Morph II: A Universal Agent: Progress Report and Proposal

Robert Levinson

UCSC-CRL-94-22
June 10, 1994

Board of Studies in Computer and Information Sciences
University of California, Santa Cruz
Santa Cruz, CA 95064
levinson@cse.ucsc.edu

ABSTRACT

This short report describes progress on the Morph Adaptive Pattern-Oriented chess system over the last four years and describes ongoing work on Morph II, a domain-independent machine learning system and problem-solver. Morph II, is a direct generalization of the original Morph model and incorporates significant improvements to all aspects of the original system.

The Universal Agent (Morph II) falls directly within the hierarchical reinforcement learning paradigm that has been gaining popularity in the last few years, but goes beyond: in this learning model, the system is responsible for abstracting its own features and patterns, developing its own learning modules, and employing transformations on the representations used. In essence the system is learning neurally (through weight propagation), genetically (through pattern evolution) and symbolically (through its own abstractions). Still, the system maintains uniformity, in that each module at any level of the hierarchy is an instantiation of the same learning strategy.

This report describes the motivations and philosophy behind MorphII, outlines current and future performance objectives, and discusses the potential impact of this new technology.

Keywords: artificial intelligence, mathematics, machine learning, neural networks, genetic algorithms, reinforcement learning, computer chess, problem-solving, tile puzzles, pattern discovery, representation change, heuristic search, games, function learning, RETE.

1 PROJECT SUMMARY

Artificial Intelligence research is in need of a unifying paradigm for general problem solving that is practical, theoretically sound, and intellectually sensible. In particular, this paradigm should unify the strengths and methods of currently factionalized schools in AI. To be of most use to the field, the paradigm should be implemented in publicly available software and placed in rigorous competition, theoretically and experimentally, with existing methods on a wide class of search and planning problems.

This project involves the exploration of such a paradigm through the development of a universal problem-solving module known as a Universal Agent. The Universal Agent is built up from first principles for effective information storage, retrieval, machine learning and heuristic search. In pursuing this unifying framework, a generalization of single-agent and multi-agent state-space search domains as special cases of a "Game of Abstract Mathematical Relations" has been developed.

The Universal Agent differs markedly from other universal problem solvers such as Soar, in that it does not divide domains into "problems" and "methods" and require a "toolkit" of human supplied heuristics and algorithms. The Universal Agent is a more fundamental approach in which each domain is viewed as a variation of the Game of Abstract Mathematical Relations and relies on mathematical and statistical techniques for discovering and exploiting the inherent structure of these domains. As far as possible, human intervention has been limited to a simple mathematical description of a given domain.

The Universal Agent falls directly within the hierarchical reinforcement learning paradigm that has been gaining popularity in the last few years, but goes beyond: in this learning model, the system is responsible for abstracting its own features and patterns, developing its own learning modules, and employing transformations on the representations used. In essence the system is learning neurally (through weight propagation), genetically (through pattern evolution) and symbolically (through its own abstractions). Still, the system maintains uniformity, in that each module at any level of the hierarchy is an instantiation of the same learning strategy.

The Universal Agent is a generalization of the Adaptive-Predictive Search model employed by the Morph Chess system in the previous project period. In this period the primary objective is to demonstrate the power of the Universal Agent (also called MorphII) by reaching Chess Master strength given just the rules of chess and by achieving reasonable strength, after training, across a whole class of state-space search problems.

The previous Morph system, starting with little domain knowledge, is able to defeat human novices while searching just 1-ply. Because of a large number of optimization breakthroughs achieved in the previous funding period, the pattern-matching and game monitoring facilities now allow MorphII to search nearly as deep as existing software-based chess programs. Now that the system is up to comparable speed (without losing generality) the fundamental hypothesis that pattern-matching, representational change and experience can effectively replace and enhance brute-force and selective search can be verified and demonstrated.

The increased speed is one significant improvement in MorphII, equally significant are the following improvements to the learning system:

- The system is domain-independent.
- All domains are viewed in terms of mathematical relations and transformations over mathematical structures.

- A pattern or feature language is no longer supplied but is instead derived from the rules and evolved using a small set of domain-independent mathematical representations and transformations known by the system.
- No limit to the complexity of patterns and analogies formed by the system. Patterns that have been discovered may be used as abstractions in higher order patterns.
- The weight combination function for state evaluation is no longer supplied but is instead derived by the system.
- Learning system is parameterless. No constants or learning rate parameters are supplied. All such constants used by the system are dynamically adjusted based on experience.
- Because of the speed enhancement, traditional search strategies may be explored as well as cognitively inspired and decision-theoretic models that exploit patterns, probabilities and learning.

