89-19

89-19

Methodologies for Intelligent Systems, 4

Editor:

Zbigniew W. Ras

North-Holland

Methodologies for Intelligent Systems, 4

Proceedings of the Fourth International Symposium on Methodologies for Intelligent Systems held October 12-14, 1989, in Charlotte, North Carolina

Editor

Zbigniew W. Ras, Ph.D.
Department of Computer Science
University of North Carolina at Charlotte
Charlotte, North Carolina

North-Holland
New York • Amsterdam • London

Elsevier Science Publishing Co., Inc.
655 Avenue of the Americas
New York, New York 10010

Sole distributors outside the United States and Canada:
Elsevier Science Publishers B.V.
P.O. Box 211, 1000 AE Amsterdam, the Netherlands

This book is printed on acid-free paper.

Library of Congress Cataloging-in-Publication Data

International Symposium on Methodologies for Intelligent Systems (4th : 1989 : Charlotte, N.C.)
Proceedings of the fourth International Symposium on Methodologies for Intelligent Systems, held October 12-14, 1989, in Charlotte, N.C.
(Methodologies for intelligent systems ; 4)
"The symposium was...sponsored by UNC-Charlotte... [et al.]"—Fwd.
1. Artificial intelligence—Congresses. 2. Reasoning—Congresses. 3. Knowledge, Theory of—Congresses. 4. Logic, Symbolic and mathematical—Congresses. I. Ras, Zbigniew. II. University of North Carolina at Charlotte. III. Title. IV. Series: Methodologies for intelligent systems ; v. 4.
Q334.1576 1989 006.3 89-22991
ISBN 0-444-01516-7

Current printing (last digit):
10 9 8 7 6 5 4 3 2 1

Manufactured in the United States of America

A MULTISEARCH APPROACH TO SEQUENCE PREDICTION.

Pawel A. Stefanski and Jan M. Zytkow
Artificial Intelligence Center, George Mason University,
Fairfax, VA 22030.

ABSTRACT

In this paper we describe MS.SPARC, a multisearch system that predicts the next event in a sequence. Events are collections of attribute-value pairs. Predictions are made by application of regularities (theory) discovered in the sequence. As the regularity or theory becomes more complete, the prediction becomes more deterministic. Formulating a theory involves a number of activities, including derivation of auxiliary attributes, selection of focus for regularity detection, regularity generation, testing completeness of the theory, etc. The whole process is complex and heterogeneous, causing serious control problems as reflected in limited capabilities of the existing systems. In response to those control problems our system is constructed on an explicit combination of several searches, using multisearch methodology which has been developed recently to support the construction of complex AI systems. The multisearch approach allows for a better understanding of the problem structure, for expanding the system by addition of new searches and for simulation of other sequence prediction systems. Natural locations are created within each search for additional search operators and heuristics. In addition to the known regularity generation methods, MS.SPARC can discover regularities that are composed of partial regularities, that hold for subsets of all events.

1. MOTIVATION

Prediction of the next event in a sequence is a common and useful task. We concentrate on situations in which all or most of the information useful for prediction comes from the sequence itself. Stock market analysis based on detecting patterns in previous stock values falls into this category, along with other empirical extrapolation tasks. The capability for finding and applying a regularity in a sequence is considered diagnostic towards a broad spectrum of intellectual skills, since as a rule, any person working on an intelligence test is repeatedly asked for the next object in a sequence. While easy to formulate, the prediction tasks span a broad spectrum of difficulty, from trivial to the most challenging problems. Earlier AI approaches [7][8][9] were usually very narrow. Some of them considered only one simple attribute, like in predicting the next character in a sequence, or were limited to a class of regularities searched for by a "black box" tool, as in the case of regression analysis. Some recent approaches offer more complex techniques, applicable to events that consist of a number of parameters of various types [1][6][10]. Still, human performance on heterogeneous data is superior to the existing systems. Even if a computer system can perform better or faster than humans on some sequences, the scope of sequences on which any given system would fail entirely is still very broad.

