89-13

MINING FOR KNOWLEDGE IN DATABASES: Goals and General Description of the INLEN System

Kenneth A. Kaufman, Ryszard S. Michalski and Larry Kerschberg

Artificial Intelligence Center
George Mason University
Fairfax VA 22030

ABSTRACT

The INLEN system combines database, knowledge base, and machine learning techniques to provide a user with an integrated system of tools for conceptually analyzing data and searching for interesting relationships and regularities in them. Machine learning techniques are used for tasks such as developing general rules from facts, determining differences between groups of facts, creating conceptual classifications of data, selecting the most relevant attributes, determining the most representative examples, and discovering equations governing numeric variables. The equations discovered are accompanied by conditions under which they apply. The above techniques are implemented as inference operators that a user can apply to a database and/or knowledge base in order to perform a given knowledge extraction function. Examples of three major inference operators are provided, specifically, for learning general rules differentiating between groups of facts, for creating conceptual classifications of facts, and for discovering equations characterizing numerical and symbolic data.

1. Introduction

This paper briefly describes the goals and general design of the INLEN system for conceptually analyzing databases and discovering regularities and patterns in them. The name INLEN derives from **I**nference and **L**earning, which represent two major capabilities of the system. INLEN integrates a relational database, a knowledge base, and a number of machine learning and inference capabilities. The latter ones enable the system to perform tasks such as creating conceptual descriptions of facts in the database, inventing classifications of data, discovering rules and unknown regularities, and formulating equations together with the conditions of their applicability. We present here a general system design and explain all the basic functions. Major operators, specifically those for determining rules from examples, for creating classifications, and for discovering equations, are illustrated by examples.

The motivating idea behind the INLEN system is to integrate three basic technologies - databases, expert systems and machine learning and inference - in order to provide a user with a powerful tool for manipulating both data and knowledge, and for extracting from that data and/or knowledge new or better knowledge. INLEN evolved from the QUIN system (**Q**uery and **I**nference), a combined database management and data analysis environment [Michalski, Baskin and Spackman, 1982; Michalski and Baskin, 1983; Spackman, 1983]. QUIN was designed both as a stand-alone

system, and as a subsystem of ADVISE, a large scale inference system for designing expert systems [Michalski and Baskin, 1983; Michalski et al., 1987; Baskin and Michalski, 1989].

In the last few years, new tools have been developed; in particular, more advanced inductive learning systems, e.g., AQ 15 [Michalski et al., 1986] and ABACUS-2 [Greene, 1988], and expert database systems [Kerschberg, 1986, 1987, 1988]. The above systems have influenced the development of INLEN. INLEN also draws upon the experiences with AGASSISTANT, a shell for developing agricultural expert systems [Katz, Fermanian and Michalski, 1986], and AURORA, a general-purpose PC-based expert system shell with learning and discovery capabilities, designed by Michalski and Katz [INIS, 1988].

2. INLEN System Design

As mentioned above, INLEN combines database, expert system and machine learning capabilities in order to create an environment for analyzing and extracting useful knowledge from a data and/or knowledge base. It includes ideas from the recently developed expert database technology to combine the storage and access abilities of a database system with the ability to derive well-founded conclusions from a knowledge-based system [Kerschberg, 1986, 1987, 1988]. INLEN integrates several advanced machine learning capabilities, which until now have existed only as separate experimental programs. Many learning systems are capable of but a small subset of what can be learned from factual data. By integrating a variety of these tools, a user will have access to a very powerful and versatile system.

The general design of INLEN is shown in Figure 1. The INLEN system consists of a relational database for storing known facts about a domain, and a knowledge base for storing rules, constraints, hierarchies, decision trees, equations accompanied with preconditions, and enabling conditions for performing various actions on the database and/or knowledge base. The knowledge base can contain not only knowledge about the contents of the database, but also metaknowledge for the dynamic upkeep of the knowledge base itself.

The purpose for integrating the above capabilities is to provide a user with a set of advanced tools for searching for and extracting useful knowledge from a database, for organizing that knowledge from different viewpoints, to test this knowledge on a set of facts, and to facilitate its integration within the original knowledge base.

Information in the database consists of relational tables (RTs), and information in the knowledge base consists of units called *knowledge segments*. A knowledge segment (KS) can be simple or compound. Simple KSs include rulesets, equations, networks and hierarchies. Compound KSs consist of combinations of any of the above, or combinations of simple KSs and RTs. The latter form may be used, for example, to represent a clustering that consists of groups of objects (represented as an RT), and the associated descriptions of the groups (represented as rules). Another example of such a representation is a relational table with a set of constraints and relationships among its attributes. Those constraints and relationships are represented as rules. Compound KSs also consist of directory tables that specify the locations of their component parts in the knowledge base or, in the case of RT components, in the database.

