CogBot

An Integrative Cognitive Architecture Aimed at Emulating Early Childhood Intelligence in a Humanoid Robot

Ben Goertzel
Itamar Arel
Cassio Pennachin

Project Summary

Might it be possible, using a combination of current technologies, to create a software program enabling a robot to think, learn and converse roughly at the level of a young human child? While achievements like this have been elusive throughout the history of AI, we suspect that due to advances in AI, hardware and cognitive science, the answer is now: Yes, if the cognitive architecture of the program is right, and the combination of technologies is carried out according to the right theoretical principles. The goal of the proposed research is to explore the hypothesis that **the cognitive synergy ensuing from integrating multiple symbolic and subsymbolic learning and memory components in an appropriate cognitive architecture and environment, can yield robust childlike intelligence**.

Toward this end, a three year software R&D project is proposed, aimed at creating a cognitive architecture called **CogBot** enabling a humanoid robot to carry out a set of mentally challenging tasks in a robot lab outfitted as a "virtual preschool." The robotic platform for the project will be the Nao humanoid robot, connected via wifi with a compute cluster running the CogBot software, including standard multiprocessor servers and NVidia Tesla vector processing servers. Human child behavior will be used not as a target to be closely imitated, but rather as a guide to formulating appropriate tasks and metrics for instructing and assessing the robot. The goal is for the robot to learn, reason, create and converse at the qualitative level of a three year old human child.

The core of the effort will be the CogBot integrative cognitive architecture, a fusion of of Itamar Arel's DeSTIN [1] deep learning system for perception and actuation, with Ben Goertzel's and Cassio Pennachin's OpenCog architecture for cognition and language [2]. Collectively, these systems involve sophisticated methods for visual and auditory pattern inference, language comprehension and generation, action planning, commonsense uncertain reasoning, and concept creation; and most of these capabilities have been evaluated in prior applications of narrower scope. In CogBot these methods will be integrated in a novel way, based on the principle of **cognitive synergy**, resulting in an overall cognitive architecture in which practical goals are achieved via multiple learning processes associated with multiple memory types cooperatively manipulating a common network-based knowledge store. While most of the components of CogBot are moderately similar to those used in prior AI systems, our hypothesis is that the proper integration of these components, coupled with their usage in an appropriately stimulating context, will lead to overall system dynamics highly dissimilar to those seen in experiments with prior cognitive architectures.

This research is expected to yield a fundamental advance in the theory and practice of human-level AGI, allied with advances in computer vision and audition, reinforcement learning, embodied language learning, credit assignment, concept creation and experientially grounded probabilistic inference.

The specific aims of the research are to:

1. Create a detailed, integrative software architecture for generally-intelligent humanoid robot control, integrating DeSTIN and OpenCog in a novel manner

2. Implement this architecture within the OpenCog open-source AGI codebase (leveraging existing code wherever possible)

3. Utilize this implementation of the architecture to control a Nao humanoid robot, and interact with this robot in a "preschool" type robot lab environment in such a way as to teach it how to carry out a broad spectrum of behaviors characterizing human early childhood intelligence (including verbal and social interactions with humans, and physical interactions with its environment)

4. Investigate and analyze the degree to which "cognitive synergy" based interactions between system components allows scalable human childlike intelligence

Broader Impacts This project offers the potential to introduce a dramatic advance in artificial general intelligence and in cognitive robotics. It will serve as a platform for further dramatic advances, in the form of ongoing R&D aimed at producing software enabling humanoid robots to achieve humanlike general intelligence beyond the early childhood level. And its successful completion is expected to have broad impact on the AI field, inspiring other researchers to pursue integrated cognitive architectures for intelligent agent control, and in general helping to revive research interest in the original, ambitious goals of the AI field.

Contents

Project Description

1 Introduction

At its inception in the 1950s, AI aimed at producing human level general intelligence in machines. Within a decade or so the difficulty of that goal became evident, and it was scaled back to one of producing systems displaying intelligence within narrow domains. Over the past few years, however, there has been a resurgence of research interest in the original goals of AI [3, 4, 5, 6], based on the assessment that, due to advances in computer hardware, computer science, cognitive psychology, neuroscience and domain-specific AI, we are in a far better position to approach these goals than were the founders of AI.

Cognitive science and neuroscience have taught us a lot about what a cognitive architecture needs to look like to support roughly human-like general intelligence. Computer hardware has advanced to the point where we can build distributed systems containing large amounts of RAM and large numbers of processors, carrying out complex tasks in real time. The AI field has spawned a host of ingenious algorithms and data structures, which have been successfully deployed for a huge variety of purposes.

There is no consensus on why all this progress has not yet yielded AI software systems with human-like general intelligence. Our hypothesis, however, is that the main reason is that

- intelligence depends on the emergence of certain high-level structures and dynamics across a system's whole knowledge base
- we have not discovered any one algorithm or approach capable of yielding the emergence of these structures
- achieving the emergence of these structures within a system formed by integrating a number of different AI algorithms and structures requires careful attention to the manner in which these algorithms and structures are integrated; and so far the integration has not been done in the correct way

The human brain appears to be an integration of an assemblage of diverse structures and dynamics, built using common components and arranged according to a sensible cognitive architecture. However, its algorithms and structures have been honed by evolution to work closely together – they are very tightly inter-adapted, in the same way that the different organs of the body are adapted to work together. Due their close interoperation they give rise to the overall systemic behaviors that characterize human-like general intelligence. We believe that the main missing ingredient in AI so far is **cognitive synergy**: the fitting-together of different intelligent components into an appropriate cognitive architecture, in such a way that the components richly and dynamically support and assist each other, interrelating very closely in a similar manner to the components of the brain or body and thus giving rise to appropriate emergent structures and dynamics. Which leads us to the central hypothesis motivating the proposed research: that **the cognitive synergy ensuing from integrating multiple symbolic and subsymbolic learning and memory components in an appropriate cognitive architecture and environment, can yield robust childlike intelligence.**

The reason this sort of intimate integration has not yet been explored much is that it's difficult on multiple levels, requiring the design of an architecture and its component algorithms with a view toward the structures and dynamics that will arise in the system once it is coupled with an appropriate environment. Typically, the AI algorithms and structures corresponding to different cognitive functions have been developed based on divergent theoretical principles, by disparate communities of researchers, and have been tuned for effective performance on different tasks in different environments. Making such diverse components work together in a truly synergetic and cooperative way is a tall order, yet we believe that this – rather than some particular algorithm, structure or architectural principle – is the "secret sauce" needed to create human-level AGI based on technologies available today. This is the guiding idea underlying the proposed CogBot [1] project.

[1] The term "CogBot" has been used before for a couple different purposes; but none has been sufficiently high-profile for us to judge it inappropriate to use it here. It has been used for an OpenCog/OpenSim interface created by Kino Coursey; and also in the name of Juergen Schmidhuber's CogBotLab.

The proposed work constitutes a multidisciplinary effort, that involves component problems in multiple AI areas, including computer vision and audition, humanoid robot control, computational linguistics, probabilistic reasoning, automatic program learning, assignment of credit and concept creation. The results of this research are expected to yield advances in each of these areas, in addition to the advancement toward human-level artificial general intelligence implicit in the achievement of childlike intelligence in a humanoid robot. However, we stress that we do not need to make general-purpose breakthroughs in all these component problems in order to succeed at our overall goal. For instance, we don't need to solve the general problem of natural language comprehension – but only the problem of natural language comprehension at the level of a young human child, in a preschool-like context. According to the cognitive synergy approach, in the context of integrated behavior, shortcomings in solutions to any component problem may be overcome by strengths in solutions to other component problems.

2 Robot Preschool

One issue that arises when pursuing the "grand goal" of human-level general intelligence is how to measure partial progress. The classic Turing Test of imitating human conversation remains too difficult to usefully motivate immediate-term AI research (see [7, 8] for arguments that it has been counterproductive for the AI field). The same holds true for comparable alternatives like the Robot College Test of creating a robot that can attend a semester of university and obtain passing grades. However, some researchers have suggested intermediary goals, that constitute partial progress toward the grand goal and yet are qualitatively different from the highly specialized problems to which most current AI systems are applied.

In this vein, Sam Adams and his team at IBM have outlined a so-called "Toddler Turing Test," in which one seeks to use AI to control a robot qualitatively displaying similar cognitive behaviors to a young human child (say, a 3 year old) [9]. In fact this sort of idea has a long and venerable history in the AI field – Alan Turing's original 1950 paper on AI [10], where he proposed the "Turing Test", contains the suggestion that

"Instead of trying to produce a programme to simulate the adult mind, why not rather try to produce one which simulates the child's?"

We find this "childlike cognition" based approach promising for many reasons, including its integrative nature: what a young child does involves a combination of perception, actuation, linguistic and pictorial communication, social interaction, conceptual problem solving and creative imagination. Human intelligence evolved in response to the demands of richly interactive environments, and a preschool is specifically designed to be a richly interactive environment with the capability to stimulate diverse mental growth. The richness of the preschool environment implies a need for a robotics based approach (rather than, say, a chatbot or virtual world based approach); and fortunately there now exist commercial humanoid robot platforms such as the Nao (see Figure 1), which makes it possible to do robotics-based AI without doing custom robotic engineering.

Another advantage of focusing on childlike cognition is that child psychologists have created a variety of instruments for measuring child intelligence. In the proposed work, we will evaluate the general intelligence of our childlike robot via combining tests typically used to measure the intelligence of young human children, with additional tests crafted based on cognitive science and the standard preschool curriculum.

2.1 Design for a Robot Preschool

We don't aim to outfit our robot lab as a precise imitation of a human preschool – this would be inappropriate since the Nao robot's body is much less capable than that of a young human child. But we aim to emulate the basic diversity and educational character of a typical human preschool.

The key notion in modern preschool design is the "learning center," an area designed and outfitted with appropriate materials for teaching a specific skill. Learning centers are designed to encourage learning by doing, which greatly facilitates learning processes based on reinforcement, imitation and correction; and also to provide multiple techniques for teaching the same skills, to accommodate different learning styles and prevent overfitting and overspecialization in the learning of new skills.

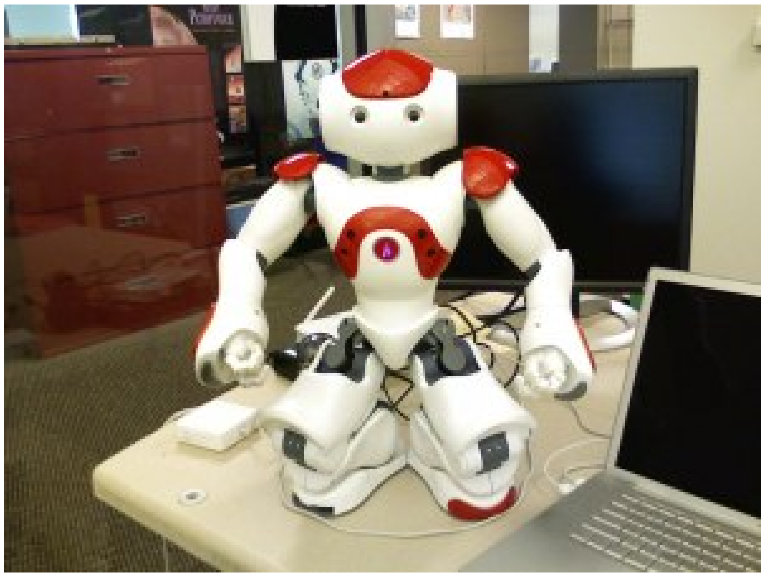


Figure 1: The Nao humanoid robot

Centers are also designed to cross-develop related skills. A "manipulatives center," for example, provides physical objects such as drawing implements, toys and puzzles, to facilitate development of motor manipulation, visual discrimination, and (through sequencing and classification games) basic logical reasoning. A "dramatics center" cross-trains interpersonal and empathetic skills along with bodily-kinesthetic, linguistic, and musical skills. Other centers, such as art, reading, writing, science and math centers are also designed to train not just one area, but to center around a primary intelligence type while also cross-developing related areas. For specific examples of the learning centers associated with particular contemporary preschools, see [11]. In many progressive, student-centered preschools, students are left largely to their own devices to move from one center to another throughout the preschool room. Generally, each center will be staffed by an instructor at some points in the day but not others, providing a variety of learning experiences.

To imitate the general character of a human preschool, we will create several centers in our robot lab. The precise architecture will be adapted via experience but initial centers will likely be:

- **a blocks center**: a table with blocks on it
- **a language center**: a circle of chairs, intended for people to sit around and talk with the robot
- **a manipulatives center**, with a variety of different objects of different shapes and sizes, intended to teach visual and motor skills
- **a ball play center**: where balls are kept in chests and there is space for the robot to kick the balls around
- **a dramatics center** where the robot can observe and enact various movements

3 Conceptual Background

The central hypothesis guiding the proposed work, that **cognitive synergy ensuing from integrating multiple symbolic and subsymbolic learning and memory components in an appropriate cognitive architecture and environment, can yield robust childlike intelligence**, is motivated by a theoretical understanding of general intelligence that stresses the adaptation of intelligence to the environment, and the integrative nature of the environments to which human intelligence is adapted. This theoretical understanding is summarized in Ben Goertzel's book *The Hidden Pattern* [12]; here we will highlight only a handful of the most relevant points.

3.1 What Is General Intelligence?

Psychologists define human general intelligence using IQ tests and related instruments [13]; but to ground AGI approaches that are not based on precise modeling of human cognition, one requires a more fundamental understanding of the nature of general intelligence. Many attempts to characterize general intelligence have been made; Legg and Hutter [14] review over 70! Our preferred characterization of intelligence is: **the capability of a system to choose actions maximizing its goal-achievement, based on its perceptions and memories, and making reasonably efficient use of its computational resources** [15]. A general intelligence is then understood as one that can do this for a variety of complex goals in a variety of complex environments.

It is difficult to say anything nontrivial about general intelligence *in general*. Marcus Hutter [16] has demonstrated that a very simple algorithm called $AIXI^{tl}$ can demonstrate arbitrarily high levels of general intelligence, if given sufficiently immense computational resources. This is interesting because it shows that general intelligence is basically a problem of computational efficiency. The particular structures and dynamics that characterize real-world general intelligences like humans arise because of the need to achieve reasonable levels of intelligence using modest space and time resources.

The theory of mind presented in [12] presents a number of *emergent structures and dynamics* that are hypothesized to characterize pragmatic general intelligence, including such things as system-wide hierarchical and heterarchical knowledge networks, and a dynamic and self-maintaining self-model. Much of the thinking underlying OCP and CogBot has centered on how to make multiple learning components combine to give rise to these emergent structures and dynamics. Appendix A briefly summarizes some of these ideas in the CogBot context.

3.2 Preschool as a View into Human-like General Intelligence

General principles like "complex goals in complex environments" and patternism are not sufficient to specify the nature of *humanlike* general intelligence. Due to the harsh reality of computational resource restrictions, real-world general intelligences are necessarily biased to particular classes of environments. Human intelligence is biased toward the physical, social and linguistic environments in which humanity evolved, and if AI systems are to possess humanlike general intelligence they must to some extent share these biases.

The reason we have chosen to situate our work in a "robot preschool," is that the preschool environment appears to be rich and complex enough to involve the key biases that have shaped human intelligence. It integrates sensorimotor interaction with a rich physical environment and linguistic and social interaction with other intelligent agents, and it provides the flexibility needed to support creative exploration as well as supervised and unsupervised learning. An artificial mind engineered and adapted to do well in preschool, is likely to have the right biases for humanlike intelligence in general.

For a more thorough look into these issues, see Appendix B which gives a comprehensive list of the capabilities characterizing human-like general intelligence, drawn from a recently held "AGI Roadmap Workshop" and also from a classic paper by Alan Newell, and argues for the utility of a preschool environment on this basis.

3.3 Integrative and Synergetic Approaches to Artificial General Intelligence

Comparing the above lists of capabilities and criteria to the array of available AI technologies leads one immediately to consider *integrative* approaches to general intelligence. No single known algorithm or data structure appears easily capable of carrying out all these functions, so if one wants to proceed *now* with creating a general intelligence that is even vaguely humanlike, one must integrate various AI technologies within some sort of unifying architecture.