The project is designed to explore the Universal Agent along three main lines of inquiry: Inquiry 1 studies the extension of Morph to be fully domain-independent based on a view of chess as an instance of the "Game of Abstract Mathematical Relations" and graph patterns as special cases of relations. Inquiry 2 involves the development of a "rule analysis" mechanism that attempts to exploit the graph theoretic structure of a given domain to develop an heuristic problem-solving strategy for that domain. Inquiry 3 involves viewing the MorphII learning mechanism as an algorithm for general function learning.

2 Results of Prior Project Stages

The first project explored Adaptive-Predictive Search (APS). This is a method by which search systems can improve through experience. In APS, knowledge is stored as pattern-weight pairs (pws), where patterns, represented by conceptual graphs, are boolean predicates over states and weights are estimates of the expected distance of states satisfying the pattern from a goal state. Starting from a virtually empty database, pws are learned from search experiences using a combination of learning techniques: temporal-difference learning, weight updating, and pattern creation/deletion. Patterns are stored in a partially-ordered hierarchy by more-general-than to facilitate efficient associative recall. Each state's evaluation is formed as a function of the weights of the most specific stored patterns that apply to that state. Ideally, an APS system should converge to a database such that simply by moving from the current state to the most promising unexplored adjacent state good solutions can be obtained (without further lookahead). In practice, APS systems couple with a "guided" search based on previous experience. In addition to weights, other statistics such as "number of uses" or variance may be stored with patterns to be used in determining their importance and whether they should be maintained by the system. APS systems are similar to genetic classifier systems, except that structural patterns are used, no fitness function is available (beyond the outcome of a given search), and the pattern representation and creation mechanisms exploit domain-dependent symbolic knowledge.

Morph, currently under study, is an adaptive pattern-oriented chess system based on the APS model. It has been identified as an integrated learning system that builds on recent machine learning developments in neural networks, genetic algorithms, and pattern-based learning. The integration of these diverse methods is one of this project's important research contributions. We have constrained Morph in various ways (e.g., a limit to 1-ply search) to build a system that is more consistent with cognitive models of human chess

performance. We have obtained a set of games from a young chessmaster that is helping us to further understand human chess performance and development. Morph's syntactic form of computation is substantially different from other AI "knowledge- oriented" approaches that have been applied to chess and other domains. Differences include:

- use of a uniform representation of search knowledge,
- rejection of learning-by-example for a cognitively-inspired experiential framework,
- responsibility for feature discovery given to the system, and
- low reliance on search.

Additionally, APS can be viewed as the learning of a "semantic distance function" where the semantics are due to the reinforcement the system receives and are expressed in terms of pattern-weight invariance classes. Given our consistent progress on all fronts we are confident that Morph will meet or exceed our objective of its development into an above-average tournament player by the end of the 3-year funding period.

The APS model is an integrated framework and project model used in teaching concepts of search, learning, and representation in our undergraduate and graduate level Artificial Intelligence and Knowledge Engineering courses at UC-Santa Cruz. Students are asked to build APS systems for a state-space search problem of their choice; numerous game domains have been implemented. Nearly all students find this to be a worthwhile experience, especially with regard to learning the trade-offs between representational complexity and search. The paper "Adaptive-Predictive Game-Playing Programs" is a tangible result of these efforts [27]. Recently, students have been required to write programs for "uninformed MetaGame," a universe in which the system has no a priori knowledge of the rules or objective of the game.

Three exciting and unforeseen developments took place in the first year or so of the second project. First was the development of strongly optimized methods for the associative retrieval of semantic networks (SNs). These methods (significant extensions to earlier work [23]) were published in the paper "Pattern Associativity and the Retrieval of Semantic Networks" [24], and have been enhanced and verified since. These methods allow APS-like associative pattern retrieval to require graph matching on only a small fraction of the database. This is achieved through exploitation of redundancy and bit encoding schemes for partially-ordered data and it not dependent on a graph-oriented view of the data. Second is the founding of a worldwide collaborative effort of more than 80 researchers to develop "PEIRCE: A Conceptual Graphs Workbench" founded on John Sowa's Conceptual Graph Theory [7, 9]. The associative pattern retrieval schemes and APS learning mechanisms are to be at the core of this architecture. Subgroups exist for natural language processing, systems modeling, vision, medicine, computer graphics, hardware, database, and learning. Third is the joining of this AI work with the MISC hardware development in our Computer Engineering department. MISC can support in hardware the semantic network operations required by PEIRCE and the pattern-matching operations required by the associative database. Both APS and PEIRCE are expected to be widely applicable across AI applications.

