Agent Computing, KB for Intelligent Forecasting, and Model Discovery for Knowledge Management

Cyrus F. Nourani
*Affiliations METAAI, USA Academic UCSB
and the University of Auckland, MSIS
Porject_METAAI@CompuServe.com

Abstract

Multiagent business objects and Intelligent Multimedia applications for Management Science and IT are defined. Amongst the areas are agent computing, modern portfolios, heterogeneous computing, business objects, intelligent databases, intelligent objects, and decision science. A new view to MIS with agent computing and intelligent multimedia is presented. Interaction amongst heterogeneous computing resources are via objects, multiagent AI and agent intelligent languages. New applications to business with intelligent object languages are presented in brief. Intelligent multimedia and the new Morph Gentzen logic from (Nourani 1997a) are applied as a preliminary basis for to forecasting. We present a brief overview to new KR techniques with G-diagrams and applications to define computable models and relevant world reasoning. G-diagrams are applied to KR to define localized keying to models and as a minimal efficient computable way for knowledge management.

Keywords Design with Software Agents, Computing Agents and Business Objects, Multiagent AI Techniques, Intelligent Business Objects and Design, Intelligent Forecasting, Stage-Composable Modules, Intelligent Multimedia, Knowledge Management

METAAI South California Correspondence Address P.O. Box 278, Cardiff By The Sea, CA. 92007, USA

1. Introduction

A new view to business computing for certain MIS applications with agent computing and intelligent multimedia is presented. Interactions amongst heterogeneous computing resources are via objects, multiagent AI, and design language abstract monitors. The applications to business based on the basis put forth for modular object languages are briefed. Intelligent Objects form our 1994 papers are reviewed introducing new business applications with agents. Software agents are specific agents designed by a language that carry out specified tasks and define software functionality. Most agents defined by our examples are software agents. With a rapid advancement of technology in recent times and a wide application of object-orientation principles in various diverse sectors of industry, the concept of business object was developed. A *business object is* a representation of something that is active in business domain, with at least the essential information on its business name and definition, attributes, behavior, relationships, and engagement rules. A business object is an application-level entity, developed completely independent of its application areas. A set of attributes describes the state of the entity, and there is a specification for the actions to take concerning the entity itself. (Nourani-Lou 1998) provides the basis for a simple methodology that will enable semi-technical personnel to develop and apply business objects in medium- and small-sized businesses within a relatively short time span, with relative ease and low cost. To this end, available modules of existing software should be reused wherever possible to minimize cost. Intelligent multimedia business objects are presented for the applications. (Nourani-Lou 1998) proposes a business object can have multiple *shadows* in accordance with the different operations of the business and at different times; a shadow is a facet (view) of the business object perceived by a user for a particular purpose.

2. The Practical Basis for Agent Computing

The term "agent" has been recently applied to refer to AI constructs that enable computation on behalf of an AI activity, for example (Genesereth-Nilsson 1987), and (Jennings 1994). It also refers to computations that take place in an autonomous and continuous fashion, while considered a high-level activity, in the sense that its definition is software and hardware implementation, independent (Nourani-1993). *Software agents* are specific agents designed by a language that carry out specified tasks and define a software functionality. Most agents defined by our examples are software agents. Objects are in the well-known sense of the word in object programming, abbreviated by OOP, see, for example, (Ten Dyke and Kunze 1989). Objects consist of abstract data, perhaps encapsulation, and operations. Most recent programming techniques apply OOP in some form. Software engineering techniques with abstract data types have had OOP on their mind. IOOP (Nourani 1995b) is a recent technique developed by the author combining AI and software agents with OOP. KB is an abbreviation for knowledge bases. The separation makes large applications more tractable, global analysis feasible, and software reusable. Modular languages deal with languages, techniques, and tools for the design and implementation of large-scale software systems in a modular and extensible way. Languages, techniques, and tools for the design and implementation of large-scale software systems in a modular and extensible way. JAVA, Modula, and SAP,

are few examples. For our project the modular programming concepts are combined with software agent computing, new Intelligent OOP-IOOP object-coobject pairs. CORBA views designs with objects to be either business objects, A business process object, or a presentation object. For these applications, an *object* is defined to be a uniquely identifiable real-world entity.