For this reason and others, an increasing amount of work in the AI community these days is integrative in one sense or another. Estimation of Distribution Algorithms integrate probabilistic reasoning with evolutionary learning [17]. Markov Logic Networks [18] integrate formal logic and probabilistic inference, as does the Probabilistic Logic Networks framework [19] utilized in CogBot, and other work in the "Progic" area such as [20]. Leslie Pack Kaelbling has synthesized low-level robotics methods (particle filtering) with logical inference [21]. Dozens of further examples could be given. The construction of practical robotic systems like the Stanley system that won the DARPA Grand Challenge [22] involve the integration of numerous components based on different principles. These algorithmic and pragmatic innovations provide ample raw materials for the construction of integrative cognitive architectures and are part of the reason why childlike AGI is more approachable now than it was 50 or even 10 years ago.

Further, many of the *cognitive architectures* described in the current AI literature are "integrative" in the sense of combining multiple, qualitatively different, interoperating algorithms. Appendix C gives a high-level overview of existing cognitive architectures, dividing them into *symbolic*, *emergentist* (e.g. neural network) and *hybrid* architectures. The hybrid architectures generally integrate symbolic and neural components, often with multiple subcomponents within each of these broad categories. However, we believe that even these excellent architectures are **not integrative enough**, in the sense that they lack sufficiently rich and nuanced interactions between the learning components associated with different kinds of memory, and hence are unlikely to give rise to the emergent structures and dynamics characterizing general intelligence. The central hypothesis explored in this proposal is that one effective way to approach the goal of qualitatively emulating a young child's intelligence is with an integrative cognitive architecture that combines multiple aspects of intelligence, achieved by diverse structures and algorithms, within a common framework designed specifically to support robust **synergetic interactions** between these aspects.

The simplest way to create an "integrative" AI architecture is to loosely couple multiple components carrying out various functions, in such a way that the different components pass inputs and outputs amongst each other but do not interfere with or modulate each others' internal functioning in real-time. However, the human brain appears to be integrative in a much tighter sense, involving rich real-time dynamical coupling between various components with distinct but related functions. In [23] we have hypothesized that the brain displays a property of **cognitive synergy**, according to which multiple learning processes can not only **dispatch subproblems** to each other, but also **share contextual understanding in real-time**, so that each one can get help from the others in a contextually savvy way. By imbuing AI architectures with cognitive synergy, we hypothesize, one can get past the bottlenecks that have plagued AI in the past. Part of the reasoning here, as elaborated in [24], is that real physical and social environments display a rich dynamic interconnection between their various aspects, so that richly dynamically interconnected integrative AI architectures will be able to achieve goals within them more effectively.

And this brings us back to the patternist perspective on intelligent systems, with its focus on the emergence of hierarchically and heterarchically structured networks of patterns, and pattern-systems modeling self and others. Ultimately the purpose of cognitive synergy in an AGI system is to enable the various AI algorithms and structures composing the system to work together effectively enough to give rise to the right *system-wide emergent structures* characterizing real-world general intelligence. The underlying theory is that intelligence is not reliant on any particular structure or algorithm, but *is* reliant on the emergence of appropriately structured networks of patterns, which can then be used to guide ongoing dynamics of pattern recognition and creation. And the underlying hypothesis is that the emergence of these structures cannot be achieved by a loosely interconnected assemblage of components, no matter how sensible the architecture; it requires a tightly connected, synergetic system.

It is possible to make these theoretical ideas about cognition mathematically rigorous; for instance, Appendix D briefly presents a formal definition of "cognitive synergy" that has been analyzed as part of an effort to prove theorems about the importance of cognitive synergy for giving rise to emergent system properties associated with general intelligence. However, while we have found such formal analyses valuable for clarifying our designs and understanding their qualitative properties, we have concluded that, for the present, the best way to explore our hypotheses about cognitive synergy and human-like general intelligence is empirically – via building and testing systems like CogBot.

3.4 Achieving Human Child Like Intelligence via Cognitive Synergy

At the broadest level, there are four primary challenges in constructing a cognitive synergy based approach to AGI:

1. choosing an **overall cognitive architecture** that possesses adequate richness and flexibility for the task of achieving childlike cognition

2. choosing **appropriate AI algorithms and data structures** to fulfill each of the functions identified in the cognitive architecture (e.g. visual perception, audition, episodic memory, language generation,

analogy,...)

3. ensuring that these algorithms and structures, within the chosen cognitive architecture, are able to cooperate in such a way as to provide appropriate **coordinated, synergetic intelligent behavior** (a critical aspect since childlike cognition is an integrated functional response to the world, rather than a loosely coupled collection of capabilities)
4. embedding one's system in an environment that provides **sufficiently rich stimuli and interactions** to enable the system to use this cooperation to ongoingly create an intelligent internal world-model and self-model

The integration of OCP and DeSTIN to control a Nao in a robot preschool context provides a viable way to address these challenges.

4 Prior Work by the Principal Investigators

We now review OpenCogPrime and DeSTIN, the two major ingredients of the hybrid CogBot architecture.

4.1 OpenCogPrime: A Cognitive Synergy Based Architecture for Controlling Intelligent Virtual Agents

OpenCogPrime (OCP) is a comprehensive architecture for cognition, language, and virtual agent control, created by the PIs Goertzel and Pennachin and their collaborators during the period since 2001 (and building on their work from the 1990s). Conceptually founded on the systems theory of intelligence outlined in [12] and alluded to above, it is currently under development within the open-source OpenCog AI framework (see `http://opencog.org` and [2]). It combines multiple AI paradigms such as uncertain-logic, computational linguistics, evolutionary program learning and connectionist attention allocation in a unified cognitive-science-based architecture. Cognitive processes embodying these different paradigms interoperate together on a common neural-symbolic knowledge store called the Atomspace.

The high-level architecture of OCP, depicted in Figure 2, involves the use of multiple cognitive processes associated with multiple types of memory to enable an intelligent agent to execute the procedures that it believes have the best probability of working toward its goals in its current context. In a robot preschool context, for example, the top-level goals will be simple things such as pleasing the teacher, learning new information and skills, and protecting the robot's body.

Memory Types in OpenCogPrime OCP's memory types are the declarative, procedural, sensory, and episodic memory types that are widely discussed in cognitive neuroscience [25], plus attentional memory for allocating system resources generically, and intentional memory for allocating system resources in a goal-directed way. Table 1 overviews these memory types, giving key references and indicating the corresponding cognitive processes, and also indicating which of the generic patternist cognitive dynamics each cognitive process corresponds to (pattern creation, association, etc.).

In terms of patternist cognitive theory, the multiple types of memory in OCP should be considered as specialized ways of storing particular types of pattern, optimized for spacetime efficiency. The cognitive processes associated with a certain type of memory deal with creating and recognizing patterns of the type for which the memory is specialized. While in principle all the different sorts of pattern could be handled in a unified memory and processing architecture, the sort of specialization used in OCP is necessary in order to achieve acceptable efficient general intelligence using currently available computational resources. And as we have argued above, efficiency is not a side-issue but rather the essence of real-world AGI (since as Hutter has shown, if one casts efficiency aside, arbitrary levels of general intelligence can be achieved via a trivially simple program).

The essence of the OCP design lies in the way the structures and processes associated with each type of memory are designed to work together in a closely coupled way, yielding cooperative intelligence going beyond what could be achieved by an architecture merely containing the same structures and processes in separate "black boxes."

The inter-cognitive-process interactions in OpenCog are designed so that

Memory Type	**Specific Cognitive Processes**	**General Cognitive Functions**
Declarative	Probabilistic Logic Networks (PLN) [19]; conceptual blending [26]	pattern creation
Procedural	MOSES (a novel probabilistic evolutionary program learning algorithm) [27]	pattern creation
Episodic	internal simulation engine [28]	association, pattern creation
Attentional	Economic Attention Networks (ECAN) [29]	association, credit assignment
Intentional	probabilistic goal hierarchy refined by PLN and ECAN, structured according to MicroPsi [30]	credit assignment, pattern creation
Sensory	In CogBot, this will be supplied by the DeSTIN component	association, attention allocation, pattern creation, credit assignment

Table 1: Memory Types and Cognitive Processes in OpenCog Prime. The third column indicates the general cognitive function that each specific cognitive process carries out, according to the patternist theory of cognition.

- conversion between different types of memory is possible, though sometimes computationally costly (e.g. an item of declarative knowledge may with some effort be interpreted procedurally or episodically, etc.)
- when a learning process concerned centrally with one type of memory encounters a situation where it learns very slowly, it can often resolve the issue by converting some of the relevant knowledge into a different type of memory: i.e. **cognitive synergy**

Goal-Oriented Dynamics in OpenCogPrime OCP's dynamics has both goal-oriented and "spontaneous" aspects; here for simplicity's sake we will focus on the goal-oriented ones. The basic goal-oriented dynamic of the OCP system, within which the various types of memory are utilized, is driven by implications known as "cognitive schematics", which take the form

$$Context \wedge Procedure \rightarrow Goal < p >$$

(summarized $C \wedge P \rightarrow G$). Semi-formally, this implication may interpreted to mean: "If the context C appears to hold currently, then if I enact the procedure P, I can expect to achieve the goal G with certainty p." Cognitive synergy means that the learning processes corresponding to the different types of memory actively cooperate in figuring out what procedures will achieve the system's goals in the relevant contexts within its environment.

OCP's cognitive schematic is significantly similar to production rules in classical architectures like SOAR and ACT-R; however, there are significant differences which are important to OCP's functionality. Unlike with classical production rules systems, uncertainty is core to OCP's knowledge representation, and each OCP cognitive schematic is labeled with an uncertain truth value, which is critical to its utilization by OCP's cognitive processes. Also, in OCP, cognitive schematics may be incomplete, missing one or two of the terms, which may then be filled in by various cognitive processes (generally in an uncertain way). A stronger similarity is to MicroPsi's triplets; the differences in this case are more low-level and technical and are discussed in [31].

Finally, the biggest difference between OCPs cognitive schematics and production rules or other similar constructs, is that in OCP this level of knowledge representation is not the only important one. CLARION uses production rules for explicit knowledge representation and then uses a totally separate subsymbolic knowledge store for implicit knowledge. In OCP both explicit and implicit knowledge are stored in the same graph of nodes and links, with

- explicit knowledge stored in probabilistic logic based nodes and links such as cognitive schematics
- implicit knowledge stored in patterns of activity among these same nodes and links, defined via the activity of the "importance" values associated with nodes and links and propagated by the ECAN attention allocation process

The meaning of a cognitive schematic in OCP is hence not entirely encapsulated in its explicit logical form, but resides largely in the activity patterns that ECAN causes its activation or exploration to give rise to. And this fact is important because the synergetic interactions of system components are in large part modulated by ECAN activity. Without the real-time combination of explicit and implicit knowledge in the system's knowledge graph, the synergetic interaction of different cognitive processes would not work so smoothly, and the emergence of effective high-level hierarchical, heterarchical and self structures would be less likely.

4.1.1 Current and Prior Applications of OpenCog

OpenCog has been used for commercial applications in the area of natural language processing and data mining; for instance, see [32] where OpenCog's PLN reasoning and RelEx language processing are combined to do automated biological hypothesis generation based on information gathered from PubMed abstracts. Most relevantly to the present proposal, has also been used to control virtual agents in virtual worlds [28], using an OpenCog variant called the OpenPetBrain (see Figure 3 for a screenshot of an OpenPetBrain-controlled virtual dog; and see `http://novamente.net/example` for some videos of these virtual dogs in action). The CogBot project is a natural extension to humanoid robotics of this prior work in virtual worlds.

While the OpenCog virtual dogs do not display intelligence closely comparable to that of real dogs (or human children), they do demonstrate a variety of interesting and relevant functionalities including

- learning new behaviors based on imitation and reinforcement
- responding to natural language commands and questions, with appropriate actions and natural language replies
- spontaneous exploration of their world, remembering their experiences and using them to bias future learning and linguistic interaction

These are simpler versions of capabilities to be demonstrated by the proposed CogBot system.

Transitioning from Virtual Agents to a Physical Robot With the help of CogBot's other core component – the DeSTIN architecture for perception and action – transitioning OCP's intelligent functionalities from the virtual-agents domain to the physical-robotics domain is expected to be a relatively smooth process.

A reasonable level of capability will be achievable by simply interposing DeSTIN as a perception/action "black box" between OpenCog and a robot. Some preliminary experiments in this direction have already been carried out, connecting the OpenPetBrain to a Nao robot using simpler, less capable software than DeSTIN in the intermediary role (off-the-shelf speech-to-text, text-to-speech and visual object recognition software).

However, to meet the goals of the proposal we must go beyond this approach, and connect robot perception and actuation software with OpenCog in a "white box" manner that allows intimate dynamic feedback between perceptual, motoric, cognitive and linguistic functions. We will achieve this via the creation and real-time utilization of links between the nodes in OpenCog's and DeSTIN's internal networks (a topic to be explored in more depth later in this proposal).

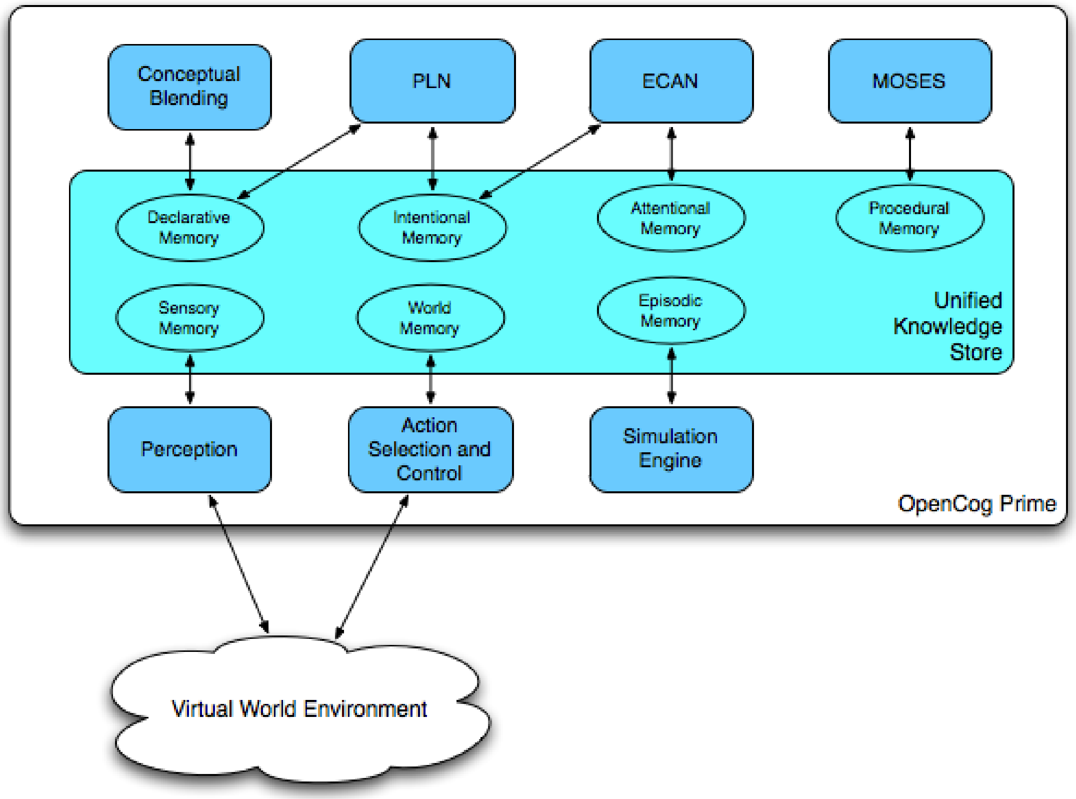


Figure 2: High-Level Architecture of OpenCogPrime. Only the more critical cognitive processes are shown. In practice, all different kinds of memory are interconnected and overlap to some extent, and processes also communicate with each other as needed to fullfil their goals.