In the second year, the project went through a conceptual reorganization, resulting in an implemented software system known as MORPH II. Encouraged by Morph I's defeating of human chess novices, we felt our primary hypothesis that **experience plus sensitivity to graph-isomorphism can replace search given the proper sensors and abstractions had been verified to a significant degree**. To prepare the way for what we hope will be a significant advance (compared to many previous AI efforts) we developed a view of chess

and search problems as games of transformations of abstract mathematical relations. Now with firm mathematical foundations (in group theory, abstract algebra and graph theory) we are directly confronting the structure of the "problems themselves" in the development of a domain-independent reinforcement learning mechanism that exploits knowledge of a graph-theoretic definition of a domain and resulting abstractions in the learning process. We are also working on complementary projects that attempt to construct heuristics from the mathematics of the operator definitions and their interactions and also "blind learning" systems that learn without knowledge of the rules (as in Metagame) - but which attempt to exploit as far as possible an "assumption of regularity". The MorphI software has been publicly available at our ftp site throughout the life of our project. The MorphII software will be made publicly available as part of the PEIRCE conceptual graphs workbench and can support general research in heuristic search, neural networks and machine learning in addition to incorporating our methods.

In the third year we have enhanced MorphII by designing an algorithm that compiles the rules of any search problem into a RETE-like network that allows the problem to be monitored and patterns to be matched incrementally (as opposed to starting fresh with each position). This is followed by a conversion of relational tables to bit matrices that allow the system to exploit the logical instruction set of the machine and also provide the potential of viewing state-space search in terms of linear algebra and similar mathematical models. With processing times 10-100 times faster than before, we feel that we are now in an ideal position to demonstrate the computational viability of our pattern-matching approach to the goals of AI and robotics.

Many publications have been completed since the beginning of the project on the topics mentioned above: hierarchical pattern retrieval [8, 31, 24, 28], PEIRCE [7, 9, 37, 46], pws [35, 26], APS [27, 14, 32, 15], games and mathematical abstraction[10, 34, 25, 31, 56], and experience and creativity [29, 33, 60].

3 Summary of Present and Future Plans

3.1 Motivations

Thirty years ago, GPS was proposed as a model for "General Problem Solving". The promise of the syntactic approach espoused soon met with the hard realities of combinatorially explosive search spaces. Since that time it has been believed that the main technique used, "means-ends analysis" (and similar problem-solving algorithms) must be coupled with large amounts of domain-specific knowledge and heuristics. We differ with this assessment and the influence it has had on AI: First, the supplying of domain-specific knowledge by humans, while certainly practical, does not directly confront a number of critical scientific issues in achieving intelligence in computation. Second, two other methods for acquiring domain knowledge are possible: a. **Experiential/reinforcement learning.** Such an approach has been popular in recent years [59]. b. **Heuristic construction through analysis of the structure of the problem domain itself.** This method has been largely ignored. In fact, many of the efforts in AI can be viewed as consciously or unconsciously avoiding consideration of the mathematics of the problem domain.

Webster defines mathematics as follows:

> **math.e.mat.ics ṁath-*-'mat-iks n pl but sing in constr 1: the science of numbers and their operations, interrelations, combinations,**

generalizations, and abstractions and of space configurations and their structure, measurement, transformations, and generalizations

Is this not the ideal tool to apply to the study of problem spaces? Further, what we call machine learning should move to a model where mathematics can operate directly on mathematics. That is, we propose that the secret to unlocking the power of computation is to get mathematics to develop and operate on itself. Lenat's AM "automated mathematician" [21] can be viewed as a serious and exciting attempt to get the machine to behave as a mathematician by supplying heuristics, constructs and discovery algorithms similar to those employed by mathematicians. It was, however, admitted that much of AM and Eurisko's [20] success was dependent on the applicability of Lisp-like representation to the domain under study [22]. A similar complaint could be made of the original Morph system: its human-supplied graph representation is ideally suited for chess. With MorphII we are going deeper than this. We are attempting to get at the root of the development of mathematical structures s so that the machine, by producing and discovering mathematical constructions through a universally applicable algorithm, can evolve the representations essential to succeed in a given domain.

3.2 Work Plan

Specifically, in this project we are extending our previous work on adaptive pattern-oriented search in three main directions, each designed to give the machine more of the power of mathematics: Inquiry 1 involves extending the Morph chess system to be fully domain-independent by viewing chess as an instance of a "Game of Abstract Mathematical Relations" and graph patterns as special cases of relations. Inquiry 2 involves the development of a "rule analysis" mechanism that attempts to exploit the graph theoretic structure of a given domain to develop an heuristic problem-solving strategy for that domain. Inquiry 3 involves viewing the APS mechanism as an algorithm for general function learning based on the progressive discovery of relations and analogies. This progressive development is based on an "Assumption of Regular Mathematical Structure" akin to Occam's Razor or the minimum description length principle but allowing for higher level constructs in the representation. By taking this view we are trying to address the representational transformation problems that plague most machine learning algorithms. Each inquiry will be pursued through a balance of theoretical analysis and empirical studies.

