristic will result in hill climbing behavior, since small modifications to the current rule set will make a system hover around local maxima.

2.5 Strength adjustment

The *strength adjustment* heuristics (SAH) are weaker than the heuristics described so far, because they do not narrow down the search space as much. However, they are useful in HANDICAPPER in conjunction with the other heuristics. There are actually two main rules included as part of the SAH: (1) weaken features which were responsible for an incorrect match, and strengthen features which would have prevented the match; and (2) strengthen features which, if they had been stronger, would have resulted in a correct match, and weaken features which contributed to preventing the correct match. Carbonell [Carbonell 1983] and Hayes-Roth [Hayes-Roth 1983] use very similar heuristics as the bases for their learning algorithms. In fact, these heuristics are the basis of much of AI concept learning work in general. What is meant by a "match" in the above statements can be explained as follows: for any new race, HANDICAPPER tries to find the best match of that event with a previous event (for a more detailed description of the weighting algorithm, see [Salzberg 1985]). Predictions are made based on the outcome of the earlier event Rule (1) above comes into play when an event is matched that predicts the wrong outcome: if the match had not occurred in the first place, another, perhaps better, prediction would have been made. Rule (2) applies when an event *was present* in episodic memory which would have led to the right prediction, but the new event failed to match it because the features the two events shared were not considered important enough. Any system that bases its learning on being reminded of events in episodic memory has to worry about how one event matches another; finding exactly the right event is the trick that will allow the system to make the best possible predictions.

2.6 Faultiness

One common phenomenon among people trying to figure out how some domain or object works is to place the blame for mistakes on things that have caused problems before. In other words, the *Faultiness* heuristic says to choose the feature which was at fault most recently or which is at fault most often. This heuristic could also be known as the "it always breaks on Mondays" heuristic, that is, the tendency we all have to attribute the blame to some feature we thought might have been at fault the last time something went wrong. There are times when this rule is wrong, of course, but there are also times when it is extremely useful at pointing out the precise problem with something while allowing you to ignore other features.

2.7 Occam's Razor

Occam's Razor is an old scientific heuristic, with a sound basis in human behavior. This heuristic claims that, given a choice among explanations, one should choose the simplest one, which usually means the one with the fewest conjunctive conditions (this rule often works in opposition to the Conservatism heuristic). There are many famous examples of this heuristic, most of them from the history of science, and probably the most famous one is the defeat of the Ptolemaic (geocentric) model of the solar system by the simpler, more elegant Copemican (heliocentric) model. Particular details of the former theory, such as epicycles, which were necessary to explain the retrograde motion of the planets in the geocentric model, were unnecessary in the heliocentric model. In fact, epicyclic movement was nicely explained by the Copemican model without any more complicated motion than the revolution of the planets around the sun. (Actually, this is not quite true. The Copemican model was originally more complicated than the Ptolemaic model, and it was Kepler who simplified things, by using elliptical orbits.)

2.8 Ambivalence

Ambivalence is a heuristic which says to look for the rule with the weakest or most ambivalent basis (here again the idea is to "distrust the unknown"). For example, I once predicted that an opponent of mine

was bluffing in a poker game, based on three factors which any experienced player will no doubt recognize: (1) the cards he had showing indicated that it was unlikely he had a good hand (the game was stud poker); (2) the cards I had showing were weak enough to lead him to believe that he might win, and (3) he avoided my eyes when I looked at him. It turned out that he was not bluffing, and I explained the failure by (3) above. Of the rules listed, (3) is the most ambivalent - the failure to make eye contact might mean that someone is bluffing, but it might also mean that he has a good hand and he does not want to give it away with his eyes.

2.9 Proximity

There is a wealth of psychological data pertaining to the *proximity* heuristic (summarized and explicated in [Nisbett & Ross 1980]). In fact, this heuristic is really two heuristics, the *temporal proximity* heuristic and the *spatial proximity* heuristic. The idea behind them both is the same: when looking for a place to assign responsibility for some event, choose the event closest in time (and prior) or closest in space (at the time of the event) to the event itself.

For example, spatial proximity is most useful when questions of physical causality are involved. If some physical object behaved in an unexpected way, then perhaps you should look for some other physical object near it as the cause. When you notice a scratch on your car door in a parking lot, the first thing you are likely to do is to look at the car next to you. If that car is parked closely, and if its door is in a position to indicate that it would have hit yours where the scratch is, then you attribute responsibility to the person who last opened the door of the other car. If some paint from the other door has scraped off onto yours, and the colors match, then you will probably feel quite certain that the other car has caused the scratch. Spatial proximity is a heuristic that is used often not only because it works, but also because it is easy to use. It is only natural when something unexpected happens for a person to look around for possible causes. Objects in physical proximity to the unexpected event are likely to be the first candidates in the search for explanations.

