Novamente: An Integrative Architecture for General Intelligence

Moshe Looks[1], Ben Goertzel[2] and Cassio Pennachin[2]

1. Object Sciences Corporation
6359 Walker Lane Suite 100
Alexandria, VA 22310
mlooks@objectsciences.com

2. Novamente LLC
11160 Veirs Mill Road, L-15 Suite 161
Wheaton, MD 20902
ben@novamente.net, cassio@novamente.net

Abstract

The Novamente AI Engine is briefly reviewed. The overall architecture is unique, drawing on system-theoretic ideas regarding complex mental dynamics and associated emergent patterns. We describe how these are facilitated by a novel knowledge representation which allows diverse cognitive processes to interact effectively. We then elaborate the two primary cognitive algorithms used to construct these processes: probabilistic term logic (PTL), and the Bayesian Optimization Algorithm (BOA). PTL is a highly flexible inference framework, applicable to domains involving uncertain, dynamic data, and autonomous agents in complex environments. BOA is a population-based optimization algorithm which can incorporate prior knowledge. While originally designed to operate on bit strings, our extended version also learns programs and predicates with variable length and tree-like structure, used to represent actions, perceptions, and internal state. We detail some of the specific dynamics and structures we expect to emerge through the interaction of the cognitive processes, outline our approach to training the system through experiential interactive learning, and conclude with a description of some recent results obtained with our partial implementation, including practical work in bioinformatics, natural language processing, and knowledge discovery.

Introduction and Motivation

The primary motivation behind the Novamente AI Engine is to build a system that can achieve complex goals in complex environments, a synopsis of the definition of intelligence given in (Goertzel 1993). The emphasis is on the plurality of *goals* and *environments*. A chess-playing program is not a general intelligence, nor is a data mining engine, nor is a program that can cleverly manipulate a researcher-constructed microworld. A general intelligence must be able to carry out a variety of different tasks in a variety of different contexts, generalizing knowledge between contexts and building up a context and task independent pragmatic understanding of itself and the world.

First among the tenets underlying the design is an understanding of mind as the interpenetration of a physical system with an abstract set of *patterns*, where a pattern is quantified in terms of algorithmic information theory (Goertzel 1997, Chaitin 1987). In essence, a pattern in an entity is an abstract program that is smaller than the entity, and can rapidly compute the entity or its approximation. For instance, a pattern in a drawing of a sine curve might be a program that can compute the curve from a formula.

The understanding of mind as pattern ties in naturally with the interpretation of intelligence, at the most abstract level, as a problem of finding compact programs that encapsulate patterns in the environment, in the system itself, and in behavior. This concept was first seriously elaborated in Solomonoff's work on the theory of induction (Solomonoff 1964, Solomonoff 1978), and has been developed more rigorously and completely in Hutter's recent work (Hutter 2000), which integrates a body of prior work on algorithmic information theory and statistical decision theory to formalize the concept of general intelligence. Novamente can be proven to be arbitrarily intelligent according to Hutter's definition, if given sufficient computational resources. Baum has expounded the cognitive science implications of this perspective on intelligence (Baum 2004).

Another tenet of the Novamente approach is the realization that intelligence most naturally emerges through situated and social experience. Abstract thoughts and representations are facilitated through the recognition and manipulation of patterns in environments with which a system has sensorimotor interaction; see for example (Boroditsky and Ramscar 2002). This interaction, embodied in the right cognitive architecture, leads to autonomy, experiential interactive learning, and goal-oriented self-modification; a mind continually adapts based on what it learns from its environment and the entities it interacts with.

The final tenet is a view of the internal organization of a mind as a collection of semi-autonomous agents embedded in a common substrate (Goertzel 1993). In this vein, Novamente is less extreme than some alternative approaches such as Minsky's Society of Mind (Minsky 1986), where agents are largely independent. Novamente is based on the idea that minds are self-organizing systems of agents, which interact with some degree of individual freedom, but are also constrained by an overall architecture involving a degree of inbuilt executive control, which nudges the self-organizing dynamics towards emergent hierarchical organization.

These abstract principles are coherently unified in a philosophy of cognition called the *psynet model* (Goertzel, 1997), which is foundational to Novamente, and provides a moderately detailed theory of the emergent structures and dynamics in intelligent systems. In the model, mental functions such as perception, action, reasoning and procedure learning are described in terms of interactions between agents. Any mind, at a given interval of time, is assumed to have a particular goal system, which may be expressed explicitly and/or implicitly. Thus, the dynamics of a cognitive system are understood to be governed by two main forces: self-organization and goal-oriented behavior. More specifically, several primary dynamical principles are posited, including:

Association. Patterns, when given attention, spread some of this attention to other patterns that they have previously been associated with in some way. Furthermore, there is Peirce's "law of mind" (Peirce 1892), which could be paraphrased in modern terms as stating that the mind is an associative memory network, whose dynamics dictate that every idea in the memory is an active agent, continually acting on those ideas with which the memory associates it.