A justification for such knowledge types is that they correspond to natural forms of representing human knowledge, especially technical knowledge. Also, by distinguishing between these different forms of knowledge and selecting appropriate data structures to represent them, we can achieve greater efficiency in storing and manipulating such structures.

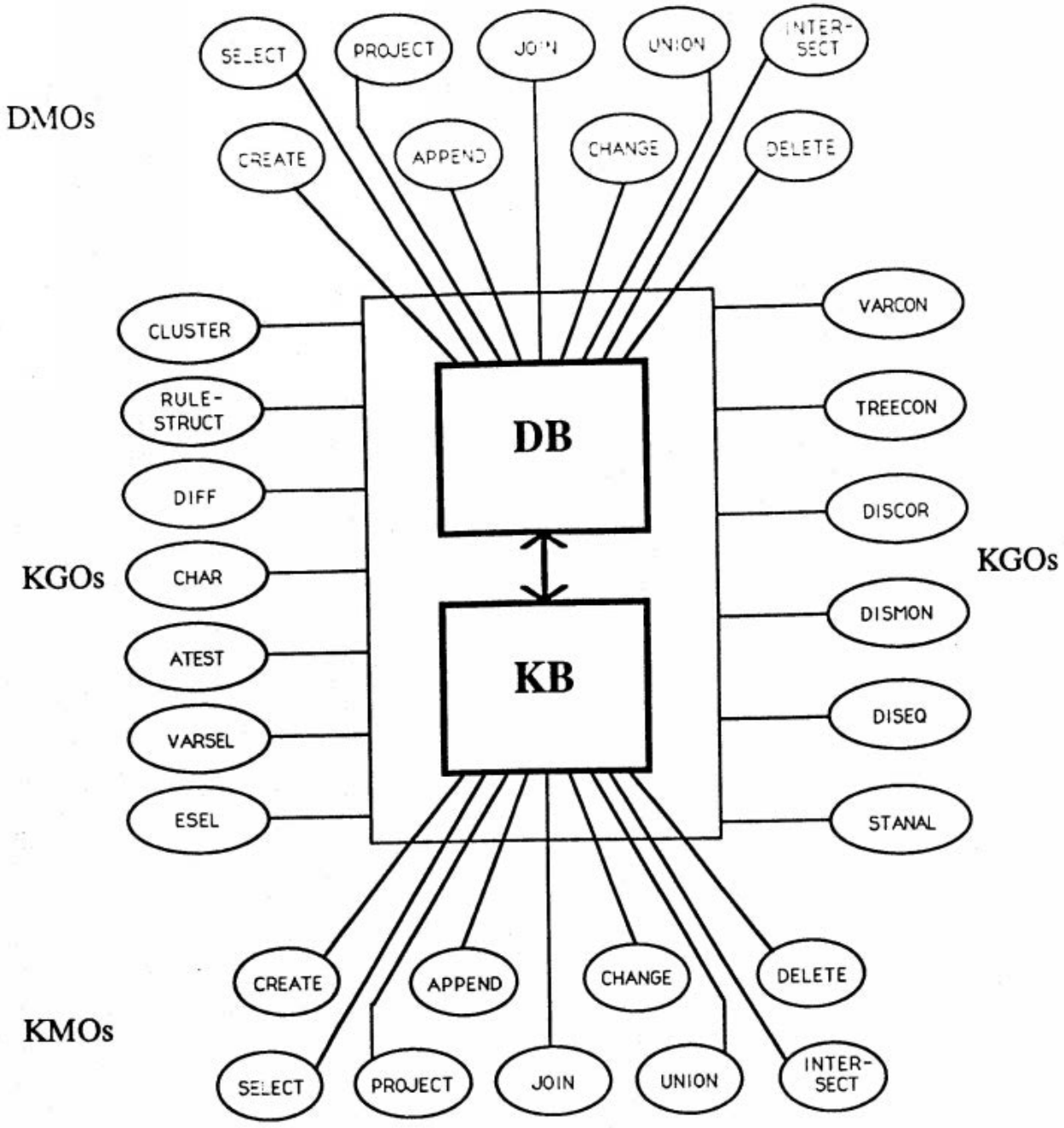

Figure 1. A Functional Diagram of INLEN

INLEN employs three sets of operators: *data management operators* (DMOs), *knowledge management operators* (KMOs), and *knowledge generation operators* (KGOs). The DMOs are standard operators for accessing, retrieving and manually altering the information in the database. Thus, they operate on RTs. The KMOs perform analogous tasks on the knowledge base, in situations in which manual input, access or adjustments are required. The knowledge generating operators interact with both the database and the knowledge base. These operators evoke various situations in which manual input, access and adjustments are required. The KGOs take input from both the database and the knowledge base. These operators invoke various machine learning programs to perform tasks such as developing general rules from facts, determining differences between groups of facts, creating conceptual classifications of data, selecting the most relevant attributes, determining the most representative examples, and discovering equations governing numeric variables. The results of KGOs are stored as knowledge segments. Examples of the performance of a few basic knowledge generating operators are given in Section 3.

A brief description of each of the DMOs, KMOs and KGOs follows.

Data Management Operators (DMOs)

The data management operators form a standard set of relational database operations for the purpose of manipulating the system's collection of facts. They are listed here for the sake of completeness.

CREATE generates a new relational table. It takes an attribute list as an argument.

APPEND adds a new tuple (row) to a relational table.

CHANGE alters some or all of the values in some or all of the tuples of a table.

DELETE removes rows or columns from a table, as specified respectively by SELECT or PROJECT operations. Alternatively, entire tables may be removed from the system.

SELECT retrieves a relational table from a database, and returns the complete table or part of it. The part represents the subset of its rows that satisfy criteria specified in the arguments of the operator.

PROJECT reduces a table by removing columns. Columns that are kept correspond to attributes specified in the arguments of the operator.

JOIN creates a relational table combining the columns of two tables. The rows are the subset of the rows of the Cartesian product of the two tables whose attributes satisfy criteria provided by the user.

UNION, performed on two tables with the same set of attributes, returns the set of tuples (rows) which appear in either of the two tables.

INTERSECT, performed on two tables with the same set of attributes, returns the set of tuples which appear in both of the input tables.

Knowledge Management Operators (KMOs)