3.3 Performance Objectives for MorphII

We have met or will soon meet our early performance objectives stated in the These objectives included the following and similar variants:

- That the system learns to play good fundamental chess (Class B or C) through training, starting from only a small amount of supplied chess knowledge.
- That the system is able, by playing an opponent that is better than it to develop to a point in which it can defeat that opponent.

Now we are in position to set our goals much higher and aim for a demonstration that will make obvious the practical consequences of employing this new technology. Thus, we set the following performance objective which we feel is well within reach: *Objective 1: MorphII when applied to the domain of chess will achieve Master strength (2200 UCSF or higher).*

MorphII is fully *domain-independent* and except for making use of a natural encoding of the rules of a given domain is given no assistance by humans. It is also a general learning mechanism that can be applied to general function learning and classification problems in addition to state-space search. MorphII is designed to compile a domain definition into a network for efficiently monitoring and performing in that domain.

Given the domain independence, we are adding the following as an important objective to be achieved in the next funding period: *Objective 2: MorphII will also achieve significant (expert-level or higher) performance in a domain other than chess.* Possible domains include checkers, backgammon and Othello in which computers already excel or more difficult problem domains such as GO or organic chemical synthesis which conform naturally to our relation-based state-space search model but where progress has been slower. Collaborations are also now being developed to do whale identification (via images of their tails) and fingerprint identification through adaptive image recognition. Finally, to demonstrate robustness we set our third objective as follows: *Objective 3: Morph II will exhibit reasonable strength (determined experimentally and in competition), after training, across a large class of state-space search problems - including new ones to be developed.*

3.4 Improvements

Many significant improvements to the original Morph model have been developed over the last three years and are incorporated into MorphII. These include:

- **A domain-independent learning mechanism based on understanding the Game of Abstract Mathematical Relations**. The game of abstract mathematical relations views a state-space search problem as a collection of objects, static relations (relationships that never change, such as those denoting board topology), dynamic relations (relationships that change from state to state, such as which piece is on which square in board games), operators (as preconditions, adds and deletes defined using static and dynamic relations), initial conditions and terminal conditions.
- **Patterns developed directly from relationships specified in the rules and through rule analysis**. See below.
- **Fully-recursive analogical pattern creation and matching**. In a recent article [39] Eduardo Morales points out limitations of the old Morph system:

  > Learning from recorded games is a natural way to learn chess. Morph already claims to do so, using a fixed set of relations between the pieces[30]. Morph, however, is limited to learning patterns that express attacking/defensive relations. Thus, for instance, it is unable to learn whether two rooks are diagonally related or whether a rook is out an edge. Once a pattern has been learned by Morph, it is no longer available for the construction of other patterns.

  These remarks, although not quite completely accurate with respect to the old Morph system, have been made completely obsolete by MorphII. The previous Morph was restricted to a human supplied pattern-representation language. Morph's "graph" patterns have been subsumed by the more general mathematical form "relation" which is used in Morph II. At the same time we are attempting to restrict Morph's pattern language to a small set of mathematical primitives from which all complex patterns can be derived. These primitives include cardinality, and, or, not, join, modulus, factorability, permutation, periodicity and cycle. MorphII has access to

all relations (static and dynamic) used to define the rules of a given domain. These relations may then be augmented using the mathematical constructions above and themselves used as elements in higher-level abstract patterns. Our investigation is to study the best means of applying the mathematical transformations without yielding to the combinatorial explosion of brute force or genetic algorithms or the arbitrariness of supplying a set of "methods" [19] - but based on developing the system's understanding of how patterns are leading to misevaluations in a given domain and deliberately constructing new pattern types to alleviate those difficulties. As each pattern type creates a new "relational table" and has its own combining function, the machine generation of pattern types is equivalent to the addition of a new module in the reinforcement hierarchy.