Differential attention allocation. Patterns that have been valuable for goal-achievement are given more attention, and are encouraged to participate in giving rise to new patterns.

Pattern creation. Patterns that have been valuable for goal-achievement are mutated and combined with each other to yield new patterns.

Credit Assignment. Habitual patterns in the system that are found valuable for goal-achievement are explicitly reinforced and made more habitual.

Furthermore, the network of patterns in the system must give rise to the following large-scale emergent structures:

Hierarchical network. Patterns are habitually in relations of control over other patterns that represent more specialized aspects of themselves.

Heterarchical network. The system retains a memory of which patterns have previously been associated with each other in any way.

Dual network. Hierarchical and heterarchical structures are combined, with the dynamics of the two structures working together harmoniously.

Self structure. A portion of the network of patterns forms into an approximate (fractal) image of the overall network of patterns.

Structures and Algorithms

The psynet model does not tell you how to build a mind, only, in general terms, what a mind should be like. It would be possible to create many different AI designs based loosely on the psynet model; one example of this is the Webmind AI Engine developed in the late 1990's (Goertzel et al. 2000, Goertzel 2002). Novamente, as a specific system inspired by the psynet model, owes many of its details to the limitations imposed by contemporary hardware performance and software design methodologies. Furthermore, Novamente is intended to utilize a minimal number of different knowledge representation structures and cognitive algorithms.

Regarding knowledge representation, we have chosen an intermediate-level *atom network* representation which somewhat resembles classic semantic networks but has dynamic aspects that are more similar to neural networks. This enables a breadth of cognitive dynamics, but in a way that utilizes drastically less memory and processing than a more low-level, neural network style approach. The details of the representation have been designed for compatibility with the system's cognitive algorithms.

Regarding cognition, we have reduced the set of fundamental algorithms to two: Probabilistic Term Logic (PTL) and the Bayesian Optimization Algorithm (BOA). The former deals with the local creation of pieces of new knowledge from existing pieces of knowledge; the latter is more oriented towards global optimization, and creates new knowledge by integrating large amounts of existing knowledge. These two algorithms themselves interact in several ways, representing the necessary interdependence of local and holistic cognition.

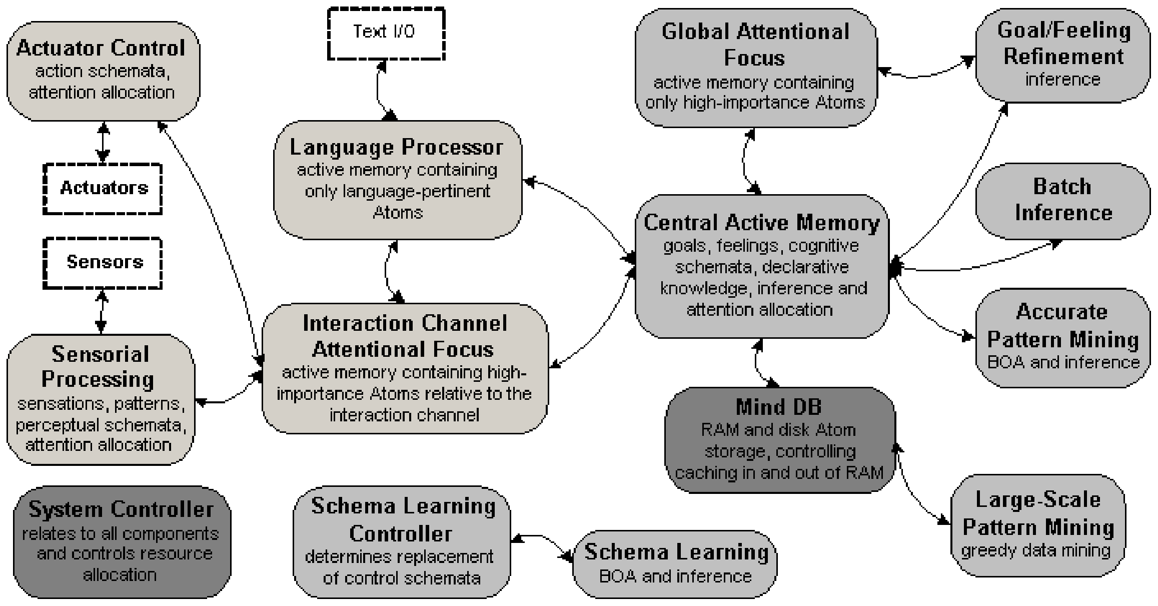
Having reduced the basic knowledge representations and cognitive algorithms to this minimal core, the diverse functional specializations required for pragmatic general intelligence are provided by the introduction of a number of node and link types in the atom network, and a high-level architecture consisting of a number of functionally specialized *lobes* each deploying the same structures and algorithms for particular purposes (see Figure 1 below).