The knowledge management operators are used to create, manipulate and modify INLEN's knowledge base, thereby allowing the knowledge base to be handled in a manner analogous to handling a database. Knowledge may take the form of simple or compound knowledge segments (KSs). Consequently, most of the knowledge management operators shown in Figure 1 are generalized for any of these forms. Unless otherwise specified, they should be thought of as operating on any KS, i.e., they can operate on rules, equations, hierarchies, etc.

The diverse representations of knowledge may be culled from the *same* database, and will therefore represent distinct viewpoints obtained using the knowledge generation operators. For example, a dynamical system whose behavior is governed by a set of differential equations could have its time series input-output behavior represented as a relation consisting of all measurable input-output variables. Each tuple would consist of the input-output variable value at some time. The KGOs could be used to create knowledge viewpoints such as functional and multi-valued dependencies from relational database theory, a set of decision rules, a causal and temporal semantic network, etc. Each of these viewpoints is valid, and should be managed by the KMOs.

Expert database tools and techniques can be used to manage the evolution of the combined knowledge/data base by incorporating knowledge discovered in the database. The arrow in Figure 1 linking the DB and the KB components represents such an interaction.

The knowledge base management operators listed below are depicted as analogues of INLEN's data management operators. Without intensive testing of the system in different domains, one cannot tell how useful these operators are, but they represent our first approximation based on the analogy to the data management operators. Further research may lead to the development of other operators, and also other knowledge representations, including the likely use of an object-oriented approach in which one data representation is replaced by an active link to the concept of a formula, a rule set, or some other representation.

Under the current design, these are the knowledge management operators and their functions:

> CREATE is used to generate a new knowledge segment, with a structure and set of attributes specified by the user. The KS will be empty until knowledge is added using either an APPEND operator or one of the knowledge generation operators.
>
> APPEND is used for the manual addition of new knowledge to a KS.
>
> CHANGE is used for the manual alteration of part of one or more items in a knowledge segment.
>
> DELETE is used to remove selected portions of a knowledge segment from the knowledge base. Alternatively, an entire KS may be erased by giving no qualifying conditions to the operator.
>
> SELECT is used to retrieve a knowledge segment from the knowledge base (and from the database in the case of component RTs.) Criteria may be provided to return only selected items (such as rules, subtrees, rows in tables, etc.) in this KS.

PROJECT is used to return a subset of a compound KS which ignores entire components (e.g. rulesets, decision trees, columns of tables) of the KS. The items specified in the operator's arguments will be included.