Meteorologists' attempts to explain tornadoes are a classic example of the joint use of the temporal and spatial proximity heuristics. The actual causes of tornadoes are not yet well understood, so groups of meteorologists in Texas and Oklahoma (part of the U.S. "tornado belt") spend their time chasing around storms to find out exactly what happens before a tornado. They keep track of all the meteorological cues they can while on a "hunt," including visual observations as well as measurements of pressure, temperature, wind speed and direction, humidity, and more. They hope that by tracking all the events that fit some relevancy criterion which occur in the proximity (temporal and spatial) of a tornado, they will be able to isolate the set(s) of events that cause the tornado. Since they do not have a good causal model of how a tornado develops, the temporal and spatial proximity heuristics are the best tool they have for focusing their observations.

3. Which heuristic to use first?

The question of how to decide the best order in which to apply these heuristics (in a given domain after a particular prediction failure) remains an open problem. My suggestion, and the implementation I have chosen for HANDICAPPER, is to apply them in order of usefulness. The advantage of this approach should be clear: the most useful heuristics are the ones which most often suggest a good hypothesis for why a failure occurred, while less useful heuristics will suggest hypotheses not quite as good (where "good" means they frequently make correct predictions in the future). For the current implementation, I was guided by experts in deciding which heuristics were the most useful, and the performance of the program (described later) supports the ordering chosen.

A more objective method of determining usefulness would be to have the program initially use the heuristics randomly, giving them all weights but starting with the weights equal. As time went by, the weights could be adjusted so that the heuristics which succeeded most often were assigned the greatest values. If the context in which a heuristics was used successfully were saved along with the fact that the rule worked, then even more specific knowledge about when the heuristic is applicable would be available. In this way, over time, a program would be able to apply its heuristic knowledge more and more appropriately, and the hypotheses it chose to explain its failures would steadily improve.

4. Rationalization

Once the heuristics have selected a feature as the potential cause of a prediction failure, that feature is passed to the *rationalization module,* which checks to make sure that the hypothesis is consistent with the program's causal knowledge. Rationalization works by forward chaining from the feature itself until a stopping point or dead end is reached. All features are connected in a bi-directional network which expresses how they may cause each other, and which includes other facts about horse racing. Stopping points in the network are explicitly marked: most of them are facts like "the horse is more likely to win" or "the horse is more likely to lose." Other features point to these facts, and the rationalization module forward-chains

(breadth first) until it either reaches one of these facts or it cannot chain any more (markers are placed to prevent looping). Here is an example of a rationalization that HANDICAPPER produced, for the horse Gallant Herb, in a race which it had predicted a different horse would win:

GALLANT-HERB has been dropping down in value of the last five races, therefore
the competition this horse is running against is gradually getting easier, therefore
the performance of this horse should be improving, therefore
this horse is better than others at this level, therefore
this horse will win over others at this level.

The important role of rationalization in the overall learning/explanation algorithm is that if HANDICAPPER fails to find a rationalization for some feature, it must return to the heuristics to find another hypothesis. In other words, suppose the heuristics suggest a feature that caused a horse to lose. If the rationalization module cannot find an inference chain which leads to the same conclusion, then control is passed back to the heuristics, which attempt to suggest another feature as the cause of the loss. This control loop continues until a good hypothesis is found or until the heuristics cannot suggest anything more (if this latter instance occurred, HANDICAPPER would consider the failure to be anomalous, but it has not happened yet). Rationalization, then, is the manner in which causal knowledge is used to insure that hypotheses generated by the system are plausible explanations of the observed results.

5. HANDICAPPER's performance

HANDICAPPER has been tested on a base of 46 races, and then re-tested on another 41 races. Initially it has no episodic memory, and the first race is used to establish one. The program's initial knowledge is constrained to a list of approximately 70 features, with no knowledge of how good or bad each feature is, and to the heuristics described in this paper. As processing continues, it makes generalizations whenever it makes a correct prediction, and it constructs new rules when it fails. Generalizations are of the form: if features A, B, and C occur on one horse, then that horse has a 43% probability of winning against a set of horses none of which have those same features. Since it began with very little knowledge, its performance on the test database was at first poor, but it quickly improved and easily outperformed the experts (a group of experts publish their predictions daily in the *Daily Racing Form,* the standard horse racing newspaper) over the entire set of races.