Knowledge Representation

Knowledge representation in Novamente involves two levels, the explicit and the emergent. This section focuses on the explicit level; the emergent level involves self-organizing structures called *maps*, and will be discussed later, after the fundamental cognitive dynamics have been introduced.

Explicit knowledge representation in Novamente involves discrete units (atoms) of several types: *nodes*, *links*, and *containers*, which are ordered or unordered collections of atoms. Each atom is associated with a *truth value*, indicating, roughly, the degree to which it correctly describes the world. Novamente has been designed with several different types of truth values in mind; the simplest of these consists of a pair of value denoting probability and weight of evidence. All atoms also have an associated *attention value*, indicating how much computational effort should be expended on them. These contain two values, specifying short and long term importance levels.

Novamente node types include tokens which derive their meaning via interrelationships with other nodes, nodes representing perceptual inputs into the system (e.g., pixels,



Actuator Control
action schemata, attention allocation
Text I/O
Global Attentional Focus
active memory containing only high-importance Atoms
Goal/Feeling Refinement
inference
Actuators
Language Processor
active memory containing only language-pertinent Atoms
Batch Inference
Sensors
Central Active Memory
goals, feelings, cognitive schemata, declarative knowledge, inference and attention allocation
Interaction Channel Attentional Focus
active memory containing high-importance Atoms relative to the interaction channel
Accurate Pattern Mining
BOA and inference
Sensorial Processing
sensations, patterns, perceptual schemata, attention allocation
Mind DB
RAM and disk Atom storage, controlling caching in and out of RAM
System Controller
relates to all components and controls resource allocation
Schema Learning Controller
determines replacement of control schemata
Schema Learning
BOA and inference
Large-Scale Pattern Mining
greedy data mining

Figure 1. Each component is a Lobe, which contains multiple atom types and mind agents. Lobes may span multiple machines, and are controlled by schemata which may be adapted/replaced by new ones learned by Schema Learning, as decided by the Schema Learning Controller. The diagram shows a configuration with a single interaction channel, that contains sensors, actuators and linguistic input; real deployments may contain multiple channels, with different properties.

points in time, etc.), nodes representing moments and intervals of time, and procedures (described below). Links represent relationships between atoms, such as fuzzy set membership, probabilistic logical relationships, implication, hypotheticality, and context. The particular types and subtypes used, and the justifications for their inclusion, are omitted for brevity.

Procedures in Novamente are objects which produce an output, possibly based on a sequence of atoms as input. They may contain *generalized combinator trees*. These are computer programs written in *sloppy combinatory logic*, a language that we have developed specifically to meet the needs of tightly integrated inference and learning.

Combinatory logic (CL) is a simple yet Turing-complete computational system (Curry and Feys 1958). The basic units are combinators, which are higher order functions that always produce new higher order functions when applied. Beyond combinators, our language contains numbers, arithmetic operators, looping constructs, and conditionals. Procedures may also contain embedded references to other procedures. Two unique features of this language that are advantageous for our purposes are that programs can be expressed as binary trees where the program elements are contained in the leaves, and that variables are not necessary (though they may be introduced where useful). Furthermore, we have generalized the evaluation system of combinatory logic so there are no type restrictions on programs, allowing them to be easily modified and evolved by Novamente's cognitive processes (hence *sloppy)*. A full exposition of the language is omitted for brevity; see (Looks, Goertzel, and Pennachin 2004).

Schemata and predicates are procedures that output atoms and truth values, respectively. Special-purpose predicates, instead of containing combinator trees, represent specific queries that report to the Novamente system some fact about its own state. Predicates may also be designated as *goal nodes*, in which case the system orients towards making them true.

Cognitive Algorithms

Novamente cognitive processes make use of two main algorithms, Probabilistic Term Logic (PTL), and the Bayesian Optimization Algorithm (BOA), described below.