JOIN is used to combine a pair of simple knowledge segments or components of compound knowledge segments. For example, a set of rules and a data table can be united into a compound KS, or two rulesets may be combined by finding conditions in the first ruleset which are satisfied by decisions in the second ruleset. Rules may then be expanded by replacing the matching conditions in the first ruleset with the conditions leading to the corresponding decisions in the second ruleset.

UNION is applied to two or more knowledge segments of the same type. It generates a list of the elements present at least once in any of the segments.

INTERSECT is applied to two or more knowledge segments of the same type. It generates a list of the elements present at least once in each of the segments.

Knowledge Generating Operators (KGOs)

The KGOs perform complex inferences on knowledge segments in order to create new knowledge. It should be noted that the KGOs also consist of primitives (such as "save" and "retrieve") in order to facilitate access to the structures they generate. These structures will generally be compound KSs which include tables in the knowledge base that locate their other components.

Many of these operators work with or generate rules. Rules in INLEN consist of a decision part implied by a condition part. The decision part consists of a conjunction of one or more statements or actions, while the condition part consists of a disjunction of conjunctions, each consisting of one or more elementary conditions (for examples, see Tables 1 and 2.)

Under the current design, these are the basic KGOs employed by INLEN:

CLUSTER performs conceptual clustering of tuples in a relational table in order to create logical groupings of objects or events represented by the tuples. It also determines a set of rules characterizing the created groups. Specifically, the operator divides rows of a relational table into two or more groups, and returns a KS consisting of a relational table, similar to the input table, but also containing additional information indicating the groups, and a ruleset characterizing the individual groups. An example of this operator is given in Section 3. User-defined parameters may influence the creation of the groups. Detailed descriptions of the conceptual clustering algorithm that performs this operator are in [Michalski, Stepp and Diday, 1981; Stepp, 1983, 1984].

RULESTRUCT also performs conceptual clustering, but applies it to a ruleset, rather than to a relational table. A compound KS is returned consisting of the original ruleset with grouping information, plus a new ruleset to explain the grouping.

DIFF (Differentiate) takes two or more classes of objects (each object represented as a tuple in a relational table), and induces general rules characterizing the differences between the classes. The output KS consists of the ruleset created by the operator, and the object classes, represented by RTs. The AQ program that executes this operator is described in [Michalski and Larson, 1983]. The rules produced are called discriminant descriptions,

i.e., they specify sufficient conditions for distinguishing one class of objects from the other class(es).

CHAR (Characterize) determines descriptions characterizing a class of objects. This operator also falls into the domain covered by the AQ program mentioned above. Here, the emphasis is on finding characteristic rules describing all examples of a class of objects, without concern about the differences between this class and other classes. Output includes the initial class plus the generated descriptions.

ATEST tests a set of decision rules for consistency and completeness on a set of examples (specified in a relational table). Consistency implies that no event in the example space is covered by two different rules. Completeness refers to the condition that every possible example will be covered by the conditions applying to at least one rule. The output KS consists of the input rules, example sets, and a relational table containing ATEST's analysis. ATEST is described in detail in [Reinke, 1984].

VARSEL determines attributes in a relational table that are most relevant for differentiating between various classes of objects. Output consists of a rule describing the selection of the variables given the input classes, and the subtable generated by projecting on the chosen variables. By keeping only the most relevant attributes in the object (example) descriptions, one can significantly reduce the computation time required by the CLUSTER or DIFF operator [Baim, 1982].

ESEL determines the examples (objects) that are most representative for given classes. Promising examples are returned as output with a rule specifying the input classes and the chosen examples, while other examples are rejected [Michalski and Larson, 1978; Cramm, 1983].

VARCON applies mathematical operators specified in its argument in order to combine variables into useful composites. The output KS will consist of the new composite variables, and a rule specifying the original table, the mathematical operators, and the created variables. For example, VARCON can be used if the sum or product of two variables might be more useful than either individual value [Davis, 1979].

TREECON takes a set of rules or decision examples, and organizes them into a decision tree, which may be a more efficient way for storing and/or using the knowledge [Michalski, 1978; Layman, 1979].

DISCOR discovers correlations between the values of attributes in a set of examples. It is implemented as a standard statistical operation of correlation and returns a table of its results.

DISMON seeks out monotonic relations between attributes in a set of examples, and in doing so, may discover an interesting relationship within the data. It is an operator that is utilized in the DISEQ operator.