3. Intelligent MIS and Business

The new MIS as an academic and business field might be depicted by the enclosed figure.

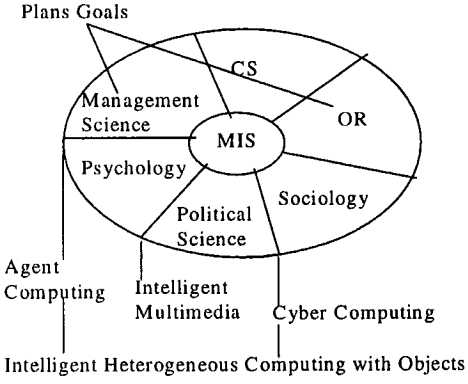


Figure 1- New MIS Essentials

Software agents are specific agents designed by a language that carry out specified tasks and define software functionality. Most agents defined by our examples are software agents. Academic MIS essentials (Lauden 1997) might be redefined as the figure indicates. There is agent computing, cyberspace computing, intelligent multimedia and heterogeneous computing. Plans and goals see for example (Genesereth-Nilsson 1987) are applied to business planning(Nourani 1998a,b) and OR.

3.1 Heterogeneous Visual Computing

Modular languages deal with languages, techniques, and tools for the design and implementation of large-scale software systems in a modular and extensible way. CORBA, JAVA, Oberon, and SAP, are few. These languages do not have specific agent computing support. For our project the modular programming concepts are combined with software agent computing, new IOOP constructs, object-coobject pairs and kernels. Modules are aggregate objects with a specific functionality defined. Parallel Module Specification What we intend by a parallel specification language is one that allows a user to define parallel modules with embedded communicating processes. The user supplies the type of parallelism and the types of messages processes are expected to communicate. The processes are embedded within a module, and there is syntax provided for communicating pairs of processes are.

3.2 Parallel Visual Objects Computing

Monitors can be defined by augmenting modules with several gate processes running in parallel with a monitor module. The message syntactic categories can be defined in very simple terms to suit the particular intended application areas. In addition, the specification language is extendible as far as the type of communication it allows the processes and the nature of parallelism. The process bodies can be specified in terms of the intended actions and the messages they send.

3.3 IOOP

(Nourani 1995b) had presented new OOP techniques called Multiagent Object Level Programming and new paradigms for OOP defining intelligent objects. New linguistics constructs are defined for object level programming with String and Splurge functions treating object visibility and messages. Treating objects as abstract data types and a two level programming approach to OOP allows us to define Pull-up abstractions to treat incompatible objects. The techniques open new areas of research on object level programming to pursue, and put us at MJOOP, the Multiagent Junction For OOP.

We present new techniques and languages for object level programming with intelligent trees implemented by agent functions. We have defined intelligent syntax and put forth the basis for automatic tree implementation techniques for object level programming. We define *OOPI*, OOP Intelligent, functions and Pull-up Abstractions in brief from our IOOP(Nourani 1995b) as a structural analogy to operation overloading. New linguistics constructs are defined for object level programming treating objects as abstract data types. The OOP, for example (Pree-1996), defined by the present programming techniques have tree rewrite automatic implementations. The techniques, concepts, and paradigms defined by the Project are as follows: Intelligent Objects, MJOOP - The Multiagent Junction to OOP; Intelligent OOP Languages. The Object Coobject Design Paradigm; The Pull-up Abstraction Function; and Designs With Intelligent Objects.