Probabilistic Term Logic

PTL is a highly flexible inference framework, applicable to many different situations, including inference involving uncertain, dynamic data and/or data of mixed type, and inference involving autonomous agents in complex environments. It was designed specifically for use in Novamente, yet also has applicability beyond the Novamente framework; see (Goertzel et al. 2004) for a full exposition. The goals motivating the development of PTL were the desire to have an inference system that:

- Operates consistently with probability theory, when deployed within any local context (which may be adaptively identified).
- Deals rapidly (but not always perfectly accurately) with large quantities of data, yet allows arbitrarily accurate and careful reasoning on smaller amounts of information, when this is deemed appropriate.

- Deals well with the fact that different beliefs and ideas are bolstered by different amounts of evidence (for an explanation of how traditional probabilistic models fail here, see Wang 1993).
- Enables robust, flexible analogical inference between different domains of knowledge (Indurkhya 1992)
- Does not require a globally consistent probability model of the world, but is able to create locally consistent models of local contexts, and maintain a dynamically-almost-consistent overall world-model, dealing gracefully with inconsistencies as they occur.
- Encompasses both abstract, precise mathematical reasoning and more speculative hypothetical, inductive, and/or analogical reasoning.
- Encompasses the inference of both declarative and procedural knowledge.
- Deals with inconsistent initial premises by dynamically iterating into a condition of "reasonable almost-consistency and reasonable compatibility with the premises", thus, for example, perceiving sensory reality in a way compatible with conceptual understanding, in the manner portrayed by Gestalt psychology (Kohler, 1993) and developed in the contemporary neural network literature, see (Haikonen 2003).
- Makes most humanly simple inferences appear brief, compact and simple. For a sustained argument that term logic exceeds predicate logic in this regard, see (Sommers and Englebretsen, 2000).

One difference between PTL and standard probabilistic frameworks is that PTL deals with multivariable truth values. Its minimal truth value object has two components: strength and weight of evidence. Alternately, it can use probability distributions (or discrete approximations thereof) as truth values. This, along with the fact that PTL PTL does not assume that all probabilities are estimated from the same sample space, makes a large difference in the handling of various realistic inference situations.

Another difference is PTL's awareness of context. The context used by PTL can be universal (everything the system has ever seen), local (only the information directly involved in a given inference), or many levels between. This provides a way of toggling between more rigorous and more speculative inference, and also a way of making inference consistent within a given context even when a system's overall knowledge base is not entirely consistent.

First-order PTL deals with probabilistic inference on (asymmetric) inheritance and (symmetric) similarity relationships, where different Novamente link types are used to represent intensional versus extensional relationships (Wang, 1995). The inference rules here are deduction ($A \rightarrow B$, $B \rightarrow C$ |- $A \rightarrow C$), inversion (Bayes rule), similarity-to-inheritance-conversion, and revision (which merges different estimates of the truth value of the same atom). Each inference rules comes with its own quantitative truth value formula, derived using probability theory and related considerations. Analogical inference is obtained as a combination of deductive and inversive inference, and via their effects on map dynamics (described later).

Higher-order PTL deals with inference on links that point to links rather than nodes, and on predicates and schemata. The truth value functions here are the same as in first-order PTL, but the interpretations of the functions are different. Variable-free inference using combinators and inference using explicit variables and quantifiers are both supported; the two styles may be freely intermingled.

The Bayesian Optimization Algorithm

BOA is a population-based optimization algorithm that works on bit strings. BOA significantly outperforms the genetic algorithm on a range of simple optimization problems by maintaining a centralized probabilistic model of the population it is evolving (Pelikan, Goldberg, and Cantú-Paz 1999; Pelikan 2002). When a candidate population has been evaluated for fitness, BOA seeks to uncover dependencies between the variables that characterize "good" candidate solutions (e.g., the correct value for position 5 in the genome depends on the value at position 7), and adds them to its model. In this way, it is hoped that BOA will explicitly discover and utilize probabilistic "building blocks", which are then used to generate new candidate solutions to populate the next generation. The basic algorithm is as follows (adapted from Pelikan 2002):

(1) Generate a random initial population P(0).
(2) Use the best instances in P(t) to learn a model M(t).
(3) Generate a new set of instances O(t) from M(t).
(4) Create P(t+1) by merging O(t) and P(t) according to some criteria.
(5) Iterate steps (2) through (4) until termination criteria are satisfied.

This algorithm tends to preserve good collections of variable assignments throughout the evolution, and can explore new areas of the search space in a more directed and focused way than GA/GP, while retaining the positive traits of a population-based optimization algorithm (diversity of candidate solutions and non-local search).