Our motivation for developing MS.SPARC was twofold. First, we wanted to extend the scope of sequences for which successful predictions can be made. Second, we wanted to apply the multisearch methodology. We believe that structured, expandable, and flexible design that are produced as a result of multisearch approach are critical in overcoming the limitations of the existing systems, and can be useful in understanding the scope and efficiency of human performance. While we do not attempt to formulate a cognitive model of human behavior, we realize the typical steps that humans take in solving "the next in a sequence" problems and the typical human control. People usually start with checking simple hypotheses based on the initial data. Subsequently they introduce new attributes and more complex regularity patterns, trying until they either make the prediction or become lost and tired.

Published 1989 by Elsevier Science Publishing, Co., Inc.
Methodologies for Intelligent Systems, 4
Zbigniew W. Ras, Editor

2. PROBLEM DESCRIPTION

In many prediction tasks requiring general intelligence, such as standard intelligence tests or Eleusis card game [2], the main difficulty lies neither in understanding the elementary steps on the way to discovery, nor in the capability for generating them. Once a regularity has been detected, it is usually trivial to enumerate the necessary steps. The problem consists in choosing a search method that can systematically yet flexibly apply various simple steps. Ideally, such a method should try each solution only once, be able to reach simple solutions first, be driven by the data and intermediate regularities found in the sequence, and stop as soon as the complete theory has been found. Solving the prediction problem is difficult because the moves towards the solution are heterogeneous. We may say that they occur in different search subspaces and linking them together causes control problems. Global control is difficult because the search should be both systematic and efficient. We believe that *conceptual complexity* is responsible for the overall difficulty of the task to a larger extent than *computational complexity*. Cutting down on conceptual complexity is the major advantage of multisearch approach. Since prediction of the next event is a complex, multifaceted task, it is an interesting application for the multisearch approach. The problem is to systematically represent different facets as different problem spaces, to link these spaces together, and impose systematic yet efficient control.

3. MULTISEARCH METHODOLOGY

Because the multisearch methodology was presented at length elsewhere [3][11], we restrict ourselves to the few basics. In a multisearch system, different searches are explicitly described by separate *search templates*. Search templates are object types, which are defined in the course of multisearch program development. Instances of those templates, that are created at the runtime during the course of problem solving are called *search nodes*. They correspond to nodes in the search tree. The system is driven by the *search interpreter* , which performs the multisearch by creating the template instances, executing the code included in each instance, and maintaining the stack of active search nodes. All search templates include the same generic list of slots. Some slots store the code applicable at a particular phase of the search process. Other slots are used for storing data that identify a particular search state and the knowledge relevant to that state. Categorization of the code into search phases provides a uniform and useful structure. The programmer is free to decide what code to put in a particular slot.

Each search node is generated at the runtime as an instance of the corresponding search template. It inherits default actions and operators from its template, unless the default values are explicitly overwritten. The values of parameters are particular to the instance and are generated from the parent node, according to the circumstances. After a search node has been generated, a sequence of actions is performed possibly resulting in generation of other nodes, in performing various actions, adjusting heuristics at other instances, etc., until the execution terminates and the result is returned to the parent node. Figure 1 traces schematically the depth-first multisearch process, as implemented in MS.SPARC.

4. DESCRIPTION OF MS.SPARC

The *input* to the system is a finite sequence of events. Each event is described as a set of attribute-value pairs. All events share the same attributes. The attributes present in the input are called basic. MS.SPARC can derive new attributes from those currently present in the sequence, using transformations whose descriptions are in the knowledge base about the domain. Those new attributes are called derived. Derived attributes can fully participate in future transformations, therefore we can have a hierarchy of derived attributes. The system is provided with descriptions of all domains of attribute values and with generic transformations. They constitute the knowledge base of the system. In addition to basic attributes we introduce an independent attribute "position", which corresponds to the event number in the sequence. This attribute participates in transformations and regularity search together with other attributes, but no regularity is ever attempted for the position alone, since its next value is obvious.