4. Intelligent Objects and Multiagent OOP

The IOOP project develops new techniques, and linguistics constructs for programming with objects implemented by agents, based on a theory of computing with trees on signatures carrying agent functions on trees (Nourani 1996b). The agents are designated functions with specified functionality and message syntax. Thus context can be carried at syntax. We present new techniques and languages for object level programming with intelligent trees implemented by agent functions. We

show in IOOP and brief in the present paper, how a two-level language paradigm and intelligent object level programming can handle what otherwise is a complicated computing phenomena. There are objects as situated automata, for which abstract syntax trees and a computing theory merging with the current practice of programming theories is quite impossible.

4.1. Programming with Intelligent Syntax

The project has a basis for programming techniques that are a foundation for the computing that is inevitable with the current and forthcoming AI and intelligent object computing (Nourani-1993b, 1995,1996b). There is a gap between the OOP object programming trends and objects as applied to AI programming (Nourani 1993b, 1996b). We had started on a project to bridge the abstract data type AI KR in our 1985 paper (Nourani-Lieberherr 1985). For OOP the AI application areas are outlined in Nourani 1993-1997 papers. The present paper offers syntactic OOP techniques applicable to AI and ordinary OOP. An important technical point is that the agents are represented by function names that appear on the free syntax trees of implementing trees. The trees defined by the present approach have function names corresponding to computing agents. The computing agent functions have a specified module defining their functionality.

4.2 Intelligent Syntax Languages

IOOP applies intelligent languages. By an intelligent language we intend a language with syntactic constructs that allow function symbols and corresponding objects, such that the function symbols are implemented by computing agents. Agents appear as functions, expressed by an abstract language that is capable of specifying modules, agents, and their communications. The most recent language support for AI programming is with languages where objects are at play without specific directions and computing access to how the syntax trees might be carried. Objects appear on various semantic networks and resemble situated automata, which implement a computation by asynchronous methods. The goal in Nourani 1993-97 had been to have language constructs that allow us to handle objects on abstract syntax trees and implement the mystical behavior of situated automata by agents. Formal techniques with intelligent syntax, String and Splurge programming linguistics allow us to treat visibility in a precise way and to get theoretical results for MJOOP and OOP. These techniques the formalization for AI computations Nourani1993-97 led to an exciting new programming theory and practice for MJOOP. The new programming techniques present the basis for programming with nondeterministic syntax.

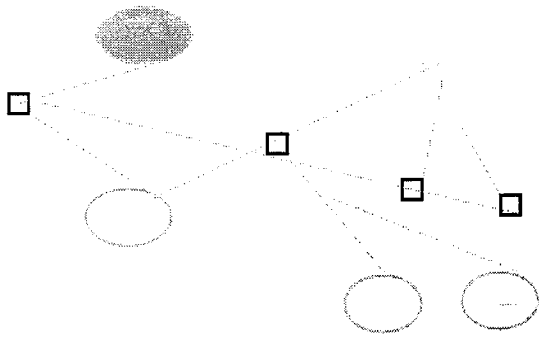


Figure 1 Module Implemented By Agents
Circles are objects, the squares are agents, and object-coobject pairs are enclosed by a rectangle. The dotted lines are agent message passing paths.

5. Intelligent Business Objects

5.1 The Optimistic Design Paradigm

Our design technique is to think of solving a problem without anything going wrong, then account for all faulty behavior, recoverable activities, and exceptional behavior. It is done by defining preconditions on coobjects, and agents defined on the coobject that can make intelligent recovery from unusual behavior. The exception knowledge has to be acquired in every case. The dual design technique is conceptually simpler when it comes to systems with extraordinary complexity. There are thousands of processes occurring conceptually concurrent in the systems designed. It allows the designer to focus on problem solving without being distracted with all that can go wrong. It is also theoretically elegant since it views exceptional behavior a symmetrically significant design parameter. The design sections are relegated by objects to modules and activities are modeled by multiagent processes.

5.2 Business Objects

The software engineering and database adaptation of the object-orientation view of encapsulation first emerged in the late seventies and ever since it has undergone a gradual evolution and development in applications. In this connection, an *object* is defined to be a uniquely identifiable real-world entity with a set of attributes that describes the state of the entity, and to have a specification of the actions to take concerning the entity itself. Objects are visible to users as icons, boxes and windows on a screen that the users can manipulate.