For Novamente, we have extended BOA to evolve programs, written in our sloppy combinatory logic representation, rather than bit strings (Looks, Goertzel, and Pennachin 2004). Previous work by Ocenasek (Ocenasek 2002) has extended binary BOA to fixed-length strings with non-binary discrete and continuous variables. There is a fundamental difference between learning fixed length strings and programs. In the former, one is evolving individuals with a fixed level of complexity and functionality. In order to evolve non-trivial program trees however, one must rely on incremental progress; a simpler program accretes complexity over time until it is correct. While BOA as described above is effective at optimizing and preserving small components, it does not innately lead to the addition of new components. In order to remedy this, we have added probabilistic variables to the instance generation process that, when activated, have effects similar to crossover in genetic programming; see (Looks, Goertzel, and Pennachin 2004) for details and examples.

We have also begun using BOA to discover surprising patterns in large bodies of data, using an approach we call *pattern mining*. In this application, BOA is given a number of predefined predicates, such as `isTall(X)`, `loves(X,Y)`, `isMale(X)`, etc. Patterns are logical combinations of predicates, e.g., `isTall(X) AND isMale(X)`. Based on the overall philosophy behind Novamente, patterns are evaluated based on "interestingness" which is composed of two factors: pattern-intensity, and novelty relative to the system's current knowledge base. Pattern-intensity refers to how well a pattern compresses regularities in the system's knowledge; this can be quantified as the difference between the actual frequency of the expression, and the frequency that would be computed assuming probabilistic independence. In a domain consisting of random men and women, the example given above would be intense, because tallness correlates with maleness. If this correlation were not known to the system nor easily derivable by the system, then it would be significant and acceptably novel, thus being considered "interesting".

This pattern mining approach might run into scalability issues, as the predicate space tends to be very large. We can resolve this problem by encoding the fact that varying degrees of similarity exist between predicates. When similarity is meaningfully quantified, an important cognitive mechanism used in creative thought called *slippage* comes into play, where ideas are transformed by substituting one concept for another in an intelligent, context-dependant fashion (Hofstadter 1986). We incorporate prior similarity information by embedding predicates in real spaces, so that similar predicates are embedded close to each other. When a refinement of this approach is used, BOA can construct new procedures that make use of existing ones, leading to hierarchical design.

A number of factors make our modified BOA variant advantageous for use within Novamente. The centralized probabilistic model used can be constructed with the aid of PTL inference and prior knowledge, allowing them to be used in instance generation. Contrariwise, fitness evaluation of instances generated by BOA with a particular model can be used to revise existing knowledge in the system, and infer new knowledge. As with most evolutionary techniques, BOA can also be used to perform a number of learning tasks inside Novamente, such as categorization, unsupervised clustering of atoms, and function learning, by using appropriate fitness functions.

Cognitive Processes

This section presents the most important cognitive processes that take place in Novamente. All of the cognitive processes described below are encapsulated within one or more *mind agents*; software objects which dynamically update the atoms in the system on an ongoing basis.

In the present Novamente code, mind agents are coded directly in C++. However, our intention is to ultimately replace these C++ objects with schemata written in sloppy combinatory logic, which will enable Novamente to reason about and modify its own cognitive mechanisms, thus allowing thoroughly self-modifying and self-improving general intelligence. The execution of this plan awaits only the implementation of some optimizations in the schema execution framework.

Inference

The most direct application of PTL is to seek interrelations between sets of important, apparently interrelated atoms, leading to the creation of links and predicates joining important atoms. As a compact way of storing/computing when entities are interrelated, dimensional embeddings is used. Additionally, important predicates are evaluated, and links are constructed between them. This process is the primary creator of relations between atoms in Novamente.

Learning of Cognitive Schemata

A critical issue here is *inference control*, which is provided by a combination of hard-coded algorithms and learned programs called *cognitive schemata*. A key milestone in Novamente intelligence will occur when BOA+PTL learning is able to learn cognitive schemata of complexity equal to that of Novamente mind agents. At this point, Novamente will be able to improve its own cognitive processes, which should lead to a phase of exponentially increasing intelligence.

Pattern Mining and Concept Creation

Pattern mining is performed on all of the atoms in the system, and new predicates are created encapsulating these patterns. It is also used to create new schemata through the combination of existing ones. Pattern mining can also be restricted to create new nodes representing logical combinations of existing nodes.

Goal-Directed Behavior

Schemata that are expected to achieve current goals are executed, the connection between the schemata and the goals having been found through inference. Furthermore, inference is used to search for schemata that will achieve particular goals in important contexts, including, as mentioned above, abstract cognitive control schemata.

System Maintenance

As time passes, system status and goal satisfaction are reevaluated, new percepts are fed into the system, and inference is used to perform context-dependant time-decay on knowledge. Atoms with the lowest long term importance are deleted from the system. System parameters are adjusted to optimize the system's behavior and prevent its degradation.