DISEQ discovers equations that describe numeric data in a set of examples, and formulates conditions for applying these equations. DISEQ returns a set of equations and the rules which determine when they apply. It is based on the ABACUS-2 system for integrated qualitative and quantitative discovery [Falkenhainer and Michalski, 1986; Greene, 1988]. ABACUS-2 is related to programs such as BACON [Langley, Bradshaw and Simon, 1983], FAHRENHEIT [Zytkow, 1987] and COPER [Kokar, 1986].

STANAL performs a statistical analysis of the data in order to determine its various statistical properties.

3. An Illustration of Selected KGOs: CLUSTER, DIFF, DISEQ

This section gives examples of how some basic KGOs work; specifically, the CLUSTER, DIFF and DISEQ operators.

CLUSTER

CLUSTER is capable of creating groupings of objects or events, and when used recursively, can generate an entire taxonomy. Unlike traditional clustering methods, CLUSTER also returns the rules that describe its grouping. The presented example is based on the results described in [Michalski and Stepp, 1983], involving the creation of a classification of microcomputers. Variables considered include the type of processor, the amount of RAM, the ROM size, the type of display, and the number of keys on the keyboard. Dividing the examples into two groups, the system grouped them according to RAM size and keyboard, while clustering into three groups was based on the processor type, ROM size and the display type. Table 1 presents the original data, and the classifications generated by the CLUSTER operator. The input to CLUSTER was a table of the characteristics of the microcomputers, and the output consisted of a table with new columns indicating the groups of the objects, plus rules characterizing the groups.

DIFF

The DIFF operator is based on the AQ inductive learning method that has been effectively used for many rule learning tasks in areas such as medicine, agriculture, physics, computer vision, chess, etc. One recent application for diagnosing potential breast cancers, given a few training examples, is described in [Michalski, Mozetic et al., 1986]. The rules generated performed well on new cases of the disease. An application of the DIFF system to concisely describe the groups created by the CLUSTER operator (Table 1) is shown in Table 2. The groups of examples are given as input, and DIFF creates rules that describe the differences between these groups. Note that the found rules are a little simpler than the descriptions produced by CLUSTER (a redundant condition specifying the processor type in the third group of the 3-grouping cluster was removed). DIFF often produces a significantly simpler description.

While this example showed an application of DIFF to create the *discriminant* rules for groups of examples, the AQ algorithm that it employs may also be used to determine *characteristic* rules that describe classes of events [Michalski, 1983]. In the INLEN system, this function is represented by the CHAR operator. In case of large example sets, there may be large differences between characteristic and discriminant rules.

DISEQ

The DISEQ operator is based on the ABACUS-2 discovery system, described in [Greene, 1988]. The operator is capable of learning equations which fit a set of tabular data. It is also capable of subdividing a set of examples into partitions in which different rules apply, and of coping with noisy data. It specifies conditions under which different rules apply. ABACUS-2 expands the capabilities of the earlier system ABACUS [Falkenhainer and Michalski, 1986], and can discover more complex regularities.

INPUT						OUTPUT	
Microcomputer	Display	RAM	ROM	Processor	No_Keys	2-Group	3-Group
Apple II	Color_TV	48K	10K	6502	52	1	1
Atari 800	Color_TV	48K	10K	6502	57-63	1	1
Comm. VIC 20	Color_TV	32K	11-16K	6502A	64-73	1	2
Exidi Sorceror	B/W_TV	48K	4K	Z80	57-63	1	2
Zenith 118	Built_in	64K	1K	8080A	64-73	2	3
Zenith 1189	Built_in	64K	8K	Z80	64-73	2	3
HP 85	Built_in	32K	80K	HP	92	1	2
Horizon	Terminal	64K	8K	Z80	57-63	1	2
Challenger	B/W_ TV	32K	10K	6502	53-56	1	1
O-S 11 Series	B/W_TV	48K	10K	6502C	53-56	1	2
TRS-80 I	B/W_TV	48K	12K	Z80	53-56	1	1
TRS-80 III	Built_in	48K	14K	Z80	64-73	1	1

CLUSTER operator takes as the input the relational table, marked INPUT, and a parameter requiring it to partition the rows in the table into 2- and then into 3-group clusterings. The two rightmost columns show the partitions generated. The CLUSTER also generates rules describing the groups, stored in the KB:

2-Group clustering:

[Group 1] <== [RAM = 16K..48K] or [No_Keys ≤ 63]
[Group 2] <== [RAM = 64K] & [No_Keys > 63]

3-Group clustering:

[Group 1] <== [Processor = 6502 v 8080A v Z80] & [ROM = 10K..14K]
[Group 2] <== [Processor = 6502A v 6502C v HP] or [ROM = 1K..8K] & [Display ≠ Built_in]
[Group 3] <== [Processor = 6502 v 8080A v Z80] & [ROM = 1K..8K] & [Display = Built_in]

Table 1. An example of the CLUSTER Operator

Microcomputer	Display	RAM	ROM	Processor	No_Keys	2-Group	3-Group
Apple II	Color_TV	48K	10K	6502	52	1	1
Atari 800	Color_TV	48K	10K	6502	57-63	1	1
Comm. VIC 20	Color_TV	32K	11-16K	6502A	64-73	1	2
Exidi Sorceror	B/W_TV	48K	4K	Z80	57-63	1	2
Zenith 118	Built_in	64K	1K	8080A	64-73	2	3
Zenith 1189	Built_in	64K	8K	Z80	64-73	2	3
HP 85	Built_in	32K	80K	HP	92	1	2
Horizon	Terminal	64K	8K	Z80	57-63	1	2
Challenger	B/W_ TV	32K	10K	6502	53-56	1	1
O-S 11 Series	B/W_TV	48K	10K	6502C	53-56	1	2
TRS-80 I	B/W_TV	48K	12K	Z80	53-56	1	1
TRS-80 III	Built_in	48K	14K	Z80	64-73	1	1

DIFF takes as input a relational table in which the last column indicates group (class) membership. In this example, the DIFF operator tries to rediscover the rules, invented by CLUSTER, from the examples of groups:

Rediscovered rules for 2-Group differentiation:

[Group 1] <== [Display ≠ Built_in] or [ROM ≥ 14K]
[Group 2] <== [RAM = 64K] & [No_Keys = 64-73]

Rediscovered rules for 3-Group differentiation:

[Group 1] <== [Processor = Z80 v 6502] & [ROM = 10K..14K]
[Group 2] <== [Processor = 6502C v 6502A v HP] or [ROM = 4K..8K] & [Display = B/W_TV v Term.]
[Group 3] <== [ROM = 1K..8K] & [Display = Built_in]

The above rules were generated by DIFF directly from examples. They are similar, but not identical to the rules created originally by CLUSTER. They provide an alternative, logically consistent, characterization of individual groups.

Table 2. An example of the DIFF Operator

The ABACUS programs have formulated equations characterizing a number of different empirical data, e.g., data specifying planetary motion, the distances between atoms in a molecule, and Stoke's Law of falling bodies. Stoke's Law specifies the velocity of an object falling through different media, and is presented in Table 3. As is shown in Table 3, the velocity of an object falling through a fluid is governed by an equation involving different variables than the equation describing the velocity of an object falling through a vacuum. DISEQ was able to find the equations for both cases.

4. Conclusion

INLEN is a large-scale integrated system capable of performing a wide variety of complex inferential operations on data in order to discover interesting regularities in them. These regularities can be detected in qualitative data, quantitative data, and in the knowledge base itself. In addition, INLEN provides functions that facilitate manipulation of both the data and the knowledge base.

One major novel idea of INLEN is that it integrates a variety of knowledge generation operators that permit a user to search for various kinds of relationships and regularities in the data. To achieve such an integration, the concept of a *knowledge segment* has been introduced. The knowledge segment stands for a variety of knowledge representations such as rules, networks, equations, etc., each possibly associated with a relational table in the database (as in the case of a set of constraints), or for any combination of such basic knowledge segments.

Many of INLEN's modules have already been implemented, as stand-alone systems or as parts of larger units. Other tools and the general integrated interface are under development. Future work will involve bringing these systems together and completing the control system to facilitate access to them in the form of simple, uniform commands.

Acknowledgements

The authors thank Pawel Stefanski, Jianping Zhang and Jan Zytkow for their comments and criticism. They are also grateful to Peter Aiken, Kathleen Byrd and Joyce Ralston for their assistance in the preparation of the paper.

This research was done in the Artificial Intelligence Center of George Mason University. The activities of the Center are supported in part by the Defense Advanced Research Projects Agency under grant, administered by the Office of Naval Research No. N00014-87-K-0874, in part by the Office of Naval Research under grant No. N00014-88-K-0226, and in part by the Office of Naval Research under grant No. N00014-88-K-0397.

