# A Possible Architecture for Conquering the Game of Go

## Extended Paper Proposal for Work in Progress

A. Beavers, M. Blankenship, C. Harrison, T. Martin, D. Reisinger, & J. Simerly
*The Machine Learning Research Group, Digital Humanities Laboratory*
(http://digitalhumanities.evansville.edu/)
*The University of Evansville*

## Introduction

Over the past two decades, research in artificial intelligence has made great strides in creating machines capable of defeating human players in games requiring a high degree of cognitive ability. The defeat of Chess Master Garry Kasparov by IBM's *Deep Blue* in 1997 signaled the birth of a new era in machine learning, even though *Deep Blue* did not play Chess the same way a human does. The recent defeat of Ken Jennings and Brad Rutter by IBM's *Watson* at Jeopardy earlier this year was all the more impressive while making it all the more surprising that a *formal* game (in the sense defined by Haugeland 1985), like Go, still remains out of the reach of current AI (Cai and Wunsch 2010, Harré 2011). One would think that if Chess and Jeopardy can be conquered by AI, Go should be an easy target, since it involves pieces that, unlike Chess, are equivalent in their value and their playability on the game board. Chess, for instance, has pieces with different properties: a pawn cannot do what a rook can do, a rook cannot act like a bishop, etc. In Go, on the other hand, all pieces have the same properties, save that half of them are white, indicating the moves of one player, and half are black, indicating the moves of the other.

The surprising simplicity of Go and its rule set (see http://gobase.org/studying/rules/) would seem to make the game a simple computational problem. But it is not. One possibility for this counter-intuitive situation may be tied to the fact that playability in Go is not determined directly by the pieces on the board, but by emergent patterns at a higher level of abstraction that have properties that individual pieces do not have. Analysis of the game space cannot therefore be understood by a simple inspection of the pieces themselves. It must involve detection and inspection of higher level patterns. Finally, in order to conquer the game, the mechanisms must have some ability to determine on the basis of pattern detection what the best moves and counter-moves are in order to win the game. All of this together suggests that a hybrid AI may be necessary in order to achieve success in this arena.

In the following, we briefly sketch such a hybrid architecture that we intend to build over the next two years, though partial progress has already been made. This architecture consists of three processing layers: an agent-based layer (ABL) representing the game space where user and machine can interact, a feature detection layer (FDL) on which higher level patterns are isolated and inspected, and a decision layer (DL) that ultimately responds to the FDL and modulates the parameters that govern local behavior at the ABL. Below, we will also discuss the broader implications of such a project. As a hint to the latter, the model in essence allows the behavior of local agents to be modulated by global variability that itself emerges from local behavior thereby exhibiting "downward causation" (see Davies 2006). Models that accomplish this task may provide insight on a broad range of scientific and philosophical issues.

## Summary of the Proposed Architecture

There is nothing particularly mysterious or difficult about the ABL. In fact, Minesweeper, a game that uses individual agents in relation to their neighbors, ships with NetLogo in its "Models Library." For the ABL, we are in fact using NetLogo, which has a handy API for connecting to MySQL. This database package has proven to be useful in past research for computing networks at sufficient speeds and so stands as a good candidate for testing. Since we will need to pass information to the FDL, which will be a multi-dimensional network, this API is particularly useful for our purposes. (If for some reason this is too slow, we will use other means. Several techniques are already available for testing.)

Normally, Go is played on a 19 x 19 grid space, but for the purposes of this project, we are using a smaller 13 x 13 space. This strategy is common in learning Go, and it will save us some work at the FDL. We are currently designing the ABL of the architecture with the simple goal of creating a model that can adhere to the Go rule set, even if its play is sloppy, as it should be without feature-detection and some form of machine learning. We are close to finishing this part of the project, though at the time of this writing, no attempt has been made to connect it to the FDL.

One of the central difficulties involved in completing this project is detecting patterns on the game board. Since the patterns can vary in size and appear on different places of the board, a one to one isomorphic map of the grid space is unhelpful. In other words, storing the individual pieces as they are arranged on the board space as a binary map is insufficient. It is not the placement of the individual pieces that concern us, *but rather the relationships between them* without regard to where they appear on the grid. Thus, the individual patterns must somehow be decoupled from their fixed locations on the board. Fortunately, we have solved this problem for simple shapes using an array of biologically-inspired visual feature detection networks (Beavers 2009) that permit "dimension-shifted" representations, where "dimension-shifted" means that the output representation is unlike or different than the input representation while still retaining the important information from the input needed for a particular information processing task (Clark 2000).