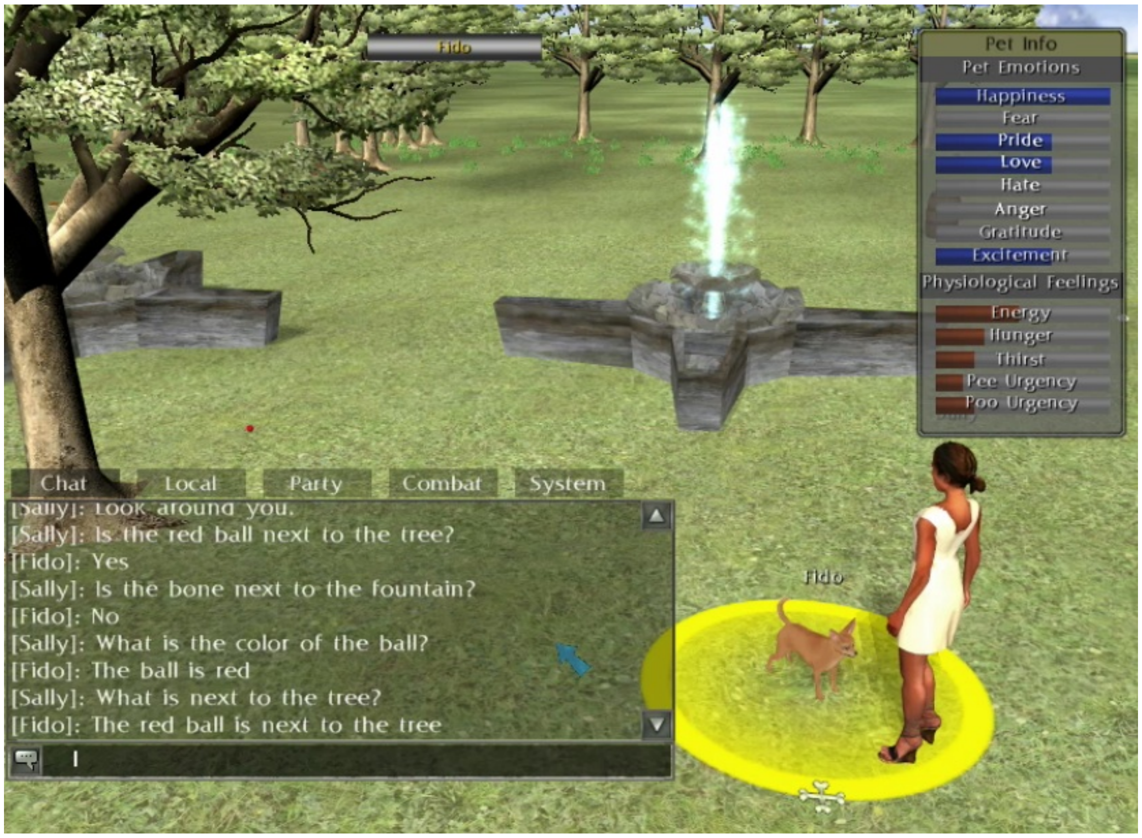
4.1.2 Analysis and Synthesis Processes in OpenCogPrime

The cognitive schematic $Context \wedge Procedure \rightarrow Goal$ leads to a conceptualization of the internal action of an intelligent system as involving two key categories of learning:

- **Analysis**: Estimating the probability p of a posited $C \wedge P \rightarrow G$ relationship
- **Synthesis**: Filling in one or two of the variables in the cognitive schematic, given assumptions regarding the remaining variables, and directed by the goal of maximizing the probability of the cognitive schematic

More specifically, where synthesis is concerned,

- The MOSES probabilistic evolutionary program learning algorithm is applied to find P, given fixed C and G. Internal simulation is also used, for the purpose of creating a simulation embodying C and seeing which P lead to the simulated achievement of G.
  - *Example: A virtual dog learns a procedure P to please its owner (the goal G) in the context C where there is a ball or stick present and the owner is saying "fetch".*
- PLN inference, acting on declarative knowledge, is used for choosing C, given fixed P and G (also incorporating sensory and episodic knowledge as appropriatel). Simulation may also be used for this purpose.



[Sally]: Look around you.
[Sally]: Is the red ball next to the tree?
[Fido]: Yes
[Sally]: Is the bone next to the fountain?
[Fido]: No
[Sally]: What is the color of the ball?
[Fido]: The ball is red
[Sally]: What is next to the tree?
[Fido]: The red ball is next to the tree

Figure 3: Screenshot of OpenCog-controlled virtual dog

  - *Example: A virtual dog wants to achieve the goal G of getting food, and it knows that the procedure P of begging has been successful at this before, so it seeks a context C where begging can be expected to get it food. Probably this will be a context involving a friendly person.*

- PLN-based goal refinement is used to create new subgoals G to sit on the right hand side of instances of the cognitive schematic.
  - *Example: Given that a virtual dog has a goal of finding food, it may learn a subgoal of following other dogs, due to observing that other dogs are often heading toward their food.*

- Concept formation heuristics are used for choosing G and for fueling goal refinement, but especially for choosing C (via providing new candidates for C). They are also used for choosing P, via a process called "predicate schematization" that turns logical predicates (declarative knowledge) into procedures.
  - *Example: At first a virtual dog may have a hard time predicting which other dogs are going to be mean to it. But it may eventually observe common features among a number of mean dogs, and thus form its own concept of "pit bull," without anyone ever teaching it this concept explicitly.*

Where analysis is concerned:

- PLN inference, acting on declarative knowledge, is used for estimating the probability of the implication in the cognitive schematic, given fixed C, P and G. Episodic knowledge is also used this regard, via enabling estimation of the probability via simple similarity matching against past experience. Simulation is also used: multiple simulations may be run, and statistics may be captured therefrom.

  - *Example: To estimate the degree to which asking Bob for food (the procedure P is "asking for food", the context C is "being with Bob") will achieve the goal G of getting food, the virtual dog may study its memory to see what happened on previous occasions where it or other dogs asked Bob for food or other things, and then integrate the evidence from these occasions.*

- Procedural knowledge, mapped into declarative knowledge and then acted on by PLN inference, can be useful for estimating the probability of the implication $C \wedge P \rightarrow G$, in cases where the probability of $C \wedge P_1 \rightarrow G$ is known for some P_1 related to P.
  - *Example: knowledge of the internal similarity between the procedure of asking for food and the procedure of asking for toys, allows the virtual dog to reason that if asking Bob for toys has been successful, maybe asking Bob for food will be successful too.*

- Inference, acting on declarative or sensory knowledge, can be useful for estimating the probability of the implication $C \wedge P \rightarrow G$, in cases where the probability of $C_1 \wedge P \rightarrow G$ is known for some C_1 related to C.
  - *Example: if Bob and Jim have a lot of features in common, and Bob often responds positively when asked for food, then maybe Jim will too.*

- Inference can be used similarly for estimating the probability of the implication $C \wedge P \rightarrow G$, in cases where the probability of $C \wedge P \rightarrow G_1$ is known for some G_1 related to G. Concept creation can be useful indirectly in calculating these probability estimates, via providing new concepts that can be used to make useful inference trails more compact and hence easier to construct.
  - *Example: The dog may reason that because Jack likes to play, and Jack and Jill are both children, maybe Jill likes to play too. It can carry out this reasoning only if its concept creation process has invented the concept of "child" via analysis of observed data.*

In these examples we have focused on cases where two terms in the cognitive schematic are fixed and the third must be filled in; but just as often, the situation is that only one of the terms is fixed. For instance, if we fix G, sometimes the best approach will be to collectively learn C and P. This requires either a procedure learning method that works interactively with a declarative-knowledge-focused concept learning or reasoning method; or a declarative learning method that works interactively with a procedure learning method. That is, it requires the sort of cognitive synergy built into the OCP design.

Appendix E follows up the above general analysis by explaining specifically how cognitive synergy manifests itself in the context of OCP's MOSES algorithm for probabilistic evolutionary procedure learning and PLN framework for probabilistic inference. The synergy between experiential learning and the use of external knowledge resources is also discussed there.

4.2 DeSTIN: A Scalable Architecture for Perception, Action and World-Modeling

OpenCogPrime is designed to handle most of the types of knowledge important for human like intelligence, but in its current form it doesn't deal with low-level sensorimotor knowledge. It could be extended to handle such knowledge in various ways, but for the proposed research we have chosen a different approach: hybridizing OCP with another cognitive architecture that is specifically oriented toward sensorimotor learning, and has already been extensively tested in the perception domain.

The DeSTIN architecture, created by PI Itamar Arel and his colleagues, is an "emergentist" cognitive architecture that addresses the problem of general intelligence using hierarchical spatiotemporal networks designed to enable scalable perception, state inference and reinforcement-learning-guided action in real-world environments. It is founded on similar neuroscience-inspired qualitative information-processing principles to Jeff Hawkins' HTM framework [33], but with a more effective algorithmic underpinning and more complete architecture. The practical work with DeSTIN to date has focused on visual and auditory processing, but it has been developed with the intention of gradually extending it into a complete system for humanoid robot control. Appendix reviews some of the practical results on temporal pattern recognition and face recognition; in the main body of the proposal we focus on the design and its potential for integration into CogBot.

In the CogBot context, the intention is to utilize DeSTIN not as a complete AGI architecture but rather as a collection of modules carrying out perception and actuation oriented processing, hybridizing it with OpenCogPrime which will handle abstract cognition and language. Here we will discuss DeSTIN primarily in the perception context, only briefly mentioning the application to actuation which is conceptually similar.

Crudely speaking,

- From an OCP perspective, DeSTIN's role in CogBot may be viewed as a highly sophisticated intermediary between OCP and a robot's sensors and actuators. OCP can communicate with DeSTIN much as it communicates with a virtual agent, and then DeSTIN can translate between this high-level perception and control language and the lower-level sensation and actuation required by the robot.
- From a DeSTIN perspective, one could view OCP's role in CogBot as augmenting the upper layers of DeSTIN's self-organizing networks, providing more abstract representations and learning algorithms adapted to these representations.

However, neither of these "selfish" perspectives is fully accurate, because as we shall see in the following section, one of the key ideas of the present proposal is to link together DeSTIN and OCP more tightly than this, enabling each one to make use of each others' internal representations – and thus achieving cognitive synergy spanning the sensorimotor and abstract layers of intelligence.

4.2.1 DeSTIN's High-Level Architecture

The DeSTIN architecture comprises three interlined hierarchies (see Figure 4):

- a deep spatiotemporal inference network carrying out perception and world-modeling
- a similarly architected critic network that provides feedback on the inference network's performance
- an action network that controls actuators based on the activity in inference network.

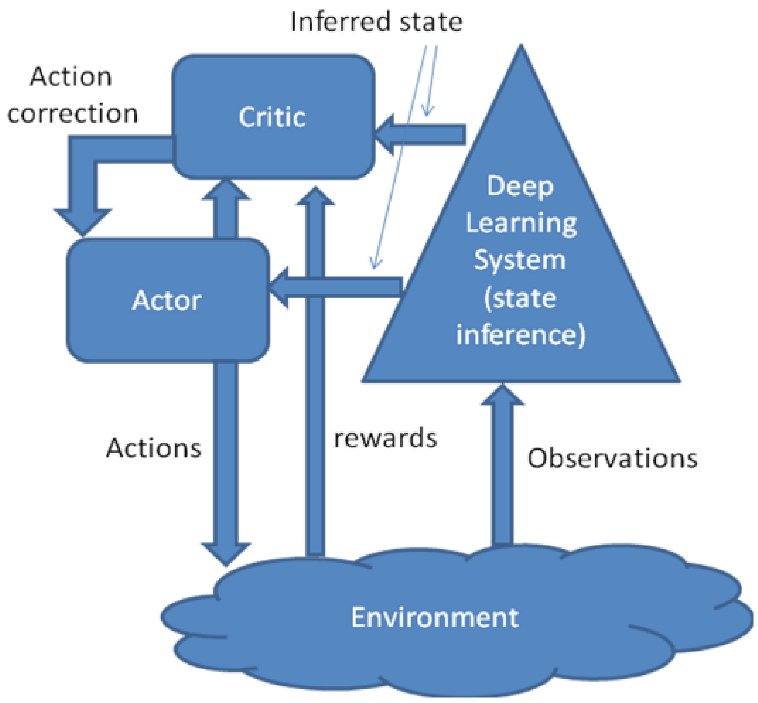


Figure 4: High-level architecture of DeSTIN

The nodes in these networks perform probabilistic pattern recognition according to algorithms to be described below; and the nodes in each of the networks may receive states of nodes in the other networks as inputs, providing rich interconnectivity and synergetic dynamics.

The Perception Hierarchy The process of computational perception processing may be generically broken down into stages such as:

1. **preprocessing** of visual data to reduce noise, enhance contrast, etc.
2. **feature extraction** to extract features at various levels such as lines, edges, ridges, corners, and more complex features related to texture, shape and motion
3. **segmentation and focusing** to determine which portions of an image are most worth attending
4. **high level processing** to carry out tasks like object, pose or movement identification

There are multiple choices to make when implementing these stages. One must choose how much feedback to implement: most computer vision systems implement these stages as a feedforward pipeline, but in the human brain there are numerous feedbacks from the brain regions carrying out the latter stages to the brain regions carrying out the former stages. For instance Poggio's neural net vision models provide excellent simulations of the feedforward connections in part of the brain, but fail to capture the feedback dynamics due to the lack of relevant neurological data to support the level of biological realism to which Poggio's models are committed [34]. And one must choose whether to use different algorithms for the different stages. One common approach is to treat each stage using a separate algorithm (e.g. handling feature extraction and segmentation via domain-specific heuristics, and high-level processing via machine learning algorithms). Another approach is to use a hierarchical architecture in which feature extraction, segmentation and focusing occur mainly on the lower layers and high level processing occurs mainly on the higher layers – but all processes are carried out coherently within the same network. DeSTIN makes the latter choice: a unified perceptual algorithm implemented in a hierarchical feedback/feedforward network, using particular dynamics to be described below.

The Control Hierarchy The process of robot control has been carried out according to a variety of paradigms, but among the most successful has been the hierarchical control paradigm, depicted roughly in Figure 5, and carried to its most sophisticated form in James Albus's 4D/RCS architecture [35]. In this paradigm control is carried out by a hierarchy of layers connected via feedforward and feedback connections, each containing multiple nodes and with higher layer corresponding to coarser spacetime granularity. Each node controls the actions carried out by a particular set of actuators at a particular spacetime granularity. The action hierarchy is linked with the perceptual hierarchy at each level, providing a mechanism via which perception and action can shape each other dynamically. DeSTIN follows this hierarchical paradigm, carrying out action determination via a mechanism similar to its perceptual hierarchy.

4.2.2 Deep versus Shallow Learning for Perceptual Data Processing

The most critical feature of DeSTIN is its uniquely robust approach to modeling the world based on perceptual data. Visual and auditory data is notoriously high-bandwidth, and DeSTIN handles this via an architecture that is designed to make efficient use of massive parallelism (e.g. GPU supercomputers), and via the use of a hierarchical architecture configured to match the hierarchical structure of real-world data. This hierarchical aspects places DeSTIN within the category of "deep learning systems."