Substance	Radius (m)	Mass (kg)	Height (m)	Time (s)	Velocity (m/s)
Vacuum	0.05	1	6	0.1	0.98453
Vacuum	0.05	2	2	0.4	3.93812
Vacuum	0.10	1	3	0.5	2.95359
Vacuum	0.10	2	7	0.1	0.98453
Glycerol	0.05	1	5	0.1	19.112
Glycerol	0.05	2	8	0.3	38.224
Glycerol	0.10	1	6	0.5	9.556
Glycerol	0.10	2	7	0.2	19.112
CastorOil	0.05	1	9	0.4	14.672
CastorOil	0.05	2	3	0.1	29.344
CastorOil	0.10	1	5	0.3	7.336
CastorOil	0.10	2	8	0.5	14.672

DISEQ searches for relationships among the data objects. It discovers that equations for the ball's velocity exist, but they depend on the medium through which the ball is falling.

Here are the rules DISEQ discovered:

If [Substance = Vacuum] **then** $v = 9.8175 * t$

If [Substance = Glycerol] **then** $v * r = 0.9556 * m$

If [Substance = CastorOil] **then** $v * r = 0.7336 * m$

where v = velocity, r = radius, t = time, and m = mass.

Table 3. The DISEQ operator formulates Stoke's Law

References

A. B. Baskin and R. S. Michalski, "An Integrated Approach to the Construction of Knowledge-Based Systems: Experiences with ADVISE and Related Programs," in *Topics in Expert System Design*, G. Guida and C. Tasso (eds.), Elsevier Science Publishers B. V., 1989.

P. W. Baim, "The PROMISE Method for Selecting Most Relevant Attributes for Inductive Learning Systems," Report No. UIUCDCS-F-82-898, Department of Computer Science, University of Illinois, Urbana IL, Sept. 1982.

R. L. Blum, "Automating the Study of Clinical Hypotheses on a Time-Oriented Data Base: The RX Project," Report No. STAN-CS-79-816, Department of Computer Science, Stanford University, Stanford CA, Nov. 1979.

S. A. . Cramm, ESEL/2: "A Program for Selecting the Most Representative Training Events for Inductive Learning", Report No. UIUCDCS-F-83-901, Department of Computer Science, University of Illinois, Urbana IL, Jan. 1983.

C. J. Date, *A Guide to INGRES*, Addison Wesley, Reading MA, 1987.

J. H. Davis, "CONVART: A Program for Constructive Induction on Time Dependent Data," Master's Thesis, Department of Computer Science, University of Illinois, Urbana IL, 1981.

B. Falkenhainer and R. S. Michalski, "Integrating Quantitative and Qualitative Discovery: The ABACUS System," Report No. UIUCDCS-F-86-967, Department of Computer Science, University of Illinois, Urbana IL, May 1986.

G. Greene, "Quantitative Discovery: Using Dependencies to Discover Non-Linear Terms," Master's Thesis, Department of Computer Science, University of Illinois, Urbana IL, 1988.

J. Hong, I. Mozetic and R. S. Michalski, "AQ15: Incremental Learning of Attribute-Based Descriptions from Examples, the Method and User's Guide," Report No. UIUCDCS-F-86-949, Department of Computer Science, University of Illinois, Urbana IL, May 1986.

International Intelligent Systems, Inc., "User's Guide to AURORA 2.0: A Discovery System," Fairfax VA, International Intelligent Systems, Inc., 1988.

B. Katz, T. W. Fermanian and R. S. Michalski, "AgAssistant: An Experimental Expert System Builder for Agricultural Applications," Report No. UIUCDCS-F-87-978, Department of Computer Science, University of Illinois, Urbana IL, Oct. 1987.

L. Kerschberg, (ed.), Expert Database Systems: Proceedings from the First International Workshop, Benjamin/Cummings Publishing Company, Menlo Park, CA, 1986.

L. Kerschberg, (ed.), Expert Database Systems: Proceedings from the First International Conference, Benjamin/Cummings Publishing Company, Menlo Park, CA, 1987.

L. Kerschberg, (ed.), Expert Database Systems: Proceedings from the Second International Conference, George Mason University, Fairfax, VA, 1988. (to appear in book form, Benjamin/Cummings Publishing Company, Menlo Park, CA, 1988.

M. M. Kokar, "Coper: A Methodology for Learning Invariant Functional Descriptions," in *Machine Learning: A Guide to Current Research*, Michalski, Mitchell, Carbonell Eds., Kluwer Academic Publishers, 1986.

P. Langley, G. L. Bradshaw and H. A. Simon, "Rediscovering Chemistry with the BACON System," in *Machine Learning: An Artificial Intelligence Approach*, Michalski, Mitchell, Carbonell Eds., Morgan Kaufmann, 1983.