The output consists of a set of regularities discovered in the input sequence. Formally, each regularity is an executable expression. It can be evaluated by the user or applied automatically to predict the next events in the sequence. Semantically, a regularity is a restriction

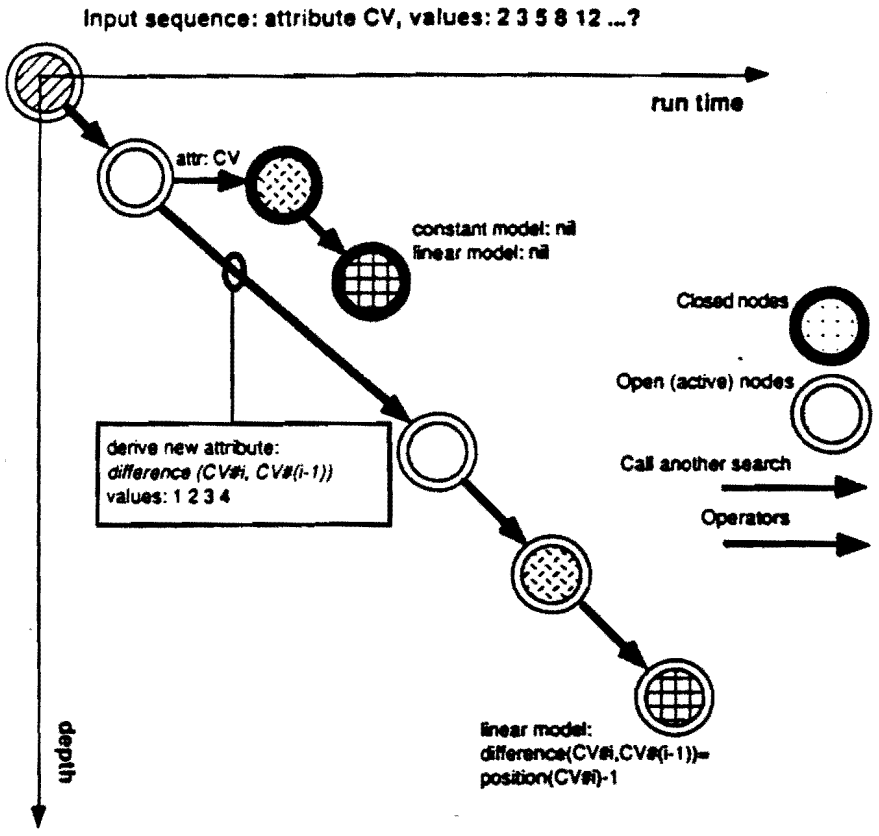

Figure 1: Depth-First Multisearch Schema in MS.SPARC

on the set of values of the basic attributes. A regularity may be limited to a basic attribute (like in: "The rank of each next card increases by 1") or to a combination of basic and derived attributes (like in: "The difference between ranks of two adjacent cards is 2 times the position of the first of them").

The MS.SPARC process starts by a call to the TRY-ALL-BASIC-ATTRIBUTES search (figure 2a). This simple search makes sure that each basic attribute is considered in further search for the regularities. To model human performance, the basic attributes can be ordered heuristically to match human preference for the attributes more likely to be tried first. The preference measure can be extended to the derived attributes.

For each of the basic attributes, the TRANSFORM search is called (figure 2b). TRANSFORM generates new attributes by application of several transformation patterns to the basic attributes or to the attributes that were generated earlier. A new attribute can be derived from one or more existing attributes. The TRANSFORM search tree has potential for an infinite growth since endless derived attributes can be generated, unless the search is heuristically constrained. TRANSFORM does not expand from a given node when a regularity has been found for the attribute generated at that node. The TRANSFORM search looks for the regularity indirectly, by calling the TRY-DOMAIN search (figure 2c). TRY-DOMAIN opens up the space of possible combinations of the existing attributes, trying the current attribute in different combinations with