Attention Allocation

Novamente, at any given time, will possess a huge number of atoms. In order to cope with restrictions on computational resources, it is critical that the system be able to intelligently focus its attention on the subset of atoms that is most important at the moment.

Attention allocation is done through the application of inference and predicate learning to *activity tables*, which record when the different cognitive processes are applied and to which atoms. The goal is to intelligently update the short and long term importance levels as time passes; the former in order to move from focusing on one set of atoms to focusing on another set, and the latter to assign credit based on the utility of carrying out the cognitive processes on different atoms, and of determining which schemata lead to goal fulfillment in different contexts.

Short and long term importance are updated using the same criterion but on different time scales, based on the distinction between what Novamente is thinking about and what Novamente considers important over time. Importance enters the system via atoms associated with the perception of events in the outside world, and atoms expected to lead to goal fulfillment. Included in the latter category is the meta-level goal of adding valuable knowledge to the system; if thinking about something has been useful in the past, it may be useful to think about it in the future. Importance is differentially spread along links via inference, based on the link type.

An important aspect of credit assignment is dealing with false causality; for example, roosters often crow prior to the sun rising. If the system observes this, and wants to cause the sun to rise, it may well cause a rooster to crow. But once this fails, or if the system holds background knowledge indicating that roosters crowing is not likely to cause the sun to rise, then this will be invoked by inference to discount the strength of the implication between rooster-crowing to sun-rising, so that it will not be strong enough to guide schema execution.

Perception and Action

In Novamente, perception and action are fully integrated with the rest of cognition. The system will initially be provided with modality-appropriate predicates and schemata, such as low-level visual processing and basic movement commands. These will be written in sloppy combinatory logic, so that they can be modified and augmented via BOA and PTL through experiential interactive learning, described later.

Since perceptual processing will occur within the same regime as the rest of the system, focus can iteratively shift between perception and cognition, leading to a balance of bottom-up and top-down constraints in tasks such as object recognition, as described in Gestalt psychology (Kohler 1992, Haikonen 2003). For ambiguous percepts such as the Necker cube (the two-dimensional shadow of a wire-frame cube), this process need not converge.

Emergent Structures

The entire Novamente design is structured in order to give rise implicitly to the advanced representations and behaviors described in the next section. However, there are several processes which are explicitly geared towards their creation. As a form of unsupervised learning, clustering is performed on the atoms in the system with BOA. Inference and pattern mining are performed on the activity table to find complex recurrent patterns of activity, which and then use them to guide future activity, and sometimes create predicates embodying them. These recurrent patterns of activity are called *maps*, and are the primary high-level structures described in the next section.

Emergent Representation and Dynamics

Perhaps the subtlest aspect of the Novamente design is the interaction whereby attention allocation dynamics are used to drive inference and learning, which feed back fluidly to direct attention allocation in turn. This allows the definition of *maps*; sets of Atoms that tend to be activated together, or tend to be activated according to a certain pattern, such as an oscillation or a strange attractor.

Generally speaking, the same types of knowledge are represented by individual atoms, as well as by large maps of atoms (some atoms gain meaning only via their coordinated activity involving other atoms). This is complementary to different tasks; atom level representation is crisp, while map level representation is more flexible and non-brittle. Below, we describe a few of the map types that can emerge in Novamente.

A *static map*, the simplest kind, is a collection of atoms that are strongly interconnected with Hebbian links; such atoms in such a map will clearly tend to be acted on simultaneously, due to the spreading of short term importance. *Schema maps*, or distributed schemata, represent complex behaviors, and consist primarily of schemata. For example, when executing a complex motor movement, a sequence of schemata may be executed, with the precise timing and parameters depending on input from perception and internal state, both of which affect the attention allocation process. A *memory map* consists largely of nodes denoting specific entities, and the relationships between them.

Many of the processes implemented explicitly on the atom level can emerge implicitly on the map level. For example, the collection of links between the individual atoms of two maps can be seen as a higher level structure linking the two maps together, and will manifest many of the same dynamics that occur along single links, in a more flexible, complex, and context-aware fashion.

Focused attention in inference is particularly decisive for map formation; it causes relatively small maps to form, as well as hierarchical maps of a certain nature. If a certain set of nodes has been held within focused attention, meaning that many predicates embodying combinations of them have been constructed, evaluated, and reasoned on, this leads to the construction of Hebbian links between them, forming a map. Focused attention in inference allows the system to minimize the use of independence assumptions, thus improving the accuracy of PTL. This process is quite expensive, and scales exponentially, so focused attention cannot hold too many items within it at one time.