We emphasized above the way that humanlike intelligence is heavily adapted to the physical environments in which humans evolved; and one key aspect of sensory data coming from our physical environments is its **hierarchical** structure. Most machine learning and pattern recognition systems are "shallow" in structure, not explicitly incorporating the hierarchical structure of the world in their architecture – and this makes them intrinsically unscalable, unless specialized hacks are incorporated. For instance, in the context of perceptual data processing, the most common hack is to couple a shallow learner with a pre-processing stage, wherein high-dimensional sensory signals are reduced to a lower-dimension feature space that can be understood by the shallow learner. The hierarchical structure of the world is thus crudely captured in the hierarchy of

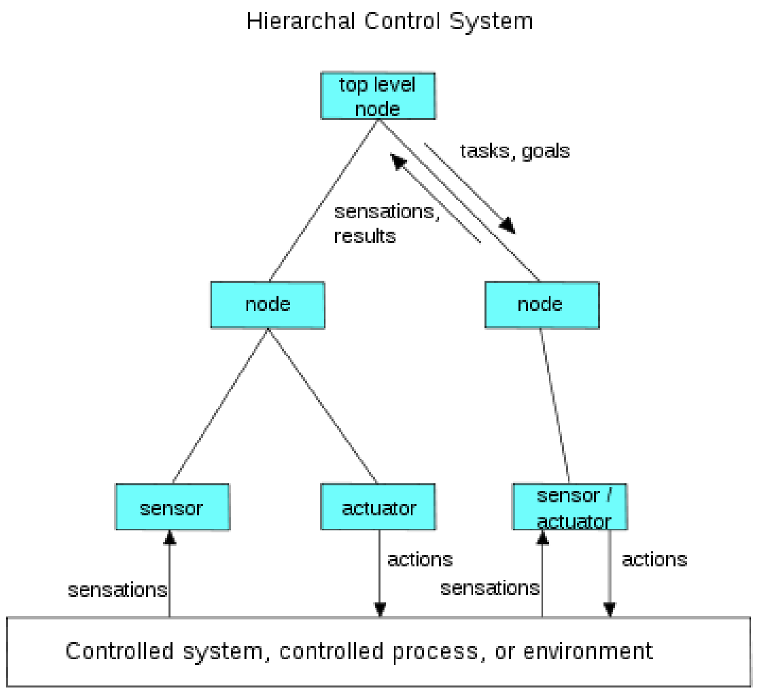


Figure 5: Hierarchal robot control architecture

"preprocessor plus shallow learner." In this sort of approach, much of the intelligence of the system shifts to the feature extraction process, which is often imperfect and always application-domain specific.

Deep machine learning, of which DeSTIN is an example, has emerged as a more promising framework for dealing with complex, high-dimensional real-world data. Deep learning systems possess a hierarchical structure that intrinsically biases them to recognize the hierarchical patterns present in real-world data. Thus, they hierarchically form a feature space that is driven by regularities in the observations, rather than by hand-crafted techniques. They also offer robustness to many of the distortions and transformations that characterize real-world signals, such as noise, displacement, scaling, etc.

DeSTIN is not the only deep learning approach to prove successful on practical problems. Deep belief networks [36] and Convolutional Neural Networks [37] have been demonstrated to successfully address pattern inference in high dimensional data (e.g. images). They owe their success to their underlying paradigm of partitioning large data structures into smaller, more manageable units, and discovering the dependencies that may or may not exist between such units. However, this paradigm has its limitations; for instance, these approaches do not represent temporal information with the same ease as spatial structure. Moreover, some key constraints are imposed on the learning schemes driving these architectures, namely the need for layer-by-layer training, and oftentimes pre-training. Via an architecture and corresponding update equations that support richer nonlinear dynamics, DeSTIN overcomes the limitations of prior deep learning approaches to perception processing, and also extends beyond perception to action and reinforcement learning.

4.2.3 DeSTIN for Perception Processing

The hierarchical architecture of DeSTIN's spatiotemporal inference network comprises an arrangement into multiple layers of "nodes" comprising multiple instantiations of an identical cortical circuit. Each node corresponds to a particular spatiotemporal region, and uses a statistical learning algorithm to characterize the sequences of patterns that are presented to it by nodes in the layer beneath it. More specifically,

- At the very lowest layer of the hierarchy nodes receive as input raw data (e.g. pixels of an image) and continuously construct a belief state that attempts to characterize the sequences of patterns viewed.
- The second layer, and all those above it, receive as input the belief states of nodes at their corresponding

lower layers, and attempt to construct belief states that capture regularities in their inputs.

- each node also receives as input the belief state of the node above it in the hierarchy (which constitutes "contextual" information)

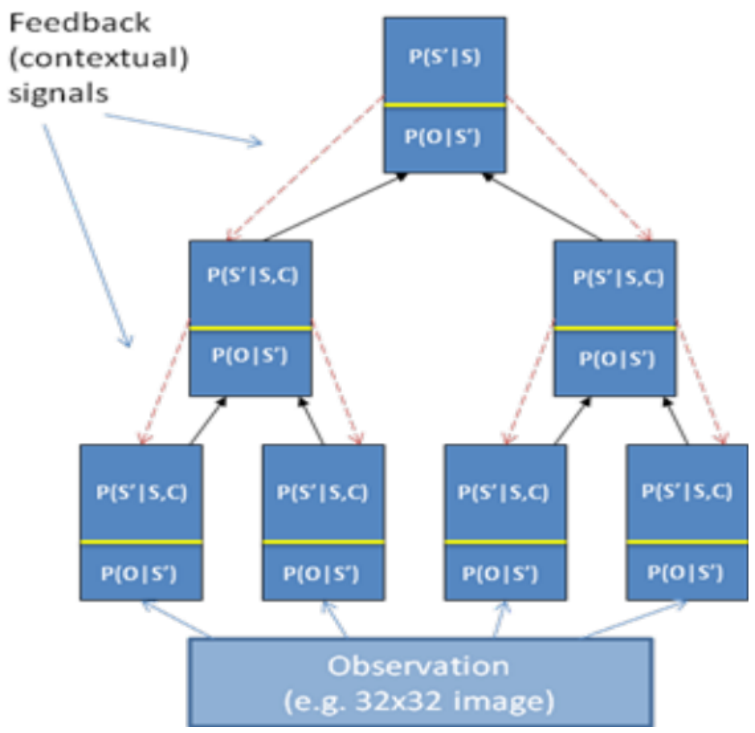


Figure 6: Small-scale instantiation of the DeSTIN perceptual hierarchy. Each box represents a node, which corresponds to a spatiotemporal region (nodes higher in the hierarchy corresponding to larger regions). O denotes the current observation in the region, C is the state of the higher-layer node, and S and S‘ denote state variables pertaining to two subsequent time steps. In each node, a statistical learning algorithm is used to predict subsequent states based on prior states, current observations, and the state of the higher-layer node.

DeSTIN's basic belief update rule, which governs the learning process and is identical for every node in the architecture, is as follows. The belief state is a probability mass function over the sequences of stimuli that the nodes learns to represent. Consequently, each node is allocated a predefined number of state variables each denoting a dynamic pattern, or sequence, that is autonomously learned. We seek to derive an update rule that maps the current observation (o), belief state (b), and the belief state of a higher-layer node (c), to a new (updated) belief state (b'), such that

$$b'(s') = \Pr(s'|o,b,c) = \frac{\Pr(s' \cap o \cap b \cap c)}{\Pr(o \cap b \cap c)}, \tag{1}$$

alternatively expressed as

$$b'(s') = \frac{\Pr(o|s',b,c)\Pr(s'|b,c)\Pr(b,c)}{\Pr(o|b,c)\Pr(b,c)}. \tag{2}$$

Under the assumption that observations depend only on true state, or $\Pr(o|s',b,c) = \Pr(o|s')$, we can further simplify the expression such that

$$b'(s') = \frac{\Pr(o|s')\Pr(s'|b,c)}{\Pr(o|b,c)}, \tag{3}$$

where $\Pr(s'|b,c) = \sum_{s\in S} \Pr(s'|s,c)\, b(s)$, yielding the belief update rule

$$b'(s') = \frac{\Pr(o|s') \sum_{s\in S} \Pr(s'|s,c)\, b(s)}{\sum_{s''\in S} \Pr(o|s'') \sum_{s\in S} \Pr(s''|s,c)\, b(s)}, \tag{4}$$

where S denotes the sequence set (i.e. belief dimension) such that the denominator term is a normalization factor. One interpretation of (4) would be that the static pattern similarity metric, $\Pr(o|s')$, is modulated by a construct that reflects the system dynamics, $\Pr(s'|s,c)$. As such, the belief state inherently captures both spatial an temporal information. In our implementation, the belief state of the parent node, c, is chosen using the selection rule

$$c = \arg\max_{s} b_p(s), \tag{5}$$

where b_p is the belief distribution of the parent node. A closer look at eq. (4) reveals that there are two core constructs to be learned, $\Pr(o|s')$ and $\Pr(s'|s,c)$. We show that the former can be learned via online clustering while the latter is learned based on experience by adjusting of the parameters with each transition from s to s' given c. The result is a robust framework that autonomously (i.e. with no human engineered pre-processing of any type) learns to represent complex data patterns, such as those found in real-life robotics applications.

Based on these equations, the DeSTIN perceptual network serves the critical role of building and maintaining a model of the state of the world. In a vision processing context, for example, it allows for powerful unsupervised classification. If shown a variety of real-world scenes, it will automatically form internal structures corresponding to the various natural categories of objects shown in the scenes, such as trees, chairs, people, etc.; and also the various natural categories of events it sees, such as reaching, pointing, falling. And, as will be discussed below, it can use feedback from DeSTIN's action and critic networks to further shape its internal world-representation based on reinforcement signals.

Benefits of DeSTIN for Perception Processing DeSTIN's perceptual network offers multiple key attributes that render it more powerful than other deep machine learning approaches to sensory data processing:

1. The belief space that is formed across the layers of the perceptual network inherently captures both *spatial and temporal regularities* in the data. Given that many applications require that temporal information be discovered for robust inference, this is a key advantage over existing schemes.

2. Spatiotemporal regularities in the observations are captured in a coherent manner (rather than being represented via two separate mechanisms)

3. All processing is both top-down and bottom-up, and both hierarchical and heterarchical, based on nonlinear feedback connections directing activity and modulating learning in multiple directions through DeSTIN's cortical circuits

4. Support for multi-modal fusing is intrinsic within the framework, yielding a powerful state inference system for real-world, partially-observable settings.

5. Each node is identical, which makes it easy to map the design to massively parallel platforms, such as graphics processing units.

Points 2-4 in the above list describe how DeSTIN's perceptual network displays its own "cognitive synergy," in a way that fits naturally into the overall synergetic dynamics of the CogBot architecture. Using this cognitive synergy, DeSTIN's perceptual network addresses a key aspect of general intelligence: the ability to robustly infer the state of the world, with which the system interacts, in an accurate and timely manner.

4.2.4 DeSTIN for Action and Control

DeSTIN's perceptual network performs unsupervised world-modeling, which is a critical aspect of intelligence but of course is not the whole story. DeSTIN's action network, coupled with the perceptual network, orchestrates actuator commands into complex movements, but also carries out other functions that are more cognitive in nature.

For instance, people learn to distinguish between cups and bowls in part via hearing other people describe some objects as cups and others as bowls. To emulate this kind of learning, DeSTIN's critic network provides positive or negative reinforcement signals based on whether the action network has correctly identified a given object as a cup or a bowl, and this signal then impacts the nodes in the action network. The critic network takes a simple external "degree or success or failure" signal and turns it into multiple reinforcement signals to be fed into the multiple layers of the action network. The result is that the action network self-organizes so as to include an implicit "cup versus bowl" classifier, whose inputs are the outputs of some of the nodes in the higher levels of the perceptual network. This classifier belongs in the action network because it is part of the procedure by which the DeSTIN system carries out the action of identifying an object as a cup or a bowl.

This example illustrates how the learning of complex concepts and procedures is divided fluidly between the perceptual network, which builds a model of the world in an unsupervised way, and the action network, which learns how to respond to the world in a manner that will receive positive reinforcement from the critic network.

5 Integration of DeSTIN and OpenCogPrime

We now describe how the two component systems of CogBot will be integrated. The basic logic of the integration is depicted in 7 but of course the essence of the integration lies in the dynamics, not the structure.

5.1 Pipeline Architectures for Perception and Action

The simplest way to integrate DeSTIN and OCP, as noted above, is simply to have OCP treat DeSTIN in the same way as it treats a virtual agent, i.e.

- OCP sends DeSTIN high-level movement commands like "step forward", "grab the ball at coordinates $(10cm, 30cm, 120cm)$"; DeSTIN then translates these into coordinated sets of servomotor commands
- DeSTIN sends OCP high-level perceptual messages such as "dark blue object with label 1443 at coordinates $(15cm, 45cm, 200cm)$"; OCP then creates semantic nodes in its own memory corresponding to these percepts

This amounts to OCP using DeSTIN's perception hierarchy as a pipeline from the robot, and its action hierarchy as a pipeline to the robot, and its critic hierarchy as a way of giving reinforcement signals to the other two hierarchies.

DeSTIN can initially be taught what servomotor commands to correlate with a high-level movement command via supervised learning: for instance, the robot can be forcibly moved according to a human's interpretation of the command "grab the A at coordinates (X, Y, Z)" while the command is fed into it from OCP, and once this is done for many values of A, X, Y, Z then DeSTIN's hierarchies will learn the command in a generalizable way.

DeSTIN can also be taught to recognize certain classes of objects and events by a similar supervised learning methodology; but the majority of DeSTIN's object and event recognition will be unsupervised, in which case recognized object or events will be identified in DeSTIN via using OCP's pattern-mining process to recognize simple attractor patterns in DeSTIN's upper layers. For instance, a specific cup will correspond to one such pattern; the general concept of "cup" will correspond to another such pattern. These mined patterns will be represented in OCP as PerceptNodes, and it will be OCP's job to supply semantic interpretations to these percepts as appropriate, based on correlating them with linguistic inputs or other percepts or concepts. All that's required of DeSTIN is that it consistently forms the same attractors in its upper layers, for stimuli that represent objects, events or other meaningful sensory regularities.

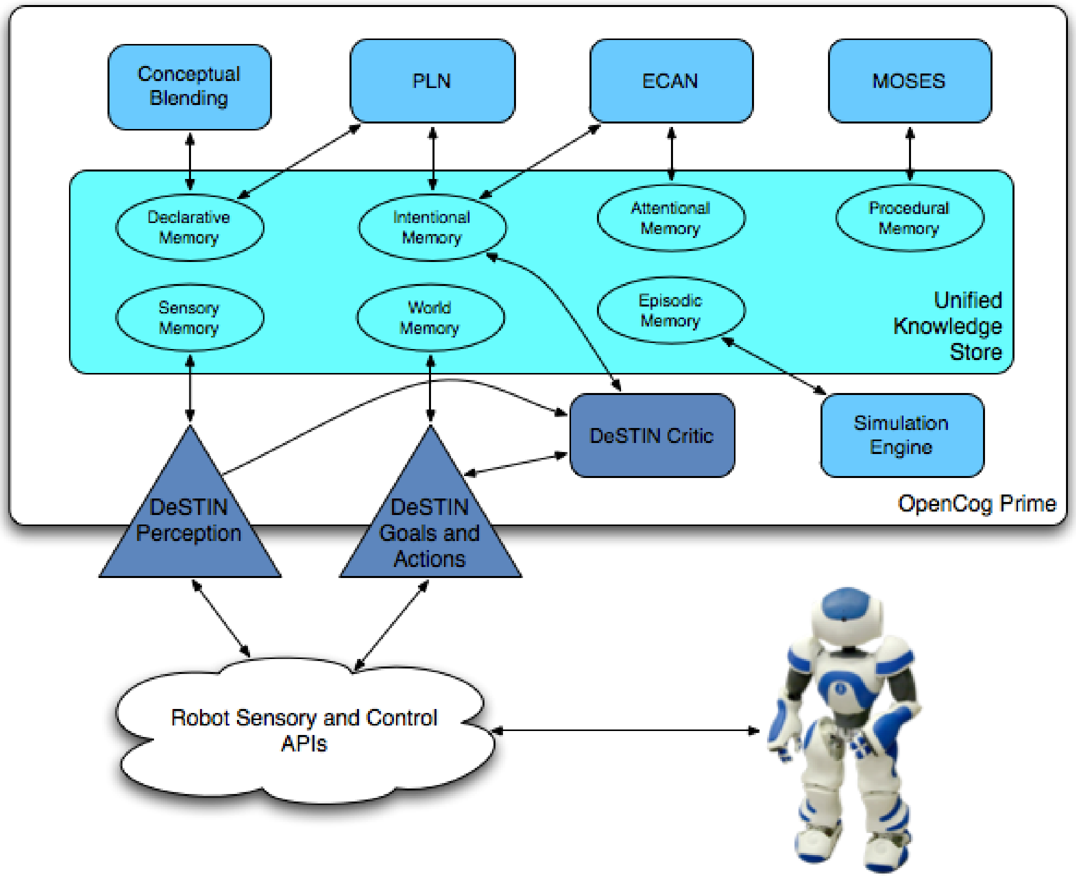


Figure 7: High-Level CogBot Architecture Diagram

Finally, the reinforcement signals that guide DeSTIN's learning via its Critic hierarchy, will be linked to the Goal nodes in OCP's memory network, so that OCP's inferences and judgments can be used to provide DeSTIN's nodes with cognitively savvy feedback on their state inferences and generated actions.