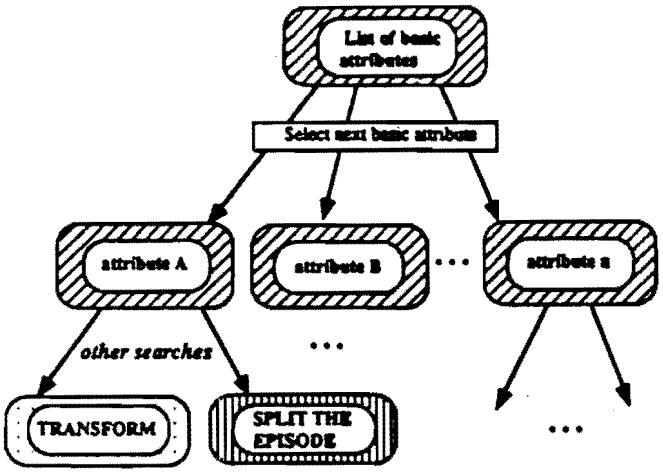

Figure 2a: Search CONSIDER-ALL-BASIC-ATTRIBUTES

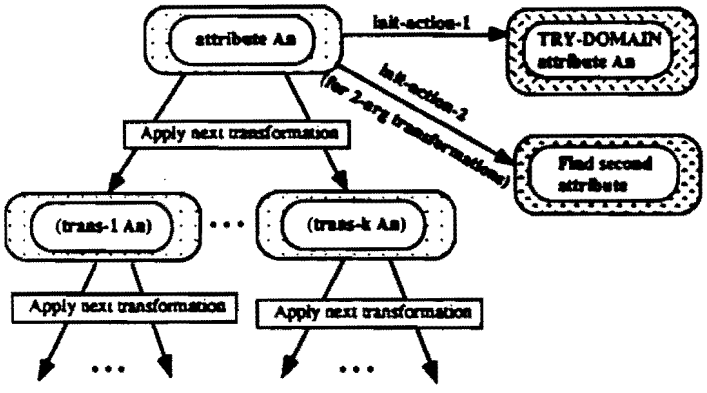

Figure 2b: Search TRANSFORM

other attributes, calling a FIND-REGULARITY search (figure 2d) for each combination of attributes. Different combinations are considered because a regularity may involve one or more attributes. The regularity *((RANK x) = 10)* involves one attribute, but *((DIFFERENCE (RANK x) (POSITION x)) = (POSITION (+ x 1)))* uses POSITION and DIFFERENCE which, in turn, is derived from the attributes RANK and POSITION, and very effectively allows to predict the value of RANK.

FIND-REGULARITY (figure 2d) takes the lists of values for the attributes that TRY-DOMAIN selected and tries to fit them to a number of regularity patterns or models. Modularity of design allows to use any number of models in FIND-REGULARITY. Currently, three models are implemented, ordered according to their predictive power and simplicity:

- a constant model, for instance *((RANK x) = 10)*, which says that all events in the episode have the same rank, 10.