T. C. Layman, "A PASCAL Program to Convert Extended Entry Decision Tables into Optimal Decision Trees," Department of Computer Science, Internal Report, University of Illinois, Urbana IL, 1979.

R. S. Michalski, "Designing Extended Entry Decision Tables and Optimal Decision Trees Using Decision Diagrams," Report No. UIUCDCS-R-78-898, Department of Computer Science, University of Illinois, Urbana IL, March 1978.

R. S. Michalski, "Theory and Methodology of Inductive Learning," in *Machine Learning: An Artificial Intelligence Approach*, Michalski, Mitchell, Carbonell Eds., Morgan Kaufmann, 1983.

R. S. Michalski and A. B. Baskin, "Integrating Multiple Knowledge Representations and Learning Capabilities in an Expert System: The ADVISE System," Proceedings of the 8th IJCAI, Karlsruhe, West Germany, August 8-12, 1983, pp. 256-258.

R. S. Michalski, A. B. Baskin and K. A. Spackman, "A Logic-based Approach to Conceptual Database Analysis," Sixth Annual Symposium on Computer Applications in Medical Care (SCAMC-6), George Washington University Medical Center, Washington, DC, November 1-2, 1982, pp. 792-796.

R. S. Michalski, A. B. Baskin, C. Uhrik and T. Channic, "The ADVISE.1 Meta-Expert System: The General Design and a Technical Description," Report No. UIUCDCS-F-87-962, Department of Computer Science, University of Illinois, Urbana IL, Jan. 1987.

R. S. Michalski, L. Iwanska, K Chen, H. Ko and P. Haddawy, "Machine Learning and Inference: An Overview of Programs and Examples of their Performance," Artificial Intelligence Laboratory, Department of Computer Science, University of Illinois, Urbana IL, Sept. 1986.

R. S. Michalski and J. B. Larson, "Selection of Most Representative Training Examples and Incremental Generation of VL_1 Hypotheses: the underlying methodology and the description of programs ESEL and AQ11," Report No. 867, Department of Computer Science, University of Illinois, Urbana, May 1978.

R. S. Michalski and J. B. Larson, rev. by K. Chen, "Incremental Generation of VL1 Hypotheses: The Underlying Methodology and the Description of the Program AQ11", Report No. UIUCDCS-F-83-905, Department of Computer Science, University of Illinois, Urbana IL, Jan. 1983.

R. S. Michalski, I. Mozetic, J. Hong and N. Lavrac, "The AQ15 Inductive Learning System: An Overview and Experiments," Report No. UIUCDCS-R-86-1260, Department of Computer Science, University of Illinois, Urbana IL, July 1986.

R. S. Michalski, R. E. Stepp and E. Diday, "A Recent Advance in Data Analysis: Clustering Objects into Classes Characterized by Conjunctive Concepts," in *Progress in Pattern Recognition*, Vol. 1, L. N. Kanall and A. Rosenfeld (Eds.), New York: North-Holland, pp. 33-56, 1981.

R. S. Michalski and R. E. Stepp, "Automated Construction of Classifications: Conceptual Clustering Versus Numerical Taxonomy," IEEE Transactions on Pattern Analysis and Machine Intelligence, 1983.

R. E. Reinke, "Knowledge Acquisition and Refinement Tools for the ADVISE Meta-Expert System," Master's Thesis, Department of Computer Science, University of Illinois, Urbana IL, July 1984.

K. A. Spackman, "QUIN: Integration of Inferential Operators within a Relational Database," ISG 83-13, UIUCDCS-F-83-917, M.S. Thesis, Department of Computer Science, University of Illinois, Urbana, 1983.

R. E. Stepp, "A Description and User's Guide for CLUSTER/2, a Program for Conceptual Clustering," Department of Computer Science, University of Illinois, Urbana IL, Nov. 1983.

R. E. Stepp, "Conjunctive Conceptual Clustering: A Methodology and Experimentation," PhD Thesis, Department of Computer Science, University of Illinois, Urbana IL, 1984.

G. Wiederhold, M.G. Walker, R. L. Blum, and S. Downs, "Acquisition of Knowledge from Data", International Symposium on Methodologies for Intelligent Systems, Knoxville TN, Oct. 1986.

J. M. Zytkow, "Combining Many Searches in the FAHRENHEIT Discovery System," Proceedings of the Fourth International Workshop on Machine Learning, Irvine, CA, Morgan Kaufmann, pp. 281-287, 1987.