5.2 Deep Integration of Cognition, Perception and Action

The integration described above is not trivial, but it is simplistic in the sense of lacking feedback from cognition into perception and action. This sort of integration will suffice for many purposes, but we don't expect it will suffice for preschool level intelligence, especially not in the areas of visual attention allocation or manipulation of unfamiliar objects. For the latter problem areas, we believe, a tighter integration between perception and cognition will be required, manifested in CogBot via a real-time dynamical relationship between the internal representations of DeSTIN and OCP. With a tighter integration the "cognitive synergy" of the system can extend beyond OCP into DeSTIN, thus binding together the system as a whole.

Structurally this does not require anything new – it merely requires the same links between PerceptNodes and DeSTIN Nodes described in the previous paragraphs. However, the difference is that, to enable feedback from cognition to the sensorimotor layers, activation will be spread from OCP to DeSTIN and back again. Within OCP this activation will be interpreted as the ShortTermImportance values used by the ECAN attention allocation mechanism; within DeSTIN it will be interpreted as neural-net style activation. The passage of activation between the two subsystems will allow each subsystem to focus the attention of the other, thus allowing OCP's long-term memory and reasoning to influence DeSTIN's pattern recognition and action generation activities. This will allow the system to think about how it is manipulating a complex object while it's in the midst of doing it, and to use abstract inference and sensory pattern recognition together to recognize unfamiliar objects or complex events.

With this sort of deep integration the hierarchical and heterarchical pattern networks mentioned in the patternist theory of intelligence will span the DeSTIN and OCP components in a coherent way. And the system's emergent self-model will also span the two components, forming a unified internal image of CogBot as a sensing, acting, thinking, remembering system.

6 Goals: Investigating an Integrative Architecture for Intelligent Humanoid Robotics

The proposed work aims to investigate the hypothesis that **the cognitive synergy ensuing from integrating multiple symbolic and subsymbolic learning and memory components in an appropriate cognitive architecture and environment, can yield robust childlike intelligence**, via pursuing four chief goals, as indicated above:

1. Create a detailed software architecture for generally-intelligent humanoid robot control, integrating DeSTIN and OpenCog
2. Implement this architecture within the OpenCog open-source codebase
3. Utilize this implementation to control a Nao humanoid robot, and interact with this robot in a "preschool" type robot lab environment in such a way as to teach it how to carry out a broad spectrum of behaviors characterizing human early childhood intelligence
4. Investigate and analyze the degree to which "cognitive synergy" based interactions between system components allows scalable human childlike intelligence

Work on the first two of these goals falls naturally into the following high-level categories:

1. Early, simple integration of DeSTIN and OpenCogPrime for initial experiments with the Nao robot.
2. Implementation of aspects of DeSTIN and OpenCogPrime that are covered in the current theoretical documentation of these systems, but are not in the current implementations.
3. Deep integration of DeSTIN and OpenCog via the tuning of both systems' activation-spreading mechanisms to utilize links spanning the two systems' representational graphs.

We now discuss these three categories of work, and then turn to evaluation and analysis.

6.1 Initial Simple Integration Between DeSTIN and OpenCogPrime

The motivation behind this portion of the work is to give the research team an end-to-end integrated system as soon as possible. The following main tasks are required for this goal to be met:

1. Tuning of DeSTIN's action hierarchy for the particular set of actuators provided by the Nao robot.
2. Training of DeSTIN's action hierarchy for the initial repertoire of high-level actions
3. Tuning of DeSTIN's perceptual hierarchy for unsupervised and supervised event and gesture recognition in the robot vision context.
4. Replacement of OpenCogPrime's existing perception and action adapters with code that reads high-level perceptions from DeSTIN and feeds back action commands into DeSTIN.

As described above, this initial integration between the two components will be fairly simplistic, resembling a "pipeline architecture", in which the components feed each other information and don't intervene in each others' internal workings. This baseline system will be useful for experimental work aimed at tuning and improving perception, action, and cognition in isolation. Work on the subsequent goals is expected to bring significant improvements over this simpler baseline.

6.2 Improvements to DeSTIN and OpenCogPrime Implementation

Some components of DeSTIN and OpenCogPrime are incomplete, or need to be improved in order to provide all the functionality required for this project. There are five major tasks that are related to improving or completing specific components of DesTIN or OpenCog:

1. Implementation of goal and action hierarchies within DeSTIN, complementing and interlinking with the currently implemented perception hierarchy.
2. Implementation of concept formation dynamics within OpenCog.
3. Tuning of PLN inference in OpenCogPrime to handle data emerging from robot sensation.
4. Tuning of OpenCogPrime's allowing economic attention allocation to regulate the interactions between procedural, declarative and episodic learning in the robotic context, which more complex than a virtual embodiment context both in richness of information and its lifespan.
5. Upgrading of OpenCogPrime's "internal simulation" module to allow simulation of other agents.

With the execution of these tasks the integrated CogBot architecture as depicted in Figure 7 will be complete. We will then be able to evaluate the key scientific hypothesis underlying the proposed research: that the cognitive synergy ensuing from integrating multiple appropriate components in an appropriate cognitive architecture, allows scalable childlike intelligence.

6.3 Deep Integration of DeSTIN and OpenCog

As noted earlier, in order to achieve the desired level of functionality for tasks like scene interpretation and assembly of complex block structures, it will be necessary to integrate DeSTIN and OpenCogPrime components more tightly than done for the first research goal, by building associative links between the nodes inside DeSTIN and OpenCogPrime respectively. This integration is mathematically straightforward as OpenCogPrime contains a Hebbian link type with similar semantics to the links between DeSTIN nodes; however, the dynamics of the integrated system may be quite complex, and to obtain intelligent functionality some tuning of the DeSTIN parameters and the OpenCogPrime ECAN (economic attention allocation) parameters will likely be required.

7 Tasks and Metrics for Evaluating the Integrated CogBot System

Systematic testing and experimentation will be a significant part of the proposed research. Drawing on existing preschool curricula and child intelligence evaluation instruments, but adapting them to the context of current humanoid robotics technology, we have created a set of 14 tasks representing the list of competency areas described above, and devised quantitative metrics corresponding to each task.

7.1 Measuring Childlike Intelligence

A variety of instruments are used to evaluate the intelligence and capability of young children. Perhaps the most common is the Wechsler Preschool and Primary Scale of Intelligence (WPPSI), an intelligence test designed for children ages 2 years 6 months to 7 years 3 months developed by David Wechsler in 1967 [38]. This instrument evaluates children aged 4 and under on the following kinds of task:

- **Block Design**: While viewing a constructed model or a picture in a Stimulus Book, the child uses one- or two-colour blocks to re-create the design within a specified time limit.
- **Information**: For Picture Items, the child responds to a question by choosing a picture from four response options. For Verbal Items, the child answers questions that address a broad range of general knowledge topics.

- **Receptive Vocabulary**: The child looks at a group of four pictures and points to the one the examiner names aloud.
- **Object Assembly**: The child is presented with the pieces of a puzzle in a standard arrangement and fits the pieces together to form a meaningful whole within 90 seconds.
- **Picture Naming**: The child names pictures that are displayed in a Stimulus Book.

Children aged 5-7 are also evaluated on 9 additional tasks: Matrix Reasoning, Vocabulary, Picture Concepts, Symbol Search, Word Reasoning, Coding, Comprehension, Picture Completion and Similarities.

Using the full list of 14 WPPSI tasks for evaluating CogBot would not be an optimal approach, because many of the 9 tasks that are too advanced for 4 year old children, are actually too easy for CogBot due to aspects of the latter's nonhuman nature. Tasks involving arithmetic and vocabulary memorization, for example, are much easier for CogBot than for a human child. On the other hand, some tasks involving motor abilities are too hard for CogBot, given the Nao's actuators. We have chosen to use the five "age four and under" WPPSI tasks, together with nine additional tasks chosen in the spirit of the Wechsler tasks, the standard preschool curriculum and the capability list given earlier, and well-suited for the particularities of a humanoid robot with a neural-symbolic cognitive architecture.

7.2 Intelligence Testing Tasks

We now describe the specific tasks we will use to test and evaluate CogBot in the course of the proposed project. All tasks are based on the preschool setting described in Section 2.1, and each task has one or more *themes*, or cognitive competencies that are necessary for its successful execution.

1. *Visual Perception.* Teachers point to objects in the manipulatives center, and CogBot should be able to identify the objects. Some objects will be hard to identify at first, either because they'll be partially hidden behind other objects, or because test cases will involve similar-looking objects. In those cases, CogBot can and should approach and manipulate the objects before making an identification attempt. Performance is measured in terms of the number of objects correctly identified, weighted by their difficulty. This task is similar to the WPPSI "Picture Naming" task.
2. *Physical Skills, Imitation Learning.* Teachers manipulate blocks in the blocks center, creating structures of varying degrees of complexity and difficulty (from laying two blocks atop each other to building larger towers and walls, for instance). Teachers can optionally tell CogBot the name of the structure they've built. CogBot should be able to reproduce, to some reasonable degree of accuracy, the structures built by the teachers, based on having observed the final structures but *not* the steps used to build them. Performance is measured in terms of the number of structures replicated, weighted by their complexity, as well as recall; and by the ability to build a structure again if teachers give it a name and then ask for it later. This task is similar to the WPPSI "Block Design" task.
3. *Declarative Memory.* Different teachers will carry different objects (colored balls and boxes) around the manipulatives, ball play and dramatics centers. Each teacher should have a "favorite" object that he/she carries most often. CogBot should be exposed to several events in which these teachers and objects are present. Afterward, when asked to grab one of the "favorite" objects, it should know how to find it if the relevant teacher is present. Performance is measured by the number of person-object associations CogBot learns correctly.
4. *Language-based Communication.* We'll use third-party speech to text (and text to speech) components to allow CogBot to be given instructions in simple English, and to formulate simple English sentences, based on OpenCogPrime's existing Natural Language Processing and Generation pipelines. In the language center, CogBot will be asked a number of simple questions about its experiences, observations, knowledge, and state. It should be able to answer these questions with simple sentences. Performance will be measured by the proportion of semantically correct (even if slightly syntactically incorrect or unusual) answers. This task corresponds to WPPSI's "Information" task.

5. *Reinforcement and Imitation Learning.* In the dramatics center, CogBot should be able to learn to play a number of "games", such as kicking a ball back and forth with another player, playing "tag" or "follow the leader", based on a combination of seeing its teachers play these games and receiving positive/negative feedback from teachers when it attempts to play the games. Performance is measured based on the number of trials it takes for CogBot to fully learn each game.

6. *Logical Reasoning.* CogBot should be able to perform probabilistic reasoning from uncertain premises observed in the preschool. Example tasks include deduction (if Teacher A more often picks up red balls than blue balls, and Teacher A is given a choice between a red box and a blue box, which one is he/she more likely to pick up?), induction (if Teacher A almost always comes to the preschool in weekday mornings, and today is a weekday, is Teacher A coming in the morning?), and abduction (if female teachers more often give the robot red balls than men, and a teacher of unspecified gender gives the robot a red ball, is that teacher a man or a woman?). Performance is measured on the number of correct conclusions drawn by CogBot, weighted by the uncertainty and amount of information available to guide each inference.

7. *Visual Attention Focus.* In the manipulatives center, CogBot is presented with a number of scenes (configurations of objects), and it should be able to identify the key objects in each scene, as well as their relationships, given a clue by a teacher about what's important now. Performance is measured, in each scene, by the proportion of objects and relationships CogBot correctly identifies. This task is similar in spirit to, but more complex than, WPPSI's "Receptive Vocabulary" task.

8. *Puzzle Solving.* In the manipulatives and ball play centers, CogBot is given simple structural tasks, such as: sorting boxes by size; grouping items by color; type, or both; placing every ball inside a box of different color (which can be made more difficult by introducing balls that are larger than some of the boxes); etc. Performance is measured by the number of puzzles correctly solved within a given time frame. This task is similar in spirit to WPPSI's "Object Assembly" task, without placing unrealistic demands on the robot's actuators nor reducing to simplistic imitation.

9. *Tool Use.* Teachers ask CogBot to fetch specific balls in the ball play center. Some of balls are positioned where that the robot can't directly reach, such as in corners behind boxes, but the robot can use sticks or other tools to move the balls to reachable locations. Performance is measured by the number of balls CogBot manages to reach, weighted by the complexity of the "action plan" needed to reach each ball.

10. *Planning.* CogBot has to formulate and execute actions plans involving several actions and steps in order to fullfil requests from a teacher. One example task is as follows: *In the ball play center, CogBot is asked to bring a specific ball to the teacher. The ball is in a corner, and CogBot can't directly reach it, but it knows it can use a tool like a stick to reach it. It has recently been in the manipulatives center, where a stick is kept in a box, so it formulates and executes a plan involving going to the manipulatives center, opening the box, grabbing the stick, going back to the ball play center, pulling the ball out of the corner, dropping the stick, grabbing the ball, and approaching the teacher.* In a case like this, performance will be evaluated on the number of such plans correctly formulated within a time limit. The time limit depends on the complexity of the plan (measured both in terms of depth and breadth, or variety of actions and decisions necessary).

11. *Motivation, Goal Refinement.* CogBot has a set of built-in goals, such as pleasing its teachers and maximizing the amount of new knowledge it absorbs or creates. Given those goals, it should be able to spontaneously learn useful subgoals. For instance, it should observe that it is rewarded by the teachers when demonstrating (through actions or communication) knowledge about its environment, but that the reward is a lot greater when that specific bit of knowledge hasn't been demonstrated to the present teacher before. From that, it should spontaneously demonstrate new knowledge to as many teachers as possible in order to maximize its reward. It should also be able to observe that the manipulatives center and the ball play center can be useful for learning new knowledge even when teachers aren't in those centers, while the language center and the dramatics center are only useful when teachers are present; and based on this understanding, it should spontaneously go to centers that are most useful

for learning in a given moment. Performance can be measured qualitatively by the complexity and ingenuity of CogBot's spontaneous behavior over time, and more quantitatively by instrumenting its code in order to observe its internal goal creation and goal-oriented action decisions.

12. *Self-Awareness.* CogBot should make intelligent decisions based on its abilities. A number of challenges will be posited to measure this meta-learning capability. For instance, when asked to perform an act it can't do (such as reaching an object in an unreachable place), it should say so. When given a chance to get the same reward for an task it can complete reliably, versus a task it can only complete occasionally, it should choose the former. Performance is measured by the number of challenges correctly solved.

13. *Theory of Mind.* CogBot should make correct decisions in situations that require modeling knowledge and biases of its teachers. A number of challenges will be posited to measure this ability. For instance, CogBot may be in the ball play center with two teachers. Teacher A puts a red ball in the red box. Teacher B then leaves and while he's absent, Teacher A moves the red ball to the blue box. Teacher B returns. CogBot is asked where Teacher B will go if asked to fetch the red ball. Performance is measured by the number of challenges correctly solved.

14. *Physical Creativity.* In the blocks center, CogBot should display the ability to create novel, interesting structures from blocks. This ability can be measured by introducing new blocks of different sizes and shapes, giving CogBot the goal of creating new structures, and observing whether it makes useful use of the new blocks, such as building structures that wouldn't be possible before, or building structures that would be possible but are made easier to build (require less blocks, are more stable) by using the new blocks.