A business object is an application-level entity, developed completely independent of its application areas. It is generally conceived as a self-contained deliverable that

has a user interface and state and knows how to cooperate with other separately developed business objects in executing a desired task. Business objects provide a natural way of describing application-independent concepts such as customer, car, employee, invoice, order, payment, processes or events that are meaningful to business operations. As (Nourani-Lou 1998) briefs the usage of business objects differs from traditional application in the context of reusability. Generally the business objects defined for use in one application can be reused, with a little orientation, in another. Business objects such as car, customer and invoice objects, for instance, could be structured to support multiple views for different business applications. To illustrate, the business object 'car' defined in one application may be specialized through inheritance to take account of the particularities of a car sales business for use in another application. Thus a business object need not be a monolithic entity. It can be factored internally into a set of cooperating objects that can react to different business situations; the CORBA business object provides one such example. In recent times business objects have been used at software level to integrate and migrate between legacy applications and databases, especially in the area of distributed computing.

5.3 IM and Portfolios

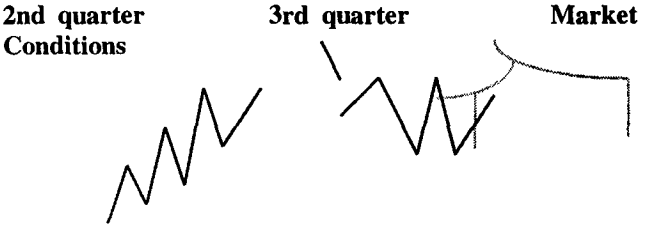
Set of operations on portfolio lists, IM-Operations on graphs, Operations of reports and simulation, set of procedure constructing operations IM - A New Computing Area defined by (Nourani 1996a, 97a). A new computing area is defined by artificial intelligence principles for multimedia. The area for which the paper provides a foundation for is what multimedia computing is bound to be applied at dimensions and computing phenomena unimagined thus far, yet inevitable with the emerging technologies. The principles defined are practical artificial intelligence and its applications to multimedia. Multimedia AI systems are proposed with new computing techniques defined. Multimedia objects and rules and multimedia programming techniques are presented via a new language called IM. Some applications outlined by (Nourani 1998) are: IM and automatic windows with rules as a basis for connectivity management Intelligent syntax computing and IM have been applied as a basis. Multiagent computing with IM a new basis for new MSIS areas and communication management. IM and automatic windows with rules have been applied as a basis for decisions and connectivity management by (Nourani 1998).

5.4 The Morph Gentzen Logic

The IM Morphed Computing Logic Logics for multimedia computing is a new projects with important applications since (Nourani 1996a,97a,99a). The basic principles are a mathematical logic where a Gentzen or natural deduction systems is defined by taking arbitrary structures and multimedia objects coded by diagram functions. Gentzen sequents are deduction rules of the forms obtained from an ordered pair {S ;L} where S and L are finite sequences of zero or more formulas each. The formal expression $S \rightarrow L$ is called a *sequent*. S is called its antecedent and L its succedent(Kleene 67). Morph Gentzen(Nourani 1997a) admits new sequents with the Morph Gentzen rules. It has new logical principles and foundations. By trans-morphing hybrid picture's corresponding functions a new hybrid picture is deduced. The techniques can be applied to arbitrary topological structures. Multimedia objects are viewed as syntactic objects defined by functions, to which the deductive system is applied. Thus we define a syntactic morphing to be a technique by which multimedia objects and hybrid pictures are homomorphically mapped via their defining functions to a new hybrid picture. The deduction rules are a Gentzen system augmented by Morphing, and Transmorphing. The logical language has function names for hybrid pictures. The MIM Morph Rule - An object defined by the functional n-tuple <f1,...,fn> can be Morphed to an object defined by the functional n-tuple <h:f1,...,h:fn>, wehre h: fi selects the morphed structure obtained fron what fi selected, provided h is a homomrphism of abstract signature structures(Nourani 1996b). The MIM Trans-Morph Rules- A set of rules whereby combining hybrid pictures p1,...,p.n. defines an Event {p1,p2,...,p.n.} with a consequent hybrid picture p. Thus the combination is an impetus event. The language and MIM rules are applied to algebraic structures. The deductive theory is a Gentzen system in which hybrid pictures are named by parameterized functions; augmented by the MIM morph and transmorph rules. The Model theory is defined from Intelligent syntax languages(Nourani 1996b,97b). A computational logic for intelligent languages is presented in brief with a soundness and completeness theorem in (Nourani 1996b). Morph Gentzen logic is proved to be sound and complete (Nourani 1996c).