- **Machine derivation of combining function for weights**. In the original Morph, the function that combined the weights of patterns (for state evaluation) was human-engineered. MorphII uses linear regression, gradient descent and other function learning methods to derive its own combining function.
- **Optimized incremental pattern-matching.** The pattern-matcher is now 10 to 100 times faster than previous when matching similar sets of patterns. Further, storage requirements have been reduced 25-fold. These improvements are achieved by a general domain-independent procedure that compiles the variable-based rule declaration of a given domain into a RETE-like network for monitoring the game incrementally. Specifically, the game is compiled into a partially-ordered hierarchy of relation tables (sets of tuples) each representing the dynamic relations in the domain. Operators act directly on the lowest level or primitive dynamic relations of a domain by adding or deleting tuples from these. The static relations are compiled away since, by definition, the tuples that satisfy them are always true. Patterns to be learned, simply become relations (usually made up of smaller relations) that are inserted into the hierarchy. Thus, pattern-matching can proceed as a natural part of the legal move generation mechanism - and incrementally. This incrementality may be the critical link to understanding how humans play chess so "efficiently". Also, the structure of the network allows one to recognize which useful patterns are "almost matched" - perhaps providing a basis for a subgoal-driven planning mechanism, also akin to humans. We look forward to exploring this mode.
- **Bit-level processing**. The storage and retrieval scheme above is described in the UDS paper. Since the time of its writing, we have taken things one step deeper. By compiling the relational tables into multi-dimensional bit-arrays, we are exploiting the word-level logic operations of the CPU. Thus, we are now able to process an individual domain at an efficiency level comparable to domain-engineered chess programs and should be able to compete on an equal footing.
- **Machine tuning of parameters by using "prediction error" as a domain-independent feedback mechanism.** As part of our research philosophy, outside of the rules, we are not allowed to provide domain-dependent information or heuristics or to "tune" the system's parameters. This leads to the important question of how a machine may find out whether it is improving given just "win-loss" feedback. We have discovered that the machine, by studying its relative prediction error in its evaluations from game to game, can successfully monitor itself. The learning model for MorphII is "parameterless", requiring no external tuning and settings such as a "learning rate" or "momentum coefficient". All "constants" used by the system are given initial neutral

settings and dynamically-adjusted based on performance. Building on the work of others[5, 58] we are developing learning rules for neural networks that require no externally supplied constants.

- **Incorporation of search.** Current results with Morph have been achieved using just 1-ply of search. Given that pattern-matching time has been greatly reduced, in competition we will be able to incorporate a multi-ply selective search, perhaps using a subgoal mechanism as outlined above. Researchers in the past have suggested to us that our pattern-matching approach when combined with search might be most promising: we intend to find out!

3.5 Heuristic construction through rule analysis

Eric Baum recently pointed out [1] the inherent potential in exploiting the information (mathematical structure) inherent in a declaration of the rules of a given domain:

> The computer science approach [] has since Shannon basically regarded a game as defined by its game tree. But what makes a game interesting is that it has a low complexity, algorithmically efficient definition apart from the game tree... Any procedure which only accesses the underlying simplicity of a game in the form of an evaluation function is inherently doing the wrong thing... The main open question is how to go beyond the evaluation function picture of games.

We agree strongly with this insight and wish to pursue it as deeply as we can. We feel that viewing a particular game as an instantiation of the Game of Abstract Mathematical Relations is the right direction to pursue. We intend to study the relationship between the rules declarations and the heuristics, features and values that determine strong play in that domain. By viewing things at the mathematical (domain-independent) level we hope to uncover the principles that guide intelligent search.

A number of planning and hierarchical planning algorithms originating conceptually in Means-Ends-Analysis [11], Abstrips[47] and TWEAK[3] require prioritizing the operators and conditions (or subgoals) in the domain. Such evaluation normally requires human application of domain-dependent knowledge. We argue that: **If a heuristic is a good one, the reason why it is good ought to be explainable within the mathematical structure of the state space under study**. For example, many chessplayers are told that controlling the center is important. In our paper, Distance: Towards the Unification of Chess Knowledge, we explain how this and a number of other traditional chess heuristics can be explained in terms of increasing or decreasing the "safe shortest path distance between the pieces". We believe that similar mathematical constructions can be applied to all search problems that can be stated as games of abstract mathematical relations.

In support of this line of research, Callan and Utgoff [2] have shown how useful features can be derived directly from the declarative definition of the rules of a given domain. Barney Pell goes further in a recent paper [42] by demonstrating how entire heuristics (features and values) can be developed from rule analysis.

3.6 General Function Learning

MorphII can be viewed as a general algorithm for function learning if one ignores the difficulty of assigning weights to individual states in a game (the role of temporal-

difference learning) by assuming state classifications are correct and the algorithm is given no knowledge of the rules of the game.

Given this function learning model we will be able to compare MorphII to more traditional function learning methods such as linear regression, perceptrons, nearest neighbor, and neural nets. We believe that MorphII should be able to outperform these methods in that there is no effective limit to its application of analogical reasoning and representational transformations.

It is well-known, for example, that in the NxN tile puzzles half the states are not solvable, namely those that involve an odd number of tile interchanges from the goal position. But how easy is it for a learning algorithm to produce this abstract mathematical concept given simply example states and their classification as "solvable" or "unsolvable" with no reference to specific operators or a goal state? The difficulty arises since under the standard tile-puzzle state representation as vectors or matrices, solvable states are equally syntactically close to solvable and unsolvable states, and no polynomial-sized (for the NxN case) set of partial state descriptions in themselves can prove solvability.