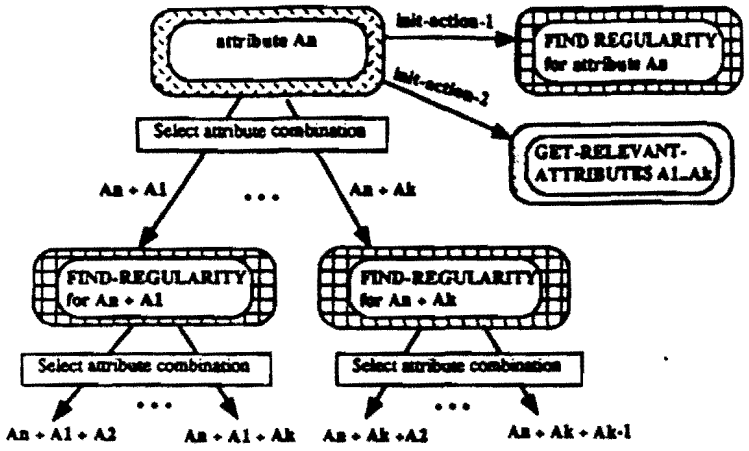

Figure 2c: Search TRY-DOMAIN

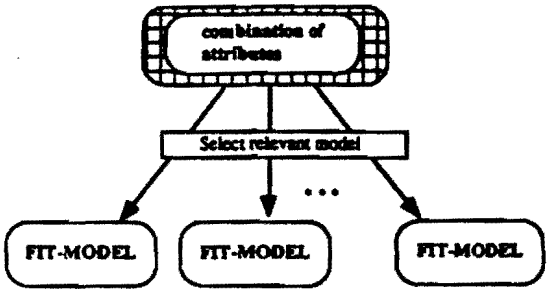

Figure 2d: Search FIND-REGULARITY

• a linear model, expressing a value of one attribute as a linear function of another attribute, for instance *((RANK x) = (+ (* 2 (POSITION x)) 1))*, restricting the rank of each event to one more than its position in the episode multiplied by 2.

• a periodic model, that involves two attributes, such as:
((((MODULO (POSITION x) 2) = 0) ((RANK x) = 2))
(((MODULO (POSITION x) 2) = 1) ((RANK x) = (3 5))))
which say that elements on even positions have rank 2, and those on odd, either 3 or 5.

Other models are being added to the system, for example a dnf model, where the regularity is expressed in terms of disjunction of conjunctions, like in:
((((COLOR x) = RED) ((MODULO (RANK x) 2) = 0)))
(((COLOR x) = BLACK) ((MODULO (RANK x) 2) = 1))))
which describes the sequence consisting of red even cards and black odd cards.

Even if TRY-DOMAIN can propose multi-attribute domains for FIND-REGULARITY, the number of attributes is practically limited by the nature of the models. Some models can take just one or two attributes, so that they are not tried when TRY-DOMAIN selects more attributes. TRY-DOMAIN proceeds until it either finds such a combination of the current attribute with the previously introduced attributes for which a regularity has been found, or until no more plausible combinations of attributes remain. Some heuristics used to select the plausible combinations of

attributes are described in the following section. When no more plausible combinations of attributes remain, the control returns to TRANSFORM, which evaluates any regularities that have been found.

Regularities are evaluated and compared based on their predictive power, measured as their non-determinacy. We present here one simple example of its use: let's assume, that the following rule was found for the basic attribute SUIT: *((COLOR x) = RED)*. It doesn't allow us to predict the SUIT attribute of the next card with the 100% certainty, but nevertheless, it is useful as a restriction on the set of allowable values (it excludes SPADES and CLUBS, leaving DIAMONDS and HEARTS). As it doesn't distinguish between two different values of the basic attribute SUIT, its non-determinacy measure is 2. The non-determinacy measure is a function of the regularity model, the attributes involved, and their values.

The final result of MS.SPARC's multisearch process is a set of regularities (theory), which can be used to predict the next event. A valid prediction can be any event whose attribute values satisfy all restrictions present in the developed theory.

5. SEARCH HEURISTICS IN MS.SPARC

Each space in a multisearch system can be searched by application of different search methods, and different spaces can be linked together in various ways. The multisearch approach generates systems that can be easily reconfigured and controlled heuristically in the interest of computational tractability, cognitive modelling, and the like. The tools include the control of the depth, the branching factor, selective expansion of active nodes, the order of node expansion, and many others. In the MS.SPARC system as in all multisearch systems, we can distinguish different kinds of heuristics:

(a) **global heuristics for the multisearch**, which are used to link the search templates and direct the overall search. Examples of those heuristics include:

- at what point and under what conditions a particular search calls another,
- the overall search control, for instance, depth first or breadth first multisearch control,
- predefined maximum depth level for each search, which can be modified, if no regularity was found for the previous value - similar to iterative deepening [4],

(b) **local heuristics for each search**, which determine the method of search (depth-first, breadth-first, best-first), overall constraints on search (maximum depth), and local search behavior specific to various nodes. The following examples include some heuristics used in MS.SPARC:

(i) global level heuristics, used to select rules

- global parameter FIRST-STOP which controls whether the search stops after the first regularity for a given basic attribute was found, or tries to continue until either "the best" regularity is found or all possibilities are exhausted.