5.5 Stock Forecasting

IM's basis for forecasting are put forth at preliminary stages(Nourani 1999b. The idea is to apply Morph-Gentzen logic(Nourani 1997a) as a basis for intelligent multimedia forecasting. The figure indicates a graphics sequent for predicting the fourth quarter earnings from the second and third combined with a market condition graph. The way a market condition graph is designed is a propriety issue. It is obtained by Morph Gentzen sequents from known stock market parameters.



4thquarter profits

6. KR and KB Model Discovery

6.1 The Knowledge World Models

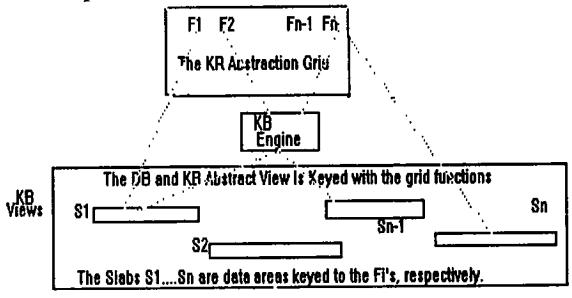
Knowledge representation has two significant roles: to define a model for the AI world, and to provide a basis for reasoning techniques to get at implicit knowledge. An ordinary diagram is the set of atomic and negated atomic sentences that are true in a model. Generalized diagrams are diagrams definable by a minimal set of functions such that everything else in the models closure can be inferred, by a minimal set of terms defining the model. Thus providing a minimal characterization of models, and a minimal set of atomic sentences on which all other atomic sentences depend. The diagram of a structure in the standard model-theoretic(Nourani 1995a) sense is the set of atomic and negated atomic sentences that are true in that structure. The generalized diagram (G-diagram) is a diagram in which the elements of the structure are all represented by a minimal family of function symbols and constants, such that it is sufficient to define the truth of formulae only for the terms generated by the minimal family of functions and constant symbols. Such assignment implicitly defines the diagram. It allows us to define a canonical model of a theory in terms of a minimal family function symbols.

6.2 Diagrams For World Computing

Existential quantified diagrams carry one main deficit. The Skolem objects are not characterized, Hilbert's epsilon symbol may resolve this. This not only allows us to make the connection between PD and abduction even closer , because we can abduce in a FOL (first order logic) framework the description of necessarily existing objects for making a proof succeed. Existentially quantified diagrams carry a main deficit. The Skolemized formulas are not characterized. Hilbert's epsilon symbol may be applied to solve this problem. Here we present the notion of an predictive diagram(Nourani-Hoppe 1995) and apply it for KR to provide a model-theoretic characterization for PD and related proof trees. An **predictive diagram** for a theory T is a diagram D[M], where M is a model for T, and for any formula q in M, either the function f: $q \rightarrow \{0,1\}$ is defined, or there exists a formula p in D[M], such that T U {p} proves q; or that T proves q by minimal abduction. A **generalized predictive diagram**, is an predictive diagram with D[M] defined from a minimal set of functions. The predictive diagram could be minimally represented by a set of functions {f1,...,fn} that inductively define the model. The free trees we had defined by the notion of provability implied by the definition , could consist of some extra Skolem functions {g1,...,gl},that appear at free trees. The f terms and g terms, tree congruences, and predictive diagrams then characterize partial deduction with free trees. As was argued recently Hoppe partial deduction can be regarded as a form of abductive reasoning, and vice versa, if differences in the underlying languages of PD and Abduction are resolved. However, the most interesting result of this work, was that abductive reasoning can be extended to perform partial deduction, if universal quantified hypotheses are used, and if thus a special consistency checking scheme, based on Hilbert's epsilon symbol is used.