In particular, one must reflect on the reasoning behind the "odd number of interchanges" concept. Each tile state in an NxN puzzle represents a permutation (a mapping from 0,1,2..N to 0,1,2..N where 0 is the blank). A permutation can be uniquely described as a certain number of cycles, where a tile's successor is the tile that is currently sitting in its original home. Each interchange either breaks a cycle (if the two tiles were in the same cycle) or creates a new cycle out of two smaller ones (if the two tiles were in different cycles). Thus each interchange changes the cycle parity from odd to even or even to odd. Each tile puzzle move involves an interchange (namely a tile with a blank). Thus, half the states will have even parity and half odd.

What is required,then, is a system that can learn a concept of cycles, cardinalities and their parity and apply similar mathematical transformations to reduce the complexity of other problems. In two of the enclosed papers, we show how traditional search domains can be viewed as transformations over nested-directed hypergraphs – thus lending themselves to graph-theoretic and other discrete mathematical approaches.

3.7 Potential Impact

State-space search has remained through the years a central problem and challenge for AI researchers and other computer scientists. Recently, many domain-specific, knowledge-intensive, and "tool kit" methods have been proposed. We are striving to bring the scope of its study back to more fundamental levels by the application of mathematically well-founded tools to a sufficiently abstract representation of the problem. Our effort is supported by domain-independent software and learning algorithms that we have developed and refined as a result of our previously funded projects. Through public domain sharing and competition we are able and willing to place our algorithms in direct comparison with other algorithms across the full spectrum of search and learning problems, thus insuring the integrity of our work as well as providing a means of demonstrating their practical utility.

Both automatic programming and automatic theorem proving remain as critical problems in the evolution of computers. Although advances are being made, we feel that a

fundamental breakthrough is required. We believe that we are helping enable this breakthrough by providing a complete, well-founded, integrated framework in which representation, search, learning, inference and retrieval may be studied. The patterns and representations MorphII discovers in chess have their counterparts in the abstractions (such as lemmas and procedures) used in theorem-proving and programming.

Likewise, the areas of robotics, machine vision, and natural language understanding need a model of systems that can "think" for themselves by developing and evaluating their own abstractions and invoking representational transformations when appropriate. This flexibility, that has largely been relegated to humans in the past, will become a part of machines once they are allowed to exploit the rich mathematical structure in the world as we are allowing MorphII.

Finally, we are proposing a comprehensive learning model that shows how symbolic and subsymbolic processes may be practically combined to achieve AI's purposes. Both symbolic and subsymbolic AI have been characterized by a large number of "solutions" that consistently prove inadequate over large classes of difficult problems – and a lack of sufficient understanding of how and why the methods behave as they do. The method proposed here is based on developing a firm understanding of the mathematics of problem-domains and allowing the system itself to exploit that mathematics. **The system, being parameterless and domain-independent, requires that the method itself reflect actual computational power and robustness rather than the tuning, diligence or creative ability of the programmer.**

In summary, AI can make a tremendous advance in both practice and theory by allowing the computer (computation) to exploit the mathematics out of which itself was built. The Universal Agent is a serious step in that direction that in addition to integrating and improving on previous research efforts [18, 55, 12, 45, 36, 40, 41, 51, 44, 38, 6, 43, 52, 53, 50, 54, 48, 49, 57, 13, 16, 4, 17][1] could well have many important practical consequences.

[1] Citations here are a representative, but far from complete, sample of the research that has influenced us.

References

[1] E.B. Baum. How a bayesian approaches games like chess. In *Games: Planning and Learning, Proceedings of the AAAI Fall Symposium*, Menlo Park, CA, October 1993. AAAI Press.

[2] J. Callan and P. Utgoff. A transformational approach to constructive induction. In *Proceedings of the Eighth International Workshop on Machine Learning*, pages 122–126. Morgan Kaufmann, 1991.

[3] D. Chapman. Planning for conjunctive goals. *Artificial Intelligence*, 32(3), 1987.

[4] J. Christensen and R. Korf. A unified theory of heuristic evaluation. In *AAAI-86*, 1986.

[5] C. Darken and J. Moody. Towards faster stochastic gradient search. *Advances in Neural Information Processing Systems*, 4:1009–1016, 1992.

[6] A. D. de Groot. *Thought and Choice in Chess*. The Hague, 1965.

[7] G. Ellis and R. A. Levinson. The birth of peirce. In H.D. Pfeiffer and T.E. Nagle, editors, *Conceptual Structures: Theory and Implementation*, Lecture Notes in Artificial Intelligence 754. Springer-Verlag, 1993.

[8] G. Ellis, R.A. Levinson, and P.J. Robinson. Managing complex objects in peirce. *Special Issue on Object-Oriented Approaches in Artificial Intelligence and Human-Computer Interaction (IJMMS)*, 1994. To Appear.

[9] Gerard Ellis and Robert Levinson, editors. *Proceedings of the First International Workshop on PEIRCE: A Conceptual Graphs Workbench*. Department of Computer Science, The University of Queensland, 1992.