In the course of running through all 45 races (no prediction was made on the first race), HANDICAPPER predicted 11 out of 45 correctly, or 24.4%. The experts were only correct 11.7% of the time. If one considers the first 15 races as "training" races, HANDICAPPER does even better: 10 out of 30 correct, or 33.3%, which indicates that the program was learning. On the second database of 41 races, which came from a different track, the program (with no tuning to account for the differences in data from a different track) did slightly worse, but still substantially better than the experts. Apparently the heuristics described here, coupled with the rationalization procedure, combine to produce an algorithm which learns very successfully despite the complexity of the domain.

From a theoretical standpoint, more work needs to be done the bases of these heuristics. Certain underlying assumptions are common to several of them, and it might be the case that there is a smaller set of more general heuristics which could be isolated. The question of whether a system could *learn* the heuristics needs to be examined, as well. The next steps in the development process must be first of all to implement the remaining heuristics, and then to apply the algorithm to other, perhaps even more complex domains. Success in other domains will lend greater support to the claim that the application of general heuristics must be an integral part of inductive learning algorithms.

Acknowledgements

Thanks to David Atkinson for many months of useful discussions and exciting ideas on inductive learning. Thanks also to Eduard Hovy and Larry Birnbaum for incisive and enlightening comments on a draft of this paper. At APEX, thanks to Jim Stansfield for additional comments and suggestions.

References

Atkinson, D. and Salzberg, S.
"The Use of Causal Explanations in Learning." *Proceedings of CSCSI-84,* London, Ontario, 1984.

Carbonell, J.
"Learning by Analogy: Formulating and Generalizing Plans from Past Experience." In Michalski, R., Carbonell, J., and Mitchell, T. (eds.), *Machine Learning,* Tioga Publishing, 1983, pp. 137-162.

Dietterich, T., and Michalski, R.
"A Comparative Review of Selected Methods of Learning from Examples." In Michalski, R., Carbonell, J., and Mitchell, T. (eds.), *Machine Learning,* Tioga Publishing, 1983, pp. 41-82.

Hayes-Roth, F.
"Learning from Proofs and Refutations." In Michalski, R., Carbonell, J., and Mitchell, T. (eds.), *Machine Learning,* Tioga Publishing, 1983, pp. 221-240.

Lenat, D.
"The Role of Heuristics in Learning by Discovery: Three Case Studies." In Michalski, R., Carbonell, J., and Mitchell, T. (eds), *Machine Learning, Tioga* Publishing, 1983, pp. 243-306.

Michalski, R.
"A Theory and Methodology of Inductive Learning." In Michalski, R., Carbonell, J., and Mitchell, T. (eds.), *Machine Learning,* Tioga Publishing, 1983, pp. 83-134.

Minsky, M.
"A Framework for Representing Knowledge." In P. Winston (ed.), *The Psychology of Computer Vision,* New York: McGraw-Hill, 1975.

Mitchell, T.
"Learning and Problem Solving." Computers and Thought Lecture, *Proceedings of IJCAI-83,* Karlsruhe, West Germany, 1983, pp. 1139-1151.

Mitchell, T., Utgoff, P., and Banerji, R.
"Learning by Experimentation: Acquiring and Refining Problem-Solving Heuristics." In Michalski, R., Carbonell, J., and Mitchell, T. (eds.), *Machine Learning,* Tioga Publishing, 1983, pp 137-162.

Nisbett, R.E. & Ross, L.
Human Inference: Strategies and Shortcomings of Social Judgment. New Jersey: Prentice-Hall, Inc. 1980.

Salzberg, S.
"Generating Hypotheses to Explain Prediction Failures." *Proceedings of AAAI-83,* Washington, D.C., 1983, pp. 352-355.

Salzberg, S.
"Pinpointing Good Hypotheses with Heuristics." In W. Gale (ed.), *Artificial Intelligence and Statistics,* 1985 (forthcoming).

Soloway, E. and Riseman, E.
"Levels of Pattern Description in Learning." *Proceedings of IJCAI-77,* Cambridge, Massachusetts, 1977, pp. 801-811.

Sussman, G.
A Computer Model of Skill Acquisition. New York: American Elsevier, 1975.