6.3 KR with Keyed Functions

We had shown in our papers(Nourani-Lieberherr 86) KR with abstract data types allows us to define automatic KR. Abstract types keep the attention of system designers on KR issues rather than programming. Even most trivial situations require masses of knowledge and data types allow us to structure knowledge. Existentially quantified diagrams for KR carry a main deficit- the Skolemized formulas are not characterized. Hilbert's epsilon symbol may be applied to solve this problem. We have presented the notion of an predictive diagram and applied it to provide a model-theoretic characterization for PD and related proof trees.



Let us see what predictive diagrams do for knowledge discovery knowledge management. Diagrams allow us to model-theoretically characterize incomplete KR. To key into the incomplete knowledge-base we apply generalized predictive diagrams whereby specified diagram functions a search engine can select onto localized data fields. A *Generalized Predictive Diagram*, is an predictive diagram with D[M] defined from a minimal set of functions. The predictive diagram could be minimally represented by the set of functions {f1,...,fn} that inductively define the model. The free trees defined by the notion of provability implied by the definition , could consist of some extra Skolem functions {g1,...,gl}, that appear at free trees. The f terms and g terms, tree congruences, and predictive diagrams then characterize partial deduction with free trees.

6.4 Discovery Computation as Plans

Data discovery from KR on diagrams might be viewed as satisfying a goal by getting at relevant data which instaniates a goal. The goal formula states what relevant data is sought. We propose methods that can be applied to planning(Nourani 1991) with diagrams such that similar effects to partial deductions can be achieved and perhaps some of the techniques can be applied to implement discovery planning. In planning with GF-diagrams that part of the plan that involves free Skolemized trees is carried along with the proof tree for a plan goal. The idea is that if the free proof tree is constructed then the plan has a model in which the goals are satisfied. The model is the initial model of the AI world for which the free Skolemized trees were constructed. Partial deductions in this approach correspond to proof trees that have free Skolemized trees in their representation. While doing proofs with free Skolemized trees we are facing proofs of the form p(g(....)) proves p(f(g(....)) and generalizations to p(f(x)) proves For all x , p(f(x)). Thus the free proofs are in some sense an abstract counterpart of the SLD.

7. Conclusions

A new basis for MIS for the new computing paradigms is essential. We have defined new design techniques and applications introducing Intelligent Business Objects. The development of intelligent business objects with IM intelligent agents to reiterate new generations of business objects in the multimedia environment is an important progress(Nourani-Lou 1998) with specific application areas to the bank of New Zealand. Techniques and specific application areas are put forth. The project further allows us to compose modules, multiagent designs and abstract implement. It allows software to be designed applying KB to specify the multiagent design. Practical applications with IM and specific business object design techniques are put forth for Intelligent Business Objects. New direction for forecasting and business planning is put forth applying Morph Gentzen. We have further showed how computable AI world models for KB and knowledge management are designed. KR with G-diagrams for models are applied for keying to KB. Discovery at KB's are with specific plan computing techniques.

References

Nourani,C.F. 1993,"Abstract Implementation Techniques For AI Systems By Computing Agents: A Conceptual Overview," Proceedings SERF-93, Orlando, Florida. . Published by the University of West Florida Software Engineering Research Forum, Melbourne, FL.

Nourani, C.F. and K.J. Lieberherr, "Data Types, Direct Implementations, and Knowledge Representation," *Proceedings. 19th_HICSS*, Honolulu, Hawaii, January 1986, Vol. II, 233-243.