Specific test cases and challenges for each task will be conceived during project execution in collaboration with the AGI Roadmap working group.

It is important to notice that, at the end of the proposed project, CogBot is expected to perform *all* the above tasks at anytime, without the need for restarts, resets, or parameter adjustments. While one could develop specialized AI systems for each of the above tasks with satisfactory (and, in some cases, superior to CogBot's) performance, the key aspect of our approach to intelligence testing is the performance, by a single system, of the complete task set, in a flexible and spontaneous way.

7.3 Empirical Study of Cognitive Synergy and Emergent Cognitive Structure

As well as studying the performance of the CogBot-powered robot on an array of tasks, we will also study the internal dynamics of the system, with a view toward understanding the contributions of each component and especially the inter-component interactions. In addition to exploratory data analysis, we will seek to test the hypothesis that cognitive synergy between system components is valuable for achieving intelligent behaviors involving cross-modality and other sorts of behavioral, perceptual and conceptual integration. With this in mind we will vary CogBot's system parameters to modulate the amount of resources associated with each cognitive process (e.g. PLN, MOSES), and the degree to which the processes interact with each other, and will study how these factors impact task performance. We will also seek to study the degree of emergence of hierarchical and heterarchical structures in the system's knowledge network, via calculating appropriate network statistics under various external conditions and internal parameter settings.

8 Project Execution Plan

8.1 Project Phases and Deliverables

The research project will be divided in six phases. We will alternate development-oriented phases, corresponding to the three major goals reported in Section 6, and experimental phases, aimed at achieving satisfactory scores in the tasks described in Section 7.

A high-level timeline is presented in Table 2. Appendix details the specific tasks associated with each of the phases in the table. As the Table and Appendix show, within each phase, work will be subdivided into monthly iterations. These phases will be managed using an agile development methodology, and the subgoals defined for each iteration will be monitored through project management software.

Phase	Execution Months
Initial Integration	1-6
Intelligence Tasks 1-4	7-10
Implementation Improvements	11-18
Intelligence Tasks 5-8	19-22
Deep Integration	23-28
Intelligence Tasks 9-14	29-36

Table 2: High-level Project Execution Timeline

8.2 Staffing and Location

Further details on project staffing, location, etc. are available on request to individuals with a serious interest.

9 Broader Implications: Preschool as a First Step

The research proposed here, if successful, will constitute a substantial advance in the state of the art in AI and cognitive robotics. However, it is not considered as an end goal, but rather as a step along a longer research path which will include even more exciting developments. For robots as for humans, preschool should be considered as the first step to bigger and better things.

Part of the long-term vision underlying the proposed research is that a robot preschool student may serve as the first step along a research path leading to robot elementary school students, and ultimately to a robot college student with adult human level intelligence. Developmental psychology gives us many clues regarding the transition process from preschool level intelligence through multiple stages to mature adult intelligence, and we have fleshed out in detail how these stages are expected to manifest themselves in the context of CogBot's OCP component [39, 40].

But, exciting as this potential developmental pathway is, it is also important to consider the implications of the proposed "robot toddler" from a broader perspective. A robot capable of carrying out preschool activities will arguably constitute a solution to the "common sense knowledge" problem, which according to many analyses has been the primary bottleneck holding back the AI field. It seems likely that the core ideas underlying the CogBot project – as well as much of the software – will be useful in a wide variety of AI projects. OCP has already been used to aid with scientific data analysis and hypothesis discovery; and we are particularly interested in the prospect of hybridizing the commonsense capability of the robot preschool student with the specialized acumen of these science-oriented OCP applications, to yield novel forms of "artificial scientist." But this is only one among very many possibilities: in fact it would be hard to list important AI application areas where software embodying commonsense knowledge and the capability to flexibly acquire and utilize it would *not* be deeply valuable.

Appendices

A CogPrime and Patternist Cognitive Theory

In [12], the view of general intelligence as achieving complex goals in complex environments is pursued in detail, and used as one of the foundations of a comprehensive "patternist" systems theory of intelligence. The notion of complexity of goals and environments is mathematically grounded in algorithmic information theory via a formalization of the "pattern" concept; and a general intelligence is modeled as a system of patterns (emerging from some physical or computational system) that guide activities of pattern recognition and formation. Here we briefly review some of these ideas and their relationship to CogPrime.

A.1 Abstract of Patternist Cognitive Theory

In patternist cognitive theory, the dynamics of a real-world general intelligence are understood to involve both "spontaneous" self-organization and directly goal-oriented behavior; and 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.
- **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.
- **Combinatory 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.

It is also hypothesized that the network of patterns characterizing a generally intelligent system must give rise to a number of 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 image of the overall network of patterns. Similar structures form to model the images of other minds with which the system interacts.

These emergent structures are posited as absolutely critical to real-world general intelligence – more so than any of the lower-level structures or dynamics that cause them to emerge. Once these structures have emerged and are governing the creation of new patterns in the system, including patterns guiding goal-oriented and spontaneous activity, then one has an effective general intelligence.

The CogBot design is not rigorously derived from the patternist theory of mind presented in [12], but it is heavily inspired thereby. The underlying logic via which the different aspects of CogBot are synergetically linked together, is ultimately derived from the patternist theoretical framework. As we review the specifics of CogBot, we will see each of the patternist dynamics and structures mentioned above come into play. And ultimately, the reason *why* we place faith in the central hypothesis of this proposal, that **"cognitive synergy" ensuing from appropriately integrating multiple symbolic and subsymbolic learning and memory components will enables robust childlike general intelligence** is rooted in patternism: according to the theory articulated in [12], this sort of cognitive synergy is what can enable an integrative intelligent system to give rise to the emergent networks of patterns enabling real-world general intelligence.

A.2 Eliciting Emergent Structures of Intelligence in CogPrime

Part of our confidence in the key hypothesis of the proposed research is due to theoretical reasoning leading to the conclusion that the integration of completed versions of OCP and DeSTIN, and the embedding of the integrated system in a preschool context, will lead to the emergence of these structures mentioned above. In more detail, we hypothesize that:

- the emergence of a system-wide **hierarchical structure** will emerge via the impact of DeSTIN's perception/action/critic hierarchy on the OCP knowledge base. OCP's knowledge base itself has no intrinsic propensity for hierarchical structure, but, via its tight coupling with DeSTIN it will achieve a hierarchical structure, thus leading to a CogBot conceptual hierarchy going all the way from lowest-level perceptions and actions up to the most abstract concepts
- the emergence of a system-wide **heterarchical, associative structure** will emerge via the synergetic combination of OCP's PLN inference and ECAN attention allocation modules. PLN builds probabilistic similarity relationship and ECAN builds probabilistic Hebbian co-occurrence relationships, and the alignment of these two kinds of relationships will lead to a thorough and powerful associative network. The tight integration of DeSTIN and OCP will cause this association-spreading to pervade DeSTIN as well, giving DeSTIN a heterarchical associative dynamic to complement its intrinsic hierarchical dynamics and spatiotemporal hierarchical structure.
- the emergence of a **self-model** will occur via PLN inference building a network of relationships that connect knowledge about the self, episodic memories of the self, and sensorimotor reactions to the self (the latter in DeSTIN's hierarchical networks)
- **assignment of credit** spanning all levels of the hierarchy will follow from the coupling of DeSTIN's critic hierarchy with OCP's goal hierarchy (created by inference, concept blending, and other methods)
- combinatory **creation of new patterns** will occur via the combined action of OCP's MOSES, PLN, conceptual blending and pattern recognition heuristics, with DeSTIN's supervised and spontaneous attractor formation. This is the subtlest part of CogBot's dynamics, and the underlying arguments are too complex to present in the context of a proposal such as this; but the key point is that the various algorithms involved are designed specifically so that the structures created by each of them may be effectively modified and improved by the other algorithms in the system, enabling collaborating build-up of potentially useful patterns in declarative, procedural, episodic, attentional and intentional memory.

While our main method of assessing CogBot's success will be via its performance on empirical tasks, we will also endeavor to measure the extent of emergence of the structures and dynamics mentioned above.

B Characterizing General Intelligence

To more fully clarify the sorts of capabilities and biases that we feel are required for human-like general intelligence (in preschool and in general), we cite here two lists in this regard: one resulting from the the AGI Roadmap Workshop held at the University of Knoxville in October 2008 [2], which was organized by two of the Principal Investigators (Goertzel and Arel), and one from a classic paper by Alan Newell.

B.1 Competencies Characterizing Human-like Intelligence

First, consider the following list of "AGI competency areas" that was collaboratively composed at the AGI Roadmap Workshop. Each broad competency area is listed together with a number of specific competencies sub-areas within its scope:

[2]See `http://www.ece.utk.edu/~itamar/AGI_Roadmap.html`; participants included: Sam Adams, IBM Research; Ben Goertzel, Novamente LLC; Itamar Arel, University of Tennessee; Joscha Bach, Institute of Cognitive Science, University of Osnabruck, Germany; Robert Coop, University of Tennessee; Rod Furlan, Singularity Institute; Matthias Scheutz, Indiana University; J. Storrs Hall, Foresight Institute; Alexei Samsonovich, George Mason University; Matt Schlesinger, Southern Illinois University; John Sowa, Vivomind Intelligence, Inc.; Stuart C. Shapiro, University at Buffalo

1. **Perception**: vision, hearing, touch, proprioception, crossmodal
2. **Actuation**: physical skills, navigation, tool use
3. **Memory**: episodic, declarative, behavioral
4. **Learning**: imitation, reinforcement, interactive verbal instruction, written media, experimentation
5. **Reasoning**: deductive, abductive, inductive, causal, physical, associational
6. **Planning**: strategic, tactical, physical, social
7. **Attention**: visual, social, behavioral
8. **Motivation**: subgoal creation, affect-based motivation, control of emotions
9. **Emotion**: expressing emotion, understanding emotion
10. **Self**: self-awareness, self-control, other-awareness
11. **Social**: empathy, appropriate social behavior, social communication, social inference, group play, theory of mind
12. **Communication**: gestural, pictorial, verbal, language acquisition, cross-modal
13. **Quantitative**: counting, grounded arithmetic, comparison, measurement
14. **Building/Creation**: concept formation, verbal invention, physical construction, social group formation

As part of the AGI Roadmap project, specific tasks were created corresponding to each of the sub-areas in the above list. It is worth noting that, of the multiple possible contexts for teaching and evaluating AGI systems discussed during the AGI Roadmap Workshop, only the "robot preschool" context was judged highly appropriate for exploring **all** of these competencies. For the CogBot project we have chosen representative tasks within (almost) each of the high-level competency areas, on which to evaluate our system's performance during the course of the project.

Each competency on this list is something that is done very subtly and thoroughly by adults, and more simply and limitedly by young children. A narrow-AI approach would focus on making a software system capable of doing one or two of these things very effectively in particularly contexts. The AGI-focused approach we propose here involves creating a single system that can do all these things in a primitive way, intermixing the activities flexibly and adapting their particulars based on circumstance.

B.2 Newell's Criteria for a Human Cognitive Architecture

A related perspective is given by Alan Newell's "functional criteria for a human cognitive architecture" [41], which require that a humanlike AGI system should:

1. Behave as an (almost) arbitrary function of the environment
2. Operate in real time
3. Exhibit rational, i.e., effective adaptive behavior
4. Use vast amounts of knowledge about the environment
5. Behave robustly in the face of error, the unexpected, and the unknown
6. Integrate diverse knowledge
7. Use (natural) language
8. Exhibit self-awareness and a sense of self

9. Learn from its environment

10. Acquire capabilities through development

11. Arise through evolution

12. Be realizable within the brain

In our view, Newell's criterion 1 is poorly-formulated, for while universal Turing computing power is easy to come by, any finite AI system must inevitably be heavily adapted to some particular class of environments for straightforward mathematical reasons [16, 15]. On the other hand, his criteria 11 and 12 are not relevant to the CogBot approach as we are not doing biological modeling but rather AGI engineering. Criterion 10 (development) is relevant to CogBot but not to the proposed work, which focuses not on development but on engineering and instruction of particular capabilities; the development is hoped to occur after the present project, to move the system from the preschool level to more advanced capabilities. However, Newell's criteria 2-9 are essential in our view, and we suggest that the robot preschool environment and the CogBot architecture are well-suited to realize them.

Harking back to our characterization of general intelligence as "achieving complex goals in complex environments," we suggest that fulfilling Newell's criteria 2-9, and mastering the scope of competency areas outlined above, are essential for achieving the array of standard preschool goals in the robot preschool environment. And we hypothesize that one effective way to do this is to create an integrative learning system with powerful cognitive synergy between its components.

B.3 Why Not Just Use Simpler Tests?

The above lists of competencies and requirements allow us to answer those who question the need for a complex framework for AGI intelligence testing and teaching, such as a robot preschool. It is tempting to ask: *Can't one evaluate one's AGI system in simpler ways first, establishing its generally cognitive capability more solidly before dealing with such messy tasks and environments?* But this doesn't make sense if the crux of general intelligence lies precisely in the ability to integrate a broad variety of competencies (as outlined in the AGI Roadmap list above) in the context of learning in real-time from a complex environment (as Newell emphasizes).

The AI field has inspired many competitions, each of which tests some particular type or aspect of intelligent behavior. Examples include robot competitions, tournaments of computer chess, poker, backgammon and so forth at computer olympiads, trading-agent competition, language and reasoning competitions like the Pascal Textual Entailment Challenge, and so on. In addition to these, there are many standard domains and problems used in the AI literature that are meant to capture the essential difficulties in a certain class of learning problems: standard datasets for face recognition, text parsing, supervised classification, theorem-proving, question-answering and so forth.

However, the value of these sorts of tests for AGI is predicated on the hypothesis that the degree of success of an AI program at carrying out some domain-specific task, is correlated the the potential of that program for being developed into a robust AGI program with broad intelligence. If humanlike AGI and problem-area-specific "narrow AI" are in fact very different sorts of pursuits requiring very different principles, as we suspect, then these tests are not strongly relevant to the AGI problem.

There are also some standard evaluation paradigms aimed at AI going beyond specific tasks. For instance, there is a literature on "multitask learning" and "transfer learning," where the goal for an AI is to learn one task quicker given another task solved previously [42, 43, 44, 45, 46]. This is one of the capabilities an AI agent will need to simultaneously learn different types of tasks as proposed in the Preschool scenario given here. And there is a literature on "shaping," where the idea is to build up the capability of an AI by training it on progressively more difficult versions of the same tasks [47]. Again, this is one sort of capability an AI will need to possess if it is to move up some type of curriculum, such as a school curriculum.

While we applaud the work done on multitask learning and shaping, we feel that exploring these processes using mathematical abstractions, or in the domain of various machine-learning or robotics test problems, may not adequately address the problem of AGI. The problem is that generalization among tasks, or from simpler to more difficult versions of the same task, is a process whose nature may depend strongly on the overall nature of the set of tasks and task-versions involved. Real-world tasks have a subtlety of interconnectedness and developmental course that is not captured in current mathematical learning frameworks nor standard AI test problems.

To put it mathematically, we suggest that the universe of real-world human tasks has a host of "special statistical properties" that have implications regarding what sorts of AI programs will be most suitable; and that, while exploring and formalizing the nature of these statistical properties is important, an easier and more reliable approach to AGI testing is to create a testing environment that embodies these properties implicitly, via its being an emulation of the cognitively meaningful aspects of the real-world human learning environment.

C Brief Review of Cognitive Architectures

CogBot possesses many unique aspects, both within its components and in the manner in which they are coordinated, but it also builds on a long tradition of work in the area of cognitive architectures. We now briefly review the scope of existing cognitive architectures, indicating where CogBot fits in and how it differs from what has come before.