(ii) heuristics for search TRANSFORM

- transformations are selected based on their simplicity: unary, then binary without parameters, then binary with parameters (like "lookback"),
- if the results of a transformation as applied to a sequence are outside of expected range, the transformation is abandoned (like in the case, when the results of DIVIDE transformation are not integers).

(iii) heuristics for search TRY-DOMAIN

- no derived attribute should be combined with any of its predecessors, e.g. (DIFFERENCE RANK POSITION) with RANK,
- if the constant regularity has been found for the earlier introduced attribute, there is no reason to combine it with the current one,
- if no regularity was found for the earlier introduced attribute, try to combine it,
- according to cognitive experiments, two (at most) different attributes can participate in any regularity. This heuristic also significantly reduces the search space for TRY-DOMAIN.

(iv) heuristics for search FIND-REGULARITY

- preference for the models that can produce simpler regularities.

(c) **domain-dependent heuristics**, represented in the knowledge base, such as:

- types, arguments, and priorities of transformations,
- allowable parameters and their priorities for each transformation (for example, "lookback" parameter for the "difference" transformation),

• unfeasible models for each domain (for example, it is worthless to look for the linear regularity for the attribute COLOR).

6. ANNOTATED EXAMPLE

Figure 3 presents the process of finding a regularity for the given sequence of numbers. Only those derived attributes which proved fruitful in finding a theory are shown. After applying the DIVIDE transformation the system realized that the results are all integers, which allowed it to continue down that path and try other transformations (this is one of the local search heuristics, described above). One of them was the DIFFERENCE transformation with lookback parameter equal 1. One more application of this transformation gives the constant regularity for the derived attribute, which can be than propagated back and returns deterministic prediction of the value for the next event.

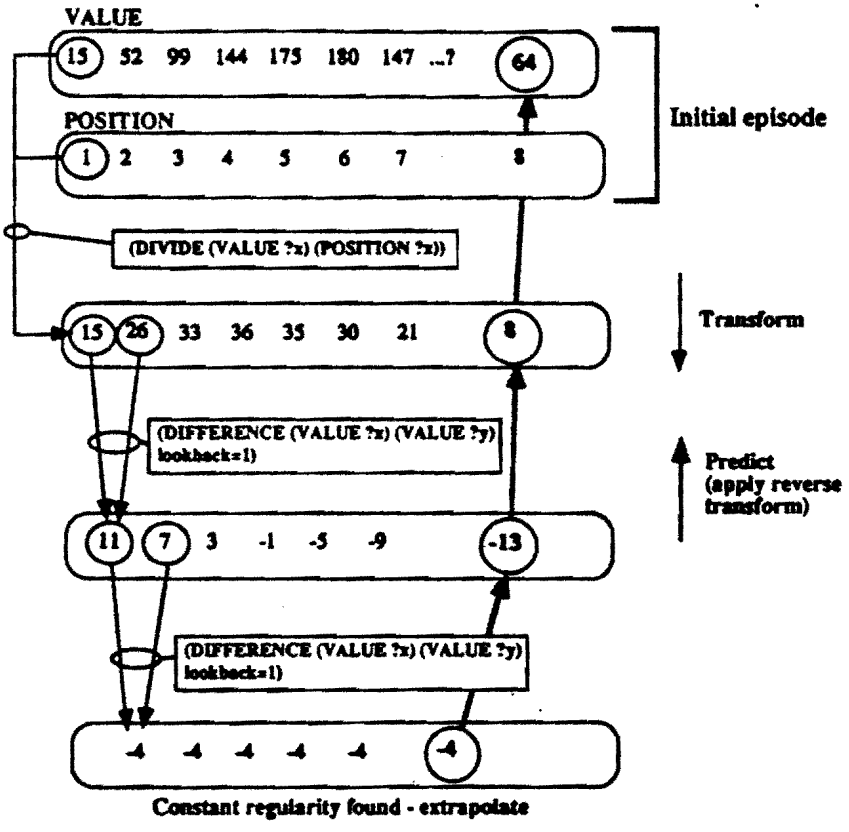

Figure 3: Example.