While the maps formed via focused attention will generally be relatively small, their members may be nodes grouping other nodes together, which may themselves be involved in maps. In this way, hierarchical maps may form via clusters of clusters. Since the clusters on each level may be interconnected, this is a manifestation of the dual network structure mentioned earlier.

Experiential Interactive Learning

In practice an intelligent system is not just a thinking machine, it is a control system embedded in a world, using cognition to carry out real-world tasks that it judges important. Not only are perception and action intrinsically critical, they may also serve as a foundation for more abstract abilities (e.g., Boroditsky and Ramscar 2002), including communication, control of abstract cognition, and self-modification.

Experiential interactive learning gives a mind a rich and workable understanding of certain basic concepts that are indispensable for making sense of itself and the world. This understanding is expressed not as a short list of basic facts about the world, but rather a rich network of relationships involving a short list of basic concepts. In Novamente, this corresponds to flexible concept maps centered around a key collection of nodes.

The notion of these basic concepts has a long history; the form that had the most impact on the Novamente design that of Wierzbicka, who attempted the first enumeration of these “semantic universals” (Wierzbicka 1972), which by now number around forty (Wierzbicka 1996). Without going so far as to accept that this list constitutes a fixed and rigid universal ontology, one can posit a loosely defined set of semantic primitives as being foundational to understanding the world, notions such as *I*, *you*, *when*, *because*, *after/before*, etc. All of the primitives are grounded in Novamente’s built-in structures and dynamics; e.g., *because* can be grounded in causality and implication links, *want* in goals nodes, and so forth. Experiential interactive learning allows higher level representations and mechanisms to develop around these impoverished mappings.

Through observing itself interact with an environment, a Novamente system will build up a complex system of internal maps describing its own behavior in various contexts, and the behavior of other systems in the environment, including other minds. This will lead to the formation of a “self-system” potentially including multiple subselves (Rowan 1990), as well as the possibility of goal-directed self-modification, in which a Novamente system modifies its own cognitive algorithms and ultimately its own data structures in order to allow it to better achieve its goals. This sort of experience-driven cognitive self-modification is the crux of Novamente general intelligence.

Human Language Processing

Human language processing presents a uniquely difficult problem for Novamente and other artificial systems. In principle, it is not necessarily solvable by the creation of learning and memory mechanisms that operate with human-level effectiveness. Learning human language without a human embodiment or evolutionary heritage is a much harder problem than learning human language in the possession of such endowments. For communication between different Novamente systems, a special language will be utilized, which is radically different from any human language, lacking a linear ordering.

Providing Novamente with understanding of human language is done through a hybrid approach which combines experiential interactive learning, and a conventional computational-linguistics based architecture implemented within the Novamente framework. In this approach, syntax processing is carried out using the link parsing framework developed by (Sleator and Temperley 1991), and the output of link parsing is mapped into Novamente nodes and links using special schemata called “semantic algorithms.” Semantic disambiguation and reference resolution (Manning and Schuetze 1999) are carried out via PTL inference. At present this Novamente-based NLP framework is capable of dealing with a wide variety of English sentences, and its comprehension is enhanced via an interactive user interface which allows users to view Novamente’s interpretation of their input and rephrase their language or correct Novamente’s interpretation as necessary.

Practical Applications

The Novamente design outline in the preceding sections of this paper is embodied in a C++ implementation that is approximately 50% complete. A number of performance issues, such as effectively swapping atoms between disk and memory, and distributed processing, create additional complications that we have omitted for brevity. PTL and BOA have both been implemented and tested successfully; and much of the natural language framework described above has been completed.

In parallel with the development of Novamente to achieve general intelligence, the system has been utilized more narrowly as an “AI toolkit” in the construction of practical commercial software applications, for example:

Bioinformatics. The Biomind Analyzer, developed by Biomind LLC together with Novamente LLC, is an enterprise system for intelligent analysis of microarray gene expression data. BOA is used to uncover interesting patterns in labeled datasets, and also to learn classification models. PTL is used for the integration of background information from a number of heterogeneous sources of biological knowledge, covering gene and protein function, research papers, gene sequence alignment, protein interactions, and pathways. This allows the Biomind Analyzer to augment the datasets it analyzes with background features corresponding to gene or protein categories, participation in pathways, etc. Inference is then used to create new relations between the genes and the functional categories provided by the background sources, effectively suggesting function assignments to genes with unknown roles.

Human Language and Knowledge Management. The Knowledge Mining Initiative, developed by Object Sciences Corp. together with Novamente LLC, utilizes Novamente's cognitive algorithms in combination with its NLP framework. Knowledge is entered via an interactive interface, which allows users to review and revise the system's understanding of that knowledge. A knowledge base is thus produced, which is augmented by reasoning, and may be queried in English or a special formal language. BOA Pattern Mining is used to spontaneously create queries that are judged interesting.