Because our early visual model cannot track two shapes on the same game board, it is necessary to increase the number of networks so that the combined output signature of all of them together is a representation of the relationship of every piece on the game board to every other piece. Though the network design is fairly simple, it will be significantly large. It will have to contain thirteen networks to represent the black and white pieces together, another thirteen for the black pieces, and then another thirteen for the white pieces. After passing the current state of the game board through this multi-dimensional network, the combined output will be a single vector that represents the spatialized array "de-spatialized," that is, dimension-shifted, so that the important information about the relationship of pieces on the board that we need is preserved. In turn, this vector can serve as input for genetic algorithms, conventional neural networks, ABM decidability networks (as outlined in Grim et. al. 1998, section 6.3), or dynamic associative networks (Beavers 2010a & b).

Construction on the FDL has already begun by using the "puzzle piece resolution" (PPR) adopted from our earlier visual detection network (Beavers 2009). The idea behind PPR is to take a fixed pattern on a grid as input and then have a network rearrange it to output a set of decomposed partial patterns that can only be reassembled in one way, that way being a copy of the original shape on the grid. This decomposition of an overall fixed grid into pieces produces a signature that represents the original shape on the grid regardless of its placement. Thus the network "recognizes" a 3 x 3 square, for instance, as such, whether it appears in the top right, bottom left, or anywhere else on the input grid.

The part of PPR that makes it work is that the FDL is sensitive not only to the pieces on the board but also to what is next to them. This sensitivity explains why thirty nine separate networks are needed to complete the FDL. The first one, for instance, tracks pieces in relation to their immediate neighbors. The second tracks pieces in relation to neighbors that are two removes away on the gird. The third tracks neighbors at three removes, etc. Because the grid space is 13 x 13, thirteen networks are needed to capture the entire set of relations that define the patterns on the board. However, since we are dealing with a two-player game involving two "forces" (white and black), it is necessary also to track each set of pieces and their overall patterns individually so that the DL can determine when and where to play; thus the need for thirty-nine networks. The output of the FDL as a whole is a binary vector that once again does not represent the pieces on the board, but the patterns that emerge from the relationships between them. This dimension-shifted signature is the input for the DL.

We have yet to decide on the best architecture for the DL, though it's purpose is clear, namely to output a set of values that modifies all or a portion of the parameters that determine local agent behavior at the ABL. This output may or may not affect all of the agents in the same way. Several candidates are on the table for automating the needed decisions. One obvious method is the use of genetic algorithms to allow the overall mechanism to learn precisely how and when to adjust its parameters in light of the current state of the game board. However, Beavers and Harrison are currently exploring another approach, that of transforming ordinary networks into teleodynamic and predictive mechanisms using a dynamic systems approach (see Beavers 2010a, b; Beavers & Harrison 2011). This strategy involves another type of network that we have labeled as a "dynamic associative network" (DAN) to distinguish it from an artificial neural network (ANN).

DANs are built up from the data to transform input into output in such a way that the output is predictive, that is, these models learn from experience to form associations based on some sort of statistical procedure that is implicitly determined by the model. (That is, no explicit statistical methods are used.) Developed in the Digital Humanities Laboratory at the University of Evansville over the past four years, DANs have been used in a variety of microworld experiments and exhibit cognitive abilities including 1) object identification based on properties and context-sensitivity, 2) comparison of similarities and differences among properties and objects, 3) shape recognition of simple shapes regardless of where they might appear in an artificial visual field (the inspiration for the FDL discussed above), 4) association across simulated sense modalities, 5) primary sequential memory of any seven digit number (inspired by Allen and Lange 1995), 6) network branching from one subnet to another based on the presence of a single stimulus, 7) eight-bit register control that could perform standard, machine-level operations as with standard Turing-style computational devices, and 8) rudimentary natural language processing based on a stimulus/response (i.e. anti-Chomskian) conception of language (Beavers 2009). The guiding principle of this research has been the search for "initial intelligence," or a basic set of low level

skills that might add up to explain larger scale cognitive initiatives.

Unlike ANNs, DANs learn by adding nodes and connections wherever needed. In more recent applications, we have reintroduced both weights and thresholds to try to improve cognitive function. To be sure, at present, we are not prepared to offer anything close to a complete explanation of cognition. Rather, this experimental enterprise has been dedicated solely to trying to determine which network shapes and control parameters might account for various cognitive skills. We have learned, for instance, that association of objects taking their proper context into account requires a "fat" network that uses a lot of overlap between connections, while sequential memory requires a "slim" network with hardly any overlap.