7. CONCLUSIONS

One of our main goals in constructing MS.SPARC was to test the multisearch methodology on a new, interesting domain, while at the same time to provide competitive solution to the prediction problem. As a result we obtained a system based on a clear structure which is flexible and easily modifiable. Many other systems can be simulated in MS.SPARC, by restricting the number of attributes and search mechanisms. Our system can be viewed both as a collection of tools for solving sequential prediction problems, and as a set of working systems which differ by some heuristics. Because of an explicit combination of several searches, we also achieved a broad scope of applications. Other important feanures of our system are:
- the core system is domain-independent, while all domain-specific knowledge resides in easily modifiable data structures (domain-dependent knowledge-base),
- the system can discover both non-deterministic and deterministic regularities,
- following human performance, the system discovers simple regularities first, and only if unsuccessful, gradually increases the complexity (as opposed to SPARC [6]),
- although not as flexible as Seek-Whence [5], our system can modify the search strategies being used, while working on the particular problem,
- system is both model- and data-driven. Search for regularities is limited by models available to the system, while on the other hand, data-dependent heuristics are used to select plausible transformations and models.
- MS.SPARC can easily incorporate more discovery methods by inserting new searches and adjusting data representation. It is therefore well modularized, and with this respect, we believe, superior to other systems.

ACKNOWLEDGEMENTS

Special thanks go to Joyce Ralston for valuable help and corrections to the earlier versions of this paper. This work was conducted in the Center for Artificial Intelligence at George Mason University, supported in part by the Defense Advanced Research Projects Agency under grant No. N00014-87-K-0874, administered by the Office of Naval Research, by the Office of Naval Research under grant No. N00014-88-K-0226, and by the Office of Naval Research under grant No. N00014-88-K-0397.

REFERENCES

[1] Dietterich, T.G. and Michalski, R.S., "Discovering Patterns in Sequences of Events", Artificial Intelligence 25(2) (1985) pp.187-232.
[2] Gardner, M., "On playing the new Eleusis, the game that simulate the search for truth", Scientific American, 237 (1977) pp.18-25.
[3] Jankowski, A., and Zytkow, J.M., "A Methodology for Multisearch Systems", Proceedings of 3rd International Symposium on Methodologies for Intelligent Systems, North-Holland, 1988, pp. 343-352.
[4] Korf, R.E., "Depth-First Iterative Deepening: An Optimal Admissible Tree Search", Artificial Intelligence, 27 (1985) pp.97-109.
[5] Meredith, M.J., "Seek-Whence: A Model of Pattern Perception", Rept. No.214, Computer Science Department, Indiana University, Bloomington, IN (1986).
[6] Michalski, R.S., Ko, H., and Chen, K., "Qualitative Prediction: The SPARC/G Methodology for Inductively Describing and Predicting Discrete Processes", in Dufour, P., and Lamsmeerde, A. van, (eds.), Expert Systems, Academic Press (1986).
[7] Pivar, M., and Finkelstein, M., "Automation using LISP, of Inductive Inference on Sequences", in Berkeley, E.C., and Bobrow, D., (eds.), The programming language LISP, Information International, Cambridge, Mass. (1964).
[8] Restle, F., "Theory of Serial Pattern Learning: Structural Trees", Psychological Review, 77 (1970), pp.481-495.
[9] Simon, H.A., and Kotovsky, K. "Human Acquisition of Concepts for Sequential Patterns", Psychological Review, 70 (1963) pp.534-546.
[10] Wong, A.K.C., and Chiu, D.K.C., "Discovery of Probabilistic Rules for Prediction".
[11] Zytkow, J.M., and Jankowski, A., "Hierarchical control and heuristics in multisearch systems", this volume.