Duch, in his survey of cognitive architectures [48], divides existing approaches into three paradigms – symbolic, emergentist and hybrid – as broadly indicated in 8. Drawing on his survey and updating slightly, we give here some key examples of each, and then explain why CogBot represents a significantly more effective approach to embodied human-like general intelligence. In our treatment of emergentist architectures, we pay particular attention to *developmental robotics* architectures, which share considerably with the proposed work in terms of aims and methods, but differ via not integrating a symbolic "language and inference" component such as CogBot includes.

In brief, we believe that the hybrid approach is the most pragmatic one given the current state of AI technology, but that the emergentist approach gets something fundamentally right, by focusing on the emergence of complex dynamics and structures from the interactions of simple components. So CogBot is a hybrid architecture which (according to the cognitive synergy principle) binds its components together very tightly dynamically, allowing the emergence of complex dynamics and structures in the integrated system. Most other hybrid architectures are less tightly coupled and hence seem ill-suited to give rise to the needed emergent complexity. The other hybrid architectures that do possess the needed tight coupling, such as MicroPsi [30], are underdeveloped and founded on insufficiently powerful learning algorithms.

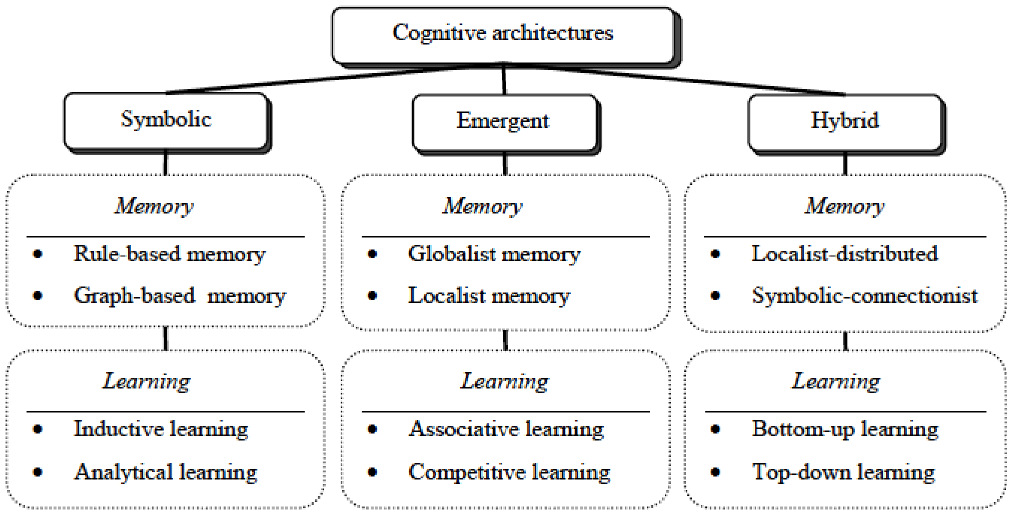


Figure 8: Duch's simplified taxonomy of cognitive architectures. CogBot falls into the "hybrid" category, but differs from other hybrid architectures in its focus on synergetic interactions between components and their potential to give rise to appropriate system-wide emergent structures enabling general intelligence.

C.1 Symbolic Cognitive Architectures

A venerable tradition in AI focuses on the physical symbol system hypothesis [49], which states that minds exist mainly to manipulate symbols that represent aspects of the world or themselves. A physical symbol system has the ability to input, output, store and alter symbolic entities, and to execute appropriate actions in order to reach its goals. Generally, symbolic cognitive architectures focus on "working memory" that draws on long-term memory as needed, and utilize a centralized control over perception, cognition and action. Although in principle such architectures could be arbitrarily capable (since symbolic systems have universal representational and computational power, in theory), in practice symbolic architectures tend to be weak in learning, creativity, procedure learning, and episodic and associative memory. Decades of work in this tradition has not resolved these issues, which has led many researchers to explore other options. A few of the more important symbolic cognitive architectures are:

- **SOAR** [50], a classic example of expert rule-based cognitive architecture designed to model general intelligence. It has recently been extended to handle sensorimotor functions, though in a somewhat cognitively unnatural way; and is not yet strong in areas such as episodic memory, creativity, handling uncertain knowledge, and reinforcement learning.
- **EPIC** [51], a cognitive architecture aimed at capturing human perceptual, cognitive and motor activities through several interconnected processors working in parallel. The system is controlled by production rules for cognitive processor and a set of perceptual (visual, auditory, tactile) and motor processors operating on symbolically coded features rather than raw sensory data. It has been connected to SOAR for problem solving, planning and learning,
- **ICARUS** [52], an integrated cognitive architecture for physical agents, with knowledge specified in the form of reactive skills, each denoting goal-relevant reactions to a class of problems. The architecture includes a number of modules: a perceptual system, a planning system, an execution system, and several memory systems. Concurrent processing is absent, attention allocation is fairly crude, and uncertain knowledge is not thoroughly handled.
- **SNePS** (Semantic Network Processing System) [53] is a logic, frame and network-based knowledge representation, reasoning, and acting system that has undergone over three decades of development. While it has been used for some interesting prototype experiments in language processing and virtual agent control, it has not yet been used for any large-scale or real-world application.
- **Cyc** [54] is an AGI architecture based on predicate logic as a knowledge representation, and using logical reasoning techniques to answer questions and derive new knowledge from old. It has been connected to a natural language engine, and designs have been created for the connection of Cyc with Albus's 4D-RCS [35]. Cyc's most unique aspect is the large database of commonsense knowledge that Cycorp has accumulated (millions of pieces of knowledge, entered by specially trained humans in predicate logic format); part of the philosophy underlying Cyc is that once a sufficient quantity of knowledge is accumulated in the knowledge base, the problem of creating human-level general intelligence will become much less difficult due to the ability to leverage this knowledge.

While these architectures contain many valuable ideas and have yielded some interesting results, we feel they are incapable *on their own* of giving rise to the emergent structures and dynamics required to yield humanlike general intelligence using feasible computational resources. However, we are more sanguine about the possibility of ideas and components from symbolic architectures playing a role in human-level AGI via incorporation in hybrid architectures.

C.2 Emergentist Cognitive Architectures

Another species of cognitive architecture expects abstract symbolic processing to emerge from lower-level "subsymbolic" dynamics, which sometimes (but not always) are designed to simulate neural networks or other aspects of human brain function. These architectures are typically strong at recognizing patterns in high-dimensional data, reinforcement learning and associative memory; but no one has yet shown how to achieve high-level functions such as abstract reasoning or complex language processing using a purely subsymbolic approach. A few of the more important subsymbolic, emergentist cognitive architectures are:

- **DeSTIN** [55, 1], which is part of CogBot, may also be considered as an autonomous AGI architecture, in which case it is emergentist and contains mechanisms to encourage language, high-level reasoning and other abstract aspects of intelligent to emerge from hierarchical pattern recognition and related self-organizing network dynamics. In CogBot DeSTIN is used as part of a hybrid architecture, which greatly reduces the reliance on DeSTIN's emergent properties.
- **Hierarchical Temporal Memory (HTM)** [33] is a hierarchical temporal pattern recognition architecture, presented as both an AI approach and a model of the cortex. So far it has been used exclusively for vision processing and we will discuss its shortcomings later in the context of our treatment of DeSTIN.
- **IBCA** (Integrated Biologically-based Cognitive Architecture) [56] is a large-scale emergent architecture that seeks to model distributed information processing in the brain, especially the posterior and frontal cortex and the hippocampus. It has been used to simulate various human psychological and psycholinguistic behaviors, but hasn't been shown to give rise to higher-level behaviors like reasoning or subgoaling.
- **NOMAD** (Neurally Organized Mobile Adaptive Device) automata [57] are based on Edelman's "Neural Darwinism" model of the brain, and feature large numbers of simulated neurons evolving by natural selection into configurations that carry out sensorimotor and categorization tasks. The emergence of higher-level cognition from this approach seems particularly unlikely. theory

DeSTIN versus HTM In the context of emergent architectures, it is perhaps worth briefly comparing DeSTIN with Hierarchical Temporal Memory (HTM), a deep learning framework introduced by Jeff Hawkins and his colleagues at Numenta in recent years [33]. HTM and DeSTIN share several key conceptual assumptions, including feedback between nodes in adjacent layers. However, HTM is based on Bayesian inference with an artificial temporal model via use of a Markov chain at each node, thereby limiting the performance of the system. Classification information in HTM (due to its very nature) is extracted only from the top layer, which, as stated before, limits performance due to the inherently lossy process involved. Another key difference between the two paradigms is that there is no natural way of capturing temporal information in HTM (and, in fact, it has not been demonstrated to date). Learning is achieved on a layer by layer basis, which is neither biologically-inspired nor scalable. Moreover, there is a clear need to perform strong, modality-dependent feature extraction in HTM (e.g. Gabor filtering) prior to applying the information to HTM restrictions that do not exist in DeSTIN. DeSTIN has been successfully demonstrated to work on high-dimensional images and capture temporal dependencies, while HTM has had limited success in such domains.

Comments on Emergentism Our general perspective on the emergentist approach is that it is philosophically correct but currently pragmatically inadequate. Eventually, *some* emergentist approach could surely succeed at giving rise to humanlike general intelligence – the human brain, after all, is plainly an emergentist system. However, we currently lack understanding of how the brain gives rise to abstract reasoning and complex language, and none of the existing emergentist systems seem remotely capable of giving rise to such phenomena. It seems to us that the creation of a successful emergentist AGI will have to wait for either a detailed understanding of how the brain gives rise to abstract thought, or a much more thorough mathematical understanding of the dynamics of complex self-organizing systems.

The concept of cognitive synergy is more relevant to emergentist than to symbolic architectures. In a complex emergentist architecture with multiple specialized components, much of the emergence is expected to arise via synergy between different richly interacting components. Symbolic systems, at least in the forms currently seen in the literature seem less likely to give rise to cognitive synergy as their dynamics tend to be simpler. And hybrid systems, as we shall see, are somewhat diverse in this regard: some rely heavily on cognitive synergies and others consist of more loosely coupled components.

C.3 Developmental Robotics Architectures

A particular subset of emergentist cognitive architectures are sufficiently important that we consider them separately here: these are *developmental robotics* architectures, focused on controlling robots without significant "hard-wiring" of knowledge or capabilities, allowing robots to learn (and learn how to learn etc.) via their engagement with the world. A significant focus is often placed here on "intrinsic motivation," wherein the robot explores the world guided by internal goals like novelty or curiosity, forming a model of the world as it goes along, based on the modeling requirements implied by its goals. Many of the foundations of this research area were laid by Juergen Schmidhuber's work in the 1990s [58, 59, 60, 61], but now with more powerful computers and robots the area is leading to more impressive practical demonstrations.

We mention here a handful of the important initiatives in this area:

- Juyang Weng's **Dav** [62] and **SAIL** [63] projects involve mobile robots that explore their environments autonomously, and learn to carry out simple tasks by building up their own world-representations through both unsupervised and teacher-driven processing of high-dimensional sensorimotor data. The underlying philosophy is based on human child development [64], the knowledge representations involved are neural network based, and a number of novel learning algorithms are involved, especially in the area of vision processing.
- **FLOWERS** [65], an initiative at the French research institute INRIA, led by Pierre-Yves Oudeyer, is also based on a principle of trying to reconstruct the processes of development of the human child's mind, spontaneously driven by intrinsic motivations. Kaplan [66] has taken this project in a directly closely related to the present one via the creation of a "robot playroom." Experiential language learning has also been a focus of the project [67], driven by innovations in speech understanding.
- Ben Kuipers has pursued an extremely innovative research program which combines qualitative reasoning [68] and reinforcement learning [69] to enable a robot to learn how to act, perceive and model the world. Kuipers' notion of "bootstrap learning" [70] involves allowing the robot to learn almost *everything* about its world, including for instance the structure of 3D space and other things that humans and other animals obtain via their genetic endowments. Compared to Kuipers' approach, CogBot falls in line with most other approaches which provide more "hard-wired" structure, following the analogy to biological organisms that are born with more innate biases.
- **IM-CLEVER**[3], a new European project coordinated by Gianluca Baldassarre and conducted by a large team of researchers at different institutions, which is focused on creating software enabling an iCub [71] humanoid robot to explore the environment and learn to carry out human childlike behaviors based on its own intrinsic motivations. As this project is the closest to our own we will discuss it in more depth below.

Like CogBot, IM-CLEVER is a humanoid robot intelligence architecture guided by intrinsic motivations, and using a hierarchical architectures for reinforcement learning and sensory abstraction. IM-CLEVER's motivational structure is based in part on Schmidhuber's information-theoretic model of curiosity [72]; and CogBot's Psi-based motivational structure utilizes probabilistic measures of novelty, which are mathematically related to Schmidhuber's measures. On the other hand, IM-CLEVER's use of reinforcement learning follows Schmidhuber's earlier work RL for cognitive robotics [73, 74], Barto's work on intrinsically motivated reinforcement learning [75, 76], and Lee's [77, 78] work on developmental reinforcement learning; whereas CogBot's assemblage of learning algorithms is more diverse, including probabilistic logic, concept blending and other symbolic methods (in the OCP component) as well as more conventional reinforcement learning methods (in the DeSTIN component).

In many respects IM-CLEVER bears a moderately strong resemblance the DeSTIN component of our CogBot architecture (although IM-CLEVER has much more focus on biological realism than DeSTIN). Apart from numerous technical differences, the really big distinction between IM-CLEVER and CogBot is that in the latter we are proposing to hybridize a hierarchical-abstraction/reinforcement-learning system (DeSTIN) with a more abstract symbolic cognition engine (OCP) that explicitly handles probabilistic logic and language. IM-CLEVER lacks the aspect of hybridization with a symbolic system, taking more of a

[3]`http://im-clever.noze.it/project/project-description`

pure emergentist strategy. Like DeSTIN considered as a standalone architecture IM-CLEVER does entail a high degree of cognitive synergy, between components dealing with perception, world-modeling, action and motivation. However, because of the "emergentist versus hybrid" difference, the IM-CLEVER and CogBot projects are not just different engineering projects but qualitatively different scientific explorations (each of which may however generate scientific lessons of value to the other). Not surprisingly, the differences between the two architectures are reflected in the evaluation measures proposed for each of them – both IM-CLEVER and CogBot take their cue from child development, but CogBot's metrics focus more on linguistic and cognitive aspects, whereas IM-CLEVER's focus more on sensorimotor coordination [79]. Also, considerable analysis has been undertaken to understand what internal emergent structures are likely to emerge from CogBot once it is completed and interacting with its environment, whereas the theoretical work related to IM-CLEVER has had a different focus.

In all, while we largely agree with the philosophy underlying developmental robotics, our intuition is that the learning and representational mechanisms underlying the current systems in this area are probably not powerful enough to lead to human child level intelligence. We expect that these systems will develop interesting behaviors but fall short of robust preschool level competency, especially in areas like language and reasoning where symbolic systems have typically proved more effective. This intuition is what impels us to pursue a hybrid approach.

C.4 Hybrid Cognitive Architectures

In response to the complementary strengths and weaknesses of the symbolic and emergentist approaches, in recent years a number of researchers have turned to integrative, hybrid architectures, which combine subsystems operating according to the two different paradigms. The combination may be done in many different ways, e.g. connection of a large symbolic subsystem with a large subsymbolic system, or the creation of a population of small agents each of which is both symbolic and subsymbolic in nature.

Nils Nilsson expressed the motivation for hybrid AGI systems very clearly in his article at the AI-50 conference (which celebrated the 50'th anniversary of the AI field) [80]. While affirming the value of the Physical Symbol System Hypothesis that underlies symbolic AI, he argues that "the PSSH explicitly assumes that, whenever necessary, symbols will be grounded in objects in the environment through the perceptual and effector capabilities of a physical symbol system." Thus, he continues,

"I grant the need for non-symbolic processes in some intelligent systems, but I think they supplement rather than replace symbol systems. I know of no examples of reasoning, understanding language, or generating complex plans that are best understood as being performed by systems using exclusively non-symbolic processes....

AI systems that achieve human-level intelligence will involve a combination of symbolic and non-symbolic processing."