In our latest experiments, we have been working to isolate control parameters to transform an ordinary database into a predictive mechanism in order to create content-addressable memory, the hallmark of true cognitive artificial intelligence (Copeland 1993; Haugeland 1985). With regard to the current project, this approach should prove useful for isolating relational parameters between nodes to govern local behavior. Part of our rationale for thinking so is that these networks are not only predictive, they are also teleodynamic, a property aptly described in Deacon 2006. A teleodynamic system is any system that can remember when it is in a state that it has been in before and can act accordingly to redirect its behavior in a goal-oriented fashion. Memory of past experience is key here, which, in turn, allows for the possibility of goal-directed or teleological behavior by machines governed by standard Newtonian "push" causation (Deacon 2006). In the case of our latest generation of DANs, a network's past experience is transformed into a set of attractors and repulsors that can guide future action. Here, *the success or failure of past experience is transformed into a recipe for future action*.

We have not yet begun construction on a DAN-based DL because we are still in the process of experimenting with different implementations (Beavers & Harrison 2011). But we are holding out hope that with the proper input DANs at the DL will find the right set of attractors and repulsors for setting the parameters at the ABL. If not, we will try some of the approaches already on the table.

Our tentativeness here is intentional: one *caveat* must be emphasized, lest we be charged with trying to pass off "vaporware" as a working model. This is very much a work in progress, and what appears above is merely a proposal that we believe may have a high probability of success. This alone makes the project worth pursuing, though we do not anticipate that a complete model will be finished during this year. Thus, we put this proposal forward for a presentation that addresses both successes and failures with the hope of getting feedback concerning problems that emerge along the way.

## Project Significance

As with most projects in AI, our distant goal here is not to solve the immediate problem at hand, in this case how to get a computer to master the game of Go, but to learn something about information processing. This project is particularly interesting since it allows the collective effects of agents acting locally to modulate their behavior by permitting global effects to feed back into the model's lower layers. There are several examples of such behavior in nature and society. Regarding the latter, a pertinent example is the way that laws and customs emerge from the transgressions and practices of individual agents which, in turn, partially determine their behavior. Regarding the former, we can consider the way that placing a rock in a stream spins up an eddy current that then alters the flow of water around it.

A more pertinent example for our purposes comes from the biology of real neural firings and their interaction with other bodily systems. Here, local neural firings can trigger a response that bathes parts of the brain in various neurotransmitters, in turn, modulating neuronal activity. (Consider, for instance, how the perceived threat of danger can trigger a shift from the parasympathetic to the sympathetic nervous systems to force a biological agent into a "fight or flight" mode.) Here, local effects trigger a global response that alter the very local effects that produced it while preserving (or attempting to preserve) homeostatic equilibrium. The proposed architecture, while certainly not a biological system, is partly analogous to this process and may serve to show one way in which downward causation is possible. This significance is clear by standing back and taking a broader look at the architecture here proposed.

Basically, the proposed hybrid model consists of a local layer of agent activity, a network structure for monitoring that activity and a higher-level network structure to change the parameters that govern the local layer to allow the model to "fight it out" intelligently in order to achieve a goal state, which is success at the game. Furthermore, the higher level network functions on the basis of attractors and repulsors that, like hormones, attract the mechanism toward certain states and repel it from others. Here, activity is not promulgated solely from agent to agent; rather, the collective effect of agents remodulates their own control parameters by way of sensors that self-monitor the state of the system. At the same time, behavior is here governed by "ruts carved out in a state space," to use a rough metaphor, that allows the system to learn from past mistakes and successes.

It is important to note that none of this is done using traditional algorithmic techniques. There is no tree branching or other heuristics to govern machine performance, just local agent activity, dimension-shifted pattern matching, memory, attraction and repulsion. The categories of traditional AI are, in other words, not applicable. This fact points to another significance of this project if we are suc-

cessful: if such an architecture will work for Go, then why not a related architecture for Chess? If so, we should be one step closer to creating a machine that plays Chess the way a human plays Chess. Of course, in this regard, it is wise to heed the words of Turing that "if a machine is expected to be infallible, it cannot also be intelligent" (Turing 1947/2004, 394).

## References Under Consideration


Abbott, R. (2006). Emergence explained: Abstractions: Getting epiphenomena to do real work. *Complexity 12*, 1, 13-26.