[10] S. Epstein and R.A. Levinson. *Proceedings of the AAAI Fall Symposium on Games: Planning and Learning*. AAAI Press, Menlo Park, 1993.

[11] G. W. Ernst and A. Newell. *GPS: A Case Study in Generality and Problem-Solving*. Academic Press, New York, 1969.

[12] N. S. Flann and T. G. Dietterich. A study of explanation-based methods for inductive learning. *Machine Learning*, 4:187–226, 1989.

[13] D. E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Welsley, Reading, MA, 1989.

[14] J. Gould and R. Levinson. Experience-based adaptive search. In R. Michalski and G. Tecuci, editors, *Machine Learning:A Multi-Strategy Approach*, volume 4, pages 579–604. Morgan Kauffman, 1994.

[15] J. Gould and R. A. Levinson. Experience-based adaptive search. In *Proc. Workshop on Machine Learning at Ninth Biennial Conf. of the Canadian Society of Computational Studies (CSCSI)*, Vancouver, Canada, May 1992.

[16] J. J. Grefenstette, editor. *Genetic algorithms and their applications*, Hillsdale, NJ, 1987. L. Erlbaum and Associates.

[17] J. H. Holland. Genetic algorithms and classifier systems: Foundations and future directions. In J. J. Grefenstette, editor, *Second International Conference on Genetic Algorithms*, Hillsdale, NJ, 1987. L. Erlbaum and Associates.

[18] H. Kaindl. Searching to variable depth chess in computer chess. In *Proceedings of IJCAI83*, pages 760–762, Karlsruhe, 1983.

[19] J. Laird, A. Newell, and P. Rosenbloom. Soar: An architecture for general intelligence. *Artificial Intelligence*, 33:1–64, 1987.

[20] D. Lenat. Eurisko: A program that learns new heuristics and domain concepts. the nature of heuristics iii: program design and results. *Artificial Intelligence*, 21(1-2), 1983.

[21] D.B. Lenat. Am: Artificial intelligence approach to discovery in mathematics as heuristic search. In R. Davis and D.B. Lenat, editors, *Knowledge-Based Systems in Artificial Intelligence.* Mcgraw-Hill, 1982.

[22] D.B. Lenat and J.S. Brown. Why am and eurisko appear to work. In *Proceedings of AAAI-83.* Morgan Kaufman, 1983.

[23] R. Levinson. *A Self-Organizing Retrieval System for Graphs.* PhD thesis, Artificial Intelligence Laboratory, University of Texas, Austin, 1985.

[24] R. Levinson. Pattern associativity and the retrieval of semantic networks. *Computers and Mathematics with Applications*, 23(6-9):573–600, 1992. Part 2 of Special Issue on Semantic Networks in Artificial Intelligence, Fritz Lehmann, editor. Also reprinted on pages 573–600 of the book, Semantic Networks in Artificial Intelligence, Fritz Lehmann, editor, Pergammon Press, 1992.

[25] R. Levinson. Distance: Towards the unification of chess knowledge. *International Computer Chess Association Journal*, 16(3):315–337, September 1993.

[26] R. Levinson. Towards domain-independent machine intelligence. In *Conceptual Graphs for Knowledge Representation*, volume 699 of *Lecture Notes in Computer Science*, pages 254–273. Springer-Verlag, 1993.

[27] R. Levinson, B. Beach, R. Snyder, T. Dayan, and K. Sohn. Adaptive-predictive game-playing programs. *Journal of Experimental and Theoretical AI*, 1992. To appear. Also appears as Tech Report UCSC-CRL-90-12, University of California, Santa Cruz.

[28] R. Levinson and G. Ellis. Multilevel hierarchical retrieval. *Knowledge-Based Systems*, 5(3):233–244, September 1992. Special Issue on Conceptual Graphs.

[29] R. Levinson and K. Karplus. Graph-isomorphism and experience-based planning. In D. Subramaniam, editor, *Proceedings of Workshop on Knowledge Compilation and Speed-Up Learning*, Amherst, MA., June 1993.

[30] R. Levinson and R. Snyder. Adaptive pattern oriented chess. In *Proceedings of AAAI-91*, pages 601–605. Morgan-Kaufman, 1991.

[31] R. A. Levinson. Uds: A universal data structure. In R. Michalski, editor, *Proc. 2ND International Conference on Conceptual Structures*, College Park, Maryland USA, 1991. To Appear. Also to appear in book, Graph-Based Knowledge Representation, Springer-Verlag 1994.

[32] R. A. Levinson. APS: An architecture for experience-based knowledge acquisition. In *Proc. Workshop on Computational Architectures for Supp orting Machine Learning and Knowledge*, Aberdeen, Scotland, July 1992.