Acknowledgements

The authors have been partially supported by Object Sciences Corp., Biomind LLC, and Vetta Technologies Ltda. during the course of this research.

References

Baum, E. B. 2004. *What is Thought?* Cambridge, MA: MIT Press.

Boroditsky, L., and Ramscar, M. 2002. The Roles of Body and Mind in Abstract Thought. *Psychological Science* 13(2):185-188.

Chaitin, G. 1987. *Algoritmic Information Theory*. New York, NY: Cambridge University Press.

Curry, H. B., and Feys, R. 1958. *Combinatory Logic*. Amsterdam, Holland: North-Holland.

Goertzel, B. 1993. *The Structure of Intelligence*. New York, NY: Springer-Verlag.

Goertzel, B. 1997. *From Complexity to Creativity*. New York, NY: Plenum Press.

Goertzel, B. 2002. *Creating Internet Intelligence*. New York, NY: Plenum Press.

Goertzel, B., Iklé, M., Freire, I. L., Pressing, J., and Pennachin. C. 2004. *Probabilistic Term Logic*. Forthcoming.

Goertzel, B., Silverman, K., Hartley C., Bugaj, S., and Ross, M. 2000. The Baby Webmind Project. In Proceedings of AISB 00, London: SSAISB.

Haikonen, P. 2003. *The Cognitive Approach to Conscious Machines*. Exeter, UK: Imprint Academic.

Hofstadter, D. 1986. *Metamagical Themas: Questing for Essence of Mind and Pattern*. New York, NY: Bantam Books..

Hutter, M. 2000. A Theory of Universal Artificial Intelligence Based on Algorithmic Complexity. Technical Report, ISDIA-14-00, ISDIA.

Indurkhya, B. 1992. *Metaphor and Cognition: An Interactionist Approach*. New York, NY: Kluwer Academic.

Kohler, W. 1992. *Gestalt Psychology*. New York:, NY: Liveright.

Looks, M., Goertzel, B., and Pennachin, C. 2004. *Learning Computer Programs with the Bayesian Optimization Algorithm*. Forthcoming.

Manning, C. and Schuetze, H. 1999. *Foundations of Statistical Natural Language Processing*. Cambride, MA: MIT Press.

Minsky, M. 1986. *The Society of Mind*. New York, NY: Simon and Schuster.

Ocenasek, J. 2002. Parallel Estimation of Distribution Algorithms. PhD. thesis, Faculty of Information Technology, Brno University of Technology.

Peirce, C. S. 1892. The Law of Mind. Reprinted in Hartshorne, C., Weiss, P., and Burks, A.W. eds. 1980. *The Collected Papers of Charles Sanders Peirce*, Cambridge, MA: Harvard University Press, 6.102-6.163.

Pelikan, M., Goldberg, D. E., and Cantú-Paz, E. 1999. BOA: The Bayesian Optimization Algorithm. Technical Report, IlliGAL Report No. 99003, Illinois Genetic Algorithms Laboratory, University of Illinois at Urbana-Champaign.

Pelikan, M. 2002. Bayesian Optimization Algorithm: From Single Level to Hierarchy. Ph.D. thesis, Department of Computer Science, University of Illinois at Urbana-Champaign.

Rowan, J. 1990. *Subpersonalities*, London: Routledge Press.

Sleator, D., and Temperley, D. 1991. Parsing English with a Link Grammar. Technical Report, CMU-CS-91-196, Dept. of Computer Science, Carnegie Mellon University.

Solomonoff, R. J. 1964. A Formal Theory of Inductive Inference, Part 1 and 2. *Inform. Contr*. 7:1-22, 224-254.

Solomonoff, R. J. 1978. Complexity-based induction systems: Comparisons and Convergence Theorems. *IEEE Trans. Inform. Theory* IT-24:4:422-432.

Sommers, F., and Englebretsen, G. 2000. *An Invitation to Formal Reasoning*. Aldershot, UK: Ashgate Publishing Company.

Wang, P. 1993. Belief Revision in Probability Theory. In Proceedings of the Ninth Conference of Uncertainty in Artificial Intelligence, 519-526. San Mateo, CA: Morgan Kaufmann.

Wang, P. 1995. Non-Axiomatic Reasoning System - Exploring the Essence of Intelligence. Ph.D. thesis, Department of Computer Science, Indiana University.

Wierzbicka, A. 1972. *Semantic Primitives*. Frankfurt: Athenäum.

Wierzbicka, A. 1996. *Semantics, Primes and Universals*. Oxford: Oxford University Press.