Allen, C., & Lange, T. (1995). Primary sequential memory: An activation-based connectionist model. *Neurocomputing 11*, 2-4, 227-243.

Beavers, A. (2009). Mechanical vs. symbolic computation: Two contrasting strategies for information processing. Society for Machines and Mentality, American Philosophical Association, New York City, December 27th-30th.

Beavers, A. (2010a). More fun with jets and sharks: Typicality effects and the search for the perfect attractors. North American Meeting of the International Association for Computing and Philosophy (IACAP), Simulations and Their Philosophical Implications, Carnegie Mellon University, Pittsburgh, July 24th-26th.

Beavers, A. (2010b). Typicality effects and resilience in evolving dynamic networks. In FS-10-03, AAAI Press.

Beavers, A., & Harrison, C. (2011). Hybrid networks: Transforming networks for social and textual analysis into teleodynamic and predictive mechanisms. Institute for Advanced Topics in the Digital Humanities, Networks and Network Analysis for the Humanities, University of California, Los Angeles, October 20th-22nd.

Berlekamp, E., & Wolfe, D. (1994). *Mathematical Go: Chilling gets the last point*. Wellesley, MA: A. K. Peters.

Boschetti, F. et. al. (2005). Defining and detecting emergence in complex networks. *Knowledge-Based Intelligent Information and Engineering Systems 3684*, 573-580.

Bouzy, B. (1996). Spatial Reasoning in the game of Go. *Workshop on Representations and Processes in Vision and Natural Language, European Conference on Artificial Intelligence,* 78–80.

Burmeister, J., & Wiles, J. (1995). The challenge of Go as a domain for AI research: A comparison between Go and Chess. *Proceedings of the Third Australian and New Zealand Conference on Intelligent Information Systems*, 181-186.

Cai, X., & Wunsch, D. (2010). Computer Go: A grand challenge to AI. In Duch, W., & Mańdziuk, J. (Eds.), *Challenges for computational intelligence* (pp. 443-465). Berlin: Springer.

Cai, X., & Wunsch, D. (2004). Evolutionary computation in playing CaptureGo game. *Proceedings of the International Conference on Cognitive and Neural Sciences*, Boston University, Boston, May 19th-22nd.

Clark, A. (2000). *Mindware: An Introduction to the philosophy of cognitive science*. Oxford, UK: Oxford. See chapter four in particular.

Copeland, J. (1993). Artificial intelligence: A philosophical introduction. Cambridge, MA: Blackwell.

Davies, P. (2006). The physics of downward causation. In Clayton, P., & Davies, P. (Eds.), *The re-emergence of emergence* (pp. 33-52). Oxford, UK: Oxford.

Deacon, T. (2006). Emergence: The hole at the wheel's hub. In Clayton, P., & Davies, P. (Eds.), *The re-emergence of emergence* (pp. 111-150). Oxford, UK: Oxford.

Grim, P., et. al. (1998). *The philosophical computer: Exploratory essays in philosophical computer modeling*. Cambridge, MA: MIT Press.

Harré, M., et. al. (2011). The aggregate complexity of decisions in the game of Go. *The European Physical Journal B*. Published online 24 March.

Haugeland, J. (1985). *Artificial intelligence: The very idea*. Cambridge, MA: MIT Press.

MacKay, D. (2003). *Information theory, inference, and learning algorithms*. New York, NY: Cambridge University Press.

Muller, M. (2003). Conditional combinatorial games and their application to analyzing capturing race in Go. *Information Science 154*, 189-202.

Muller, M. (2002). Position evaluation in computer Go. *International Computer Games Association Journal 25*, 4, 219-228.

Richards, N., et. al. (1998). Evolving neural networks to play Go. *Applied Intelligence 8*, 85-96.

Sawyer, R. (2001). Simulating emergence and downward causation in small groups. *Multi-Agent-Based Simulation 1979*, 49-67.

Sutton, R. (1988). Learning to predict by the method of temporal differences. *Machine Learning 3*, 9-44.

Turing, A. (1947/2004). Lecture on the Automatic Computing Engine. In Copeland, J. (Ed.), *The essential Turing: The ideas that gave birth to the computer age*, (pp. 375-394). Oxford, UK: Oxford.

Zobrist, A. (1969). A model of visual organization for the game of Go. *Proceedings of the Spring Joint Computer Conference of the Association for Computing Machinery*, 103–112.