[33] R. A. Levinson. Creativity, abduction and analogy (section introduction). In Terry Dartnall, editor, *AI and Creativity.* kluwer, 1992.

[34] R.A. Levinson. Exploiting the physics of state-space search. In *Proceedings of AAAI Symposium on Games:Planning and Learning*, pages 157–165. AAAI Press, 1993.

[35] Robert Levinson. A pattern-weight formulation of search knowledge. Technical Report UCSC-CRL-91-15, University of California Santa Cruz, 1994a. Revision to appear in Computational Intelligence.

[36] R. S. Michalski and P. Negri. An experiment on inductive learning in chess end games. In E. W. Elock and D. Michie, editors, *Machine Representation of Knowledge, Machine Intelligence*, volume 8, pages 175–192. Ellis Horwood, 1977.

[37] G. W. Mineau, B. Moulin, and J. F. Sowa, editors. *Issues in Parallel Hardware for Graph Retrieval*, 1993.

[38] S. Minton. Constraint based generalization- learning game playing plans from single examples. In *Proceedings of AAAI-84*, pages 251–254. AAAI, 1984.

[39] E. Morales. Learning patterns for playing strategies. *International Computer Chess Association Journal*, 17(1):15–26, March 1994.

[40] T Niblett and A. Shapiro. Automatic induction of classification rules for chess endgames. Technical Report MIP-R-129, Machine Intelligence Research Unit, University of Edinburgh, 1981.

[41] P. O'Rorke. A comparative study of inductive learning systems AQ11 and ID3. Technical Report Intelligent Systems Group Report No. 81-14, Department of computer Science, University of Illinois at Urbana-Champaign, 1981.

[42] B. Pell. A strategic metagame player for chesslike games. In *Proceedings of AAAI-94*, 1994. To appear.

[43] H. Pfleger and G. Treppner. *Chess: The Mechanics of the Mind.* The Crowood Press, North Pomfret, VT, 1987.

[44] J. Pitrat. A program for learning to play chess. In *Pattern Recognition and Artificial Intelligence*. Academic Press, 1976.

[45] J. R. Quinlan. Learning efficient classification procedures and their application to chess end games. In R. S. Michalski, J. G. Carbonell, and T. M. Mitchell, editors, *Machine Learning*, pages 463–482. Morgan Kaufmann, San Mateo, CA, 1983.

[46] James D. Roberts. Associative processing: A paradigm for massively parallel AI. In *AAAI Spring Symposium on Innovative Applications of Massive Parallelism in AI.* Stanford Symposium, 1993.

[47] E.D. Sacerdoti. Planning in a hierarchy of abstraction spaces. *Artificial Intelligence*, 5(2):115–135, 1974.

[48] A. L. Samuel. Some studies in machine learning using the game of checkers. *IBM Journal of Research and Development*, 3(3):211–229, 1959.

[49] A. L. Samuel. Some studies in machine learning using the game of checkers–ii recent progress. *IBM Journal of Research and Development*, 11(6):601–617, 1967.

[50] T. Scherzer, L. Scherzer, and D. Tjaden. Learning in Bebe. In T. A. Marsland and J. Schaeffer, editors, *Computer, Chess and Cognition*, chapter 12, pages 197–216. Springer-Verlag, 1990.

[51] A. D. Shapiro. *Structured Induction in Expert Systems.* Turing Institute Press with Addison-Wesley, 1987.

[52] H. A. Simon and K. Gilmartin. A simulation of memory for chess positions. *Cognitive Psychology*, 5(1):29–46, 1973.

[53] S. Skiena. An overview of machine learning in computer chess. *International Computer Chess Association Journal*, 9(3):20–28, 1986.

[54] D. J. Slate. A chess program that uses its transposition table to learn from experience. *International Computer Chess Association Journal*, 10(2):59–71, 1987.

[55] D. J. Slate and L. R. Atkin. Chess 4.5–The Northwestern University Chess Program. In P. W. Frey, editor, *Chess Skill in Man and Machine*, pages 82–118. Springer-Verlag, 1977.

[56] R. Snyder. Distance: Towards the unification of chess knowledge. Master's thesis, University of California, Santa Cruz, 1993.

[57] R. S. Sutton. Learning to predict by the methods of temporal differences. *Machine Learning*, 3(1):9–44, August 1988.

[58] R. S. Sutton. Gain adaptation beats least squares? *Proceedings of the 7th Yale Workshop on Adaptive and Learning Systems*, pages 161–166, 1992.

[59] R.S. Sutton. Special issue on reinforcement learning. *Machine Learning*, 1991.

[60] C. Wescott and R. A. Levinson. Experience-based music composition. In *Proceedings AAAI Spring Symposium on AI and Creativity*, pages 50–56, 1993.