Jennings, N.R., 1997 , Cooperation in Industrial Multi-Agent Systems, World Scientific Publishing Company.

Ten Dyke, R.P. and J.C. Kunz 1989, "Object Oriented Programming, IBM Systems Journal, vol. 28, no. 3, 1989.

Nourani,C.F. 1995, The IOOP Project, SIGPLAN Notices 30:2, 56-54, February 1995.

Pree, W., 1995"Design Patterns for Object-Oriented Software Development", Addison-Wesley/ACM-Press.

Nourani,C.F. 1993b ,"Double Vision Computing," IAS-4, Proceedings Intelligent Autonomous Systems, Karlsruhe, Germany, 1995.

Gensereth, M, and N.J. Nilsson ,1987, Logical Foundations of Artificial Intelligence, Morgan-Kaufman.

Nourani,C.F.1997a,"MIM Logic," Summer Logic Colloquium, Prague, BSL, August 1998.

Nourani,C.F. 1997b, Intelligent Languages- A Preliminary Syntactic Theory, May 15, 1995,Mathematical Foundations of Computer Science;1998, 23rd International Symposium, MFCS'98, Brno, Czech Republic, August 1998, Jozef Gruska, and Jiri Zlatuska;(Eds.): Lecture Notes in Computer Science;1450, Springer, 1998.

Shelton, R. E. 1995. "Business Objects:OOBE Framework & Reference Model" from *Data Management Review* , March.

Nourani,C.F. and G.S.L. Lou 1988,"Intelligent Business Objects And Agent Computing," METAAI and The Univerity of Auckland MSIS, April 1998. International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA'99) to be held in Las Vegas (USA), June 28 - July 1, 1999.

Gentzen, G. 1943, Beweisbarkeit und Unbewiesbarket von Anfangsfallen der trasnfininten Induktion in der reinen Zahlentheorie, *Math Ann 119, 140-161,1943.*

Lauden 1997 Lauden, K and J Lauden, MIS Essentials, 1997.

Prawitz 65 Prawitz, D,"Natural Deduction: A proof theoretic study. Stokholm, Almqvist and Wiksell.

Kleene, S.C. ,1967, Mathematical Logic, J.Wiley and sons.

Nourani, C.F.1998a, " Agent Computing and Intelligent Multimedia The Challenges and Opportunities for Business,Commerce, Management Science and Information Systems, February 1998, METAAI and the University of Auckland, MSIS.

Nourani,C.F., 1996a," Intelligent Multimedia," 1996, A Poster Announcement, ICCIMA February 1998, Monash University, Victoria, Australia.

Nourani, C.F. 1996b,"Slalom Tree Computing," AI Communications, The European AI Journal, December 1996, IOS Press, Amsterdam.

Nourani, C.F. 1999a, Intelligent Multimedia- New Computing Techniques and Its Applications. Proceedings CSIT'99, 1st International Workshop on Computer Science and Information Technologies, January 18-22,

1999, Moscow, Russia. Ch. Freytag and V. Wolfengagen (Eds.): MEPhI Publishing 1999, ISBN 5-7262-0263-5
Nourani,C.F. and Th. Hoppe 1995, G-Diagrams For Models, Free Proof Trees, And Abductive Automated Reasoning, February 21, 1995, *Automated Reasoning-AISB*, England, April 1995
Nourani 91 Nourani, C.F. "Planning and Plausible Reasoning in AI," Proc. Scandinavian Conference in AI, May 1991, Roskilde, Denmark.
Nourani 95a Nourani,C.F.,"Free Proof Trees and Model-theoretic Planning," February 23, 1995,Automated Reasoning AISB, England, April 1995.
Nourani,C.F. 1999b,"Agent Computing, Management Science, and Intelligent Forecasting," February 1998.
Proceedings ICAI99, Las Vegas, Nevada, July 1999. r