A few of the more important hybrid cognitive architectures are:

- **ACT-R** [81] is more on the symbolic side, but incorporates connectionist-style activation spreading in a significant role; and there is an experimental thoroughly connectionist implementation to complement the primary mainly-symbolic implementation. Its combination of SOAR-style "production rules" with large-scale connectionist dynamics allows it to simulate a variety of human psychological phenomena, but abstract reasoning, creativity and transfer learning are still missing. An interesting article summarizing the strengths and shortcomings of ACT-R appeared recently in BBS [82].

- **CLARION** [83] is a hybrid architecture that combines a symbolic component for reasoning on "explicit knowledge" with a connectionist component for managing "implicit knowledge." Learning of implicit knowledge may be done via neural net, reinforcement learning, or other methods. The integration of symbolic and subsymbolic methods is powerful, but a great deal is still missing such as episodic knowledge and learning and creativity. Learning in the symbolic and subsymbolic portions is carried out separately rather than dynamically coupled, minimizing "cognitive synergy" effects.

- **DUAL** [84] is the most impressive system to come out of Marvin Minsky's "Society of Mind" paradigm. It features a population of agents, each of which combines symbolic and connectionist representation, self-organizing to collectively carry out tasks such as perception, analogy and associative memory. The approach seems innovative and promising, but it is unclear how the approach will scale to high-dimensional data or complex reasoning problems due to the lack of a more structured high-level cognitive architecture.

- **LIDA** [85] is a comprehensive cognitive architecture heavily based on Bernard Baars' "Global Workspace Theory". It articulates a "cognitive cycle" integrating various forms of memory and intelligent processing in a single processing loop. The architecture ties in well with both neuroscience and cognitive psychology, but it deals most thoroughly with "lower level" aspects of intelligence, handling more advanced aspects like language and reasoning only somewhat sketchily. There is a clear mapping between LIDA structures and processes and corresponding structures and processing in OCP; so that it's only a mild stretch to view CogBot as an instantiation of the general LIDA approach that extends further both in the lower level (to enable robot action and sensation via DeSTIN) and the higher level (to enable advanced language and reasoning via OCP mechanisms that have no direct LIDA analogues).
- **MicroPsi** [30] is an integrative architecture based on Dietrich Dorner's Psi model of motivation, emotion and intelligence. It has been tested on some practical control applications, and also on simulating artificial agents in a simple virtual world. MicroPsi's comprehensiveness and basis in neuroscience and psychology are impressive, but in the current version of MicroPsi, learning and reasoning are carried out by algorithms that seem unlikely to scale. OCP incorporates the Psi model for motivation and emotion, so that MicroPsi and CogBot may be considered very closely related systems. But similar to LIDA, MicroPsi currently focuses on the "lower level" aspects of intelligence, not yet directly handling advanced processes like language and abstract reasoning.
- **PolyScheme** [86] integrates multiple methods of representation, reasoning and inference schemes for general problem solving. Each Polyscheme specialist models a different aspect of the world using specific representation and inference techniques, interacting with other specialists and learning from them. Polyscheme has been used to model infant reasoning including object identity, events, causality, spatial relations. The integration of reasoning methods is powerful, but the overall cognitive architecture is simplistic compared to other systems and seems focused more on problem-solving than on the broader problem of intelligent agent control.
- **Shruti** [87] is a fascinating biologically-inspired model of human reflexive inference, represents in connectionist architecture relations, types, entities and causal rules using focal-clusters. However, much like Hofstadter's earlier Copycat architecture [88], Shruti seems more interesting as a prototype exploration of ideas than as a practical AGI system; at least, after a significant time of development it has not proved significant effective in any applications
- **OpenCog** [2] is discussed extensively in this proposal, and is intended to handle all aspects of human-like general intelligence except low-level perception and action, for which it must be hybridized with some other architecture such as DeSTIN

As our own CogBot approach is a hybrid architecture, it will come as no surprise that we believe several of the existing hybrid architectures are fundamentally going in the right direction. However, nearly all the existing hybrid architectures have severe shortcomings which we feel will prevent them from achieving robust humanlike AGI.

Many of the hybrid architectures are in essence "multiple, disparate algorithms carrying out separate functions, encapsulated in black boxes and communicating results with each other." For instance, PolyScheme, ACT-R and CLARION all display this "modularity" property to a significant extent. These architectures lack the rich, real-time interaction between the *internal dynamics* of various memory and learning processes that we believe is critical to achieving humanlike general intelligence using realistic computational resources. On the other hand, those architectures that feature richer integration – such as DUAL, Shruti, LIDA and MicroPsi – have the flaw of relying (at least in their current versions) on overly simplistic learning algorithms, which drastically limits their scalability.

It does seem plausible to us that some of these hybrid architectures could be dramatically extended or modified so as to produce humanlike general intelligence. For instance, one could replace LIDA's learning algorithms with others that interrelate with each other in a nuanced synergetic way; or one could replace MicroPsi's simple learning and reasoning methods with much more powerful and scalable ones acting on the same data structures. However, making these changes would dramatically alter the cognitive architectures in question on multiple levels, and after analyzing the matter in detail we have judged it considerably more feasible to build CogBot from OCP and DeSTIN, as we propose.

C.4.1 Synergetic Hybrid Cognitive Architectures: OpenCogPrime and CogBot

Although CogBot is not an "emergentist" architecture per se, it is founded on a theory of intelligence which holds that emergence is key to general intelligence. What differentiates it from other hybrid architectures is its attempt to achieve emergence via tight integration of multiple, differently-implemented, highly robust and scalable learning processes and memory structures. We hypothesize that this sort of integration, rather than any particular algorithm or data structure, is the "secret sauce" needed to achieve human childlike AGI using current technologies. We hypothesize that CogBot's synergetic dynamics have the potential to give rise to the critical emergent structures of intelligence.

D A Formalization of Cognitive Synergy

The notion of cognitive synergy is fundamentally conceptual rather than formal; but it can also be grounded mathematically in a way that is useful for understanding the interrelations between the particular algorithms in CogBot. In [31], the different types of memory critical for human-like intelligence (e.g. declarative, episodic, procedural,...) are formalized using the language of category theory, and then probability theory is used to define the **learning capability** $\gamma(L_1)$ of an optimization process defined over a memory category K_1 (relative to a particular set of goals and contexts), and the **mutual learning capability** $\gamma(L_1, L_2)$ of a pair of optimization processes operating over two memory categories K_1, K_2 between which a morphism exists.

The *learning synergy* of the pair (L_1, L_2) is then defined as

$$syn(L_1, L_2) \equiv \frac{\gamma(L_1, L_2)}{\gamma(L_1)}$$

and the *mutual synergy* as

$$msyn(L_1, L_2) \equiv min(syn(L_1, L_2), syn(L_2, L_1))$$

More broadly, if we have a set of three learning processes L_1, L_2 and L_3 associated with three knowledge categories K_1, K_2 and K_3, and each pair of these learning processes demonstrates significant mutual synergy, then we may say that the process-triple demonstrates *cognitive synergy*. In general, the **pairwise cognitive synergy** of a set of n goal driven processes acting on different categories may be quantified as a $p'th$ power average of the pairwise mutual synergies.

Given this formalization, the hypothesis that cognitive synergy in an appropriate integrative AGI system gives rise to emergent structures such as hierarchical, heterarchical and self networks, may be explored on the level of pure mathematics. However, there are significant challenges here, because the outcome of such mathematical investigations depends heavily on one's formal assumptions about the *environment* the AGI system is operating in, and properly formalizing the important properties of real-world environments is a difficult problem.

E Specific Examples of Cognitive Synergy in OpenCogPrime

Here we give some specific examples of cognitive synergy as manifested in the OCP architecture as currently implemented.

E.1 Synergy Between Declarative and Procedural Knowledge

As an example of how cognitive synergy arises within the CogBot architecture, consider the MOSES [27] and PLN [19] learning algorithms, which deal with OpenCog's procedural and declarative knowledge respectively.

MOSES seeks to evolve fitter and fitter program trees (i.e. procedural knowledge items). Sometimes it may choose to transform a program tree P_1 into declarative form, ask PLN to do reasoning on this declarative version, and then take PLN's conclusion and turn it back into a hypothesis that some other program tree P_2 will be fit, or a hypothesis about P_1's fitness. If this sort of strategy is frequently valuable (relative to the relevant class of goals and contexts), then this would imply that PLN and MOSES have high mutual synergy, i.e. $syn(MOSES, PLN)$ is high.

On the other hand, the reverse strategy is also possible: if PLN is trying to reason about the outcome of a certain process, it may choose to turn that process into a procedure and ask MOSES to estimate the fitness of the process according to a certain fitness function ... and it may then take MOSES's conclusion and turn it into declarative knowledge about the process, to be used in ongoing reasoning. If this sort of strategy proves valuable for PLN (relative to a particular class of goals and contexts), this implies that $syn(PLN, MOSES)$ is high.

If both of these strategies are effective then the (symmetric) mutual synergy $msyn(MOSES, PLN)$ is high relative to the goals and contexts in question. Mathematically demonstrating the existence of such synergies in OCP is a subject of current theoretical work, but we have preliminary evidence for their existence from our experimental work to date.

We now present more algorithmic detail regarding the operation and synergetic interaction of MOSES and PLN.

E.1.1 Cognitive Synergy in MOSES

MOSES, OCP's primary algorithm for learning procedural knowledge, has been tested on a variety of application problems including standard program induction test problems, virtual agent control, biological data analysis and text classification [27]. It represents procedures internally as program trees. Each node in a MOSES program tree is supplied with a "knob," comprising a set of values that may potentially be chosen to replace the data item or operator at that node. So for instance a node containing the number 7 may be supplied with a knob that can take on any integer value. A node containing a while loop may be supplied with a knob that can take on various possible control flow operators including conditionals or the identity. A node containing a procedure representing a particular robot movement, may be supplied with a knob that can take on values corresponding to multiple possible movements. Following a metaphor suggested by Douglas Hofstadter [89], MOSES learning covers both "knob twiddling" (setting the values of knobs) and "knob creation."

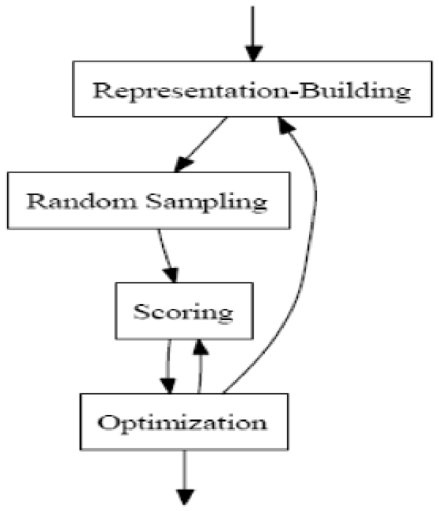


Figure 9: High-Level Control Flow of MOSES Algorithm

MOSES is invoked within OCP in a number of ways, but most commonly for finding a procedure P satisfying a probabilistic implication $C\&P \rightarrow G$ as described above, where C is an observed context and G is a system goal. In this case the probability value of the implication provides the "scoring function" that MOSES uses to assess the quality of candidate procedures.

For example, suppose an OCP-controlled robot is trying to learn to play the game of "tag." Then its context C is that others are trying to play a game they call "tag" with it; and we may assume its goals are to please them and itself, and that it has figured out that in order to achieve this goal it should learn some procedure to follow when interacting with others who have said they are playing "tag." In this case a potential tag-playing procedure might contain nodes for physical actions like $step\_forward(speed\ s)$, as well as control flow nodes containing operators like $ifelse$ (for instance, there would probably be a conditional telling the robot to do something different depending on whether someone seems to be chasing it). Each of these program tree nodes would have an appropriate knob assigned to it. And the scoring function would evaluate a procedure P in terms of how successfully the robot played tag when controlling its behaviors according to P (noting that it may also be using other control procedures concurrently with P). It's worth noting here that evaluating the scoring function in this case involves some inference already, because in order to tell if it is playing tag successfully, in a real-world context, it must watch and understand the behavior of the other players.

MOSES follows the high-level control flow depicted in Figure 9, which corresponds to the following process for evolving a metapopulation of "demes" of programs (each deme being a set of relatively similar programs, forming a sort of island in program space):

1. Construct an initial set of knobs based on some prior (e.g., based on an empty program; or more interestingly, using prior knowledge **supplied by PLN inference** based on the system's memory) and use it to generate an initial random sampling of programs. Add this deme to the metapopulation.
2. Select a deme from the metapopulation and update its sample, as follows:
   (a) Select some promising programs from the deme's existing sample to use for modeling, according to the scoring function.
   (b) Considering the promising programs as collections of knob settings, generate new collections of knob settings by applying some (competent) optimization algorithm. For best performance on difficult problems, it is important to use an optimization algorithm that makes use of the system's memory in its choices, **consulting PLN inference** to help estimate which collections of knob settings will work best.
   (c) Convert the new collections of knob settings into their corresponding programs, reduce the programs to normal form, evaluate their scores, and integrate them into the deme's sample, replacing less promising programs. In the case that scoring is expensive, score evaluation may be preceded by score estimation, which may use **PLN inference**, enaction of procedures in an **internal simulation environment**, and/or similarity matching against **episodic memory**.
3. For each new program that meet the criterion for creating a new deme, if any:
   (a) Construct a new set of knobs (a process called "representation-building") to define a region centered around the program (the deme's *exemplar)*, and use it to generate a *new* random sampling of programs, producing a new deme.
   (b) Integrate the new deme into the metapopulation, possibly displacing less promising demes.
4. Repeat from step 2.

Each part of the MOSES algorithm plays its role; if any one part is removed the performance suffers significantly [27]. However, the main point we want to highlight here is the role played by synergetic interactions between MOSES and other cognitive components such as PLN, simulation and episodic memory, as indicated in **boldface** in the above pseudocode. MOSES is a powerful procedure learning algorithm, but used on its own it runs into scalability problems like any other such algorithm; the reason we feel it has potential to play a major role in a human-level AI system is its capacity for productive interoperation with other cognitive components.

Continuing the "tag" example, the power of MOSES's integration with other cognitive processes would come into play if, before learning to play tag, the robot has already played simpler games involving chasing. If the robot already has experience chasing and being chased by other agents, then its episodic and declarative memory will contain knowledge about how to pursue and avoid other agents in the context of running around an environment full of objects, and this knowledge will be deployable within the appropriate parts of MOSES's Steps 1 and 2. Cross-process and cross-memory-type integration make it tractable for MOSES to act as a "transfer learning" algorithm, not just a task-specific machine-learning algorithm.

E.1.2 Cognitive Synergy in PLN

While MOSES handles much of OCP's procedural learning, and OpenCog's internal simulation engine handles most episodic knowledge, OCP's primary tool for handling declarative knowledge is an uncertain inference framework called Probabilistic Logic Networks (PLN). The complexities of PLN are the topic of a lengthy technical monograph [19], and here we will eschew most details and focus mainly on pointing out how PLN seeks to achieve efficient inference control via integration with other cognitive processes.

As a logic, PLN is broadly integrative: it combines certain term logic rules with more standard predicate logic rules, and utilizes both fuzzy truth values and a variant of imprecise probabilities called *indefinite probabilities*. PLN mathematics tells how these uncertain truth values propagate through its logic rules, so that uncertain premises give rise to conclusions with reasonably accurately estimated uncertainty values. This careful management of uncertainty is critical for the application of logical inference in the robotics context, where most knowledge is abstracted from experience and is hence highly uncertain.

PLN can be used in either forward or backward chaining mode; and in the language introduced above, it can be used for either analysis or synthesis. As an example, we will consider backward chaining analysis, exemplified by the problem of a robot preschool-student trying to determine whether a new playmate "Bob" is likely to be a regular visitor to is preschool or not (evaluating the truth value of the implication $Bob \rightarrow regular\_visitor$). The basic backward chaining process for PLN analysis looks like:

1. Given an implication $L \equiv A \rightarrow B$ whose truth value must be estimated (for instance $L \equiv C\&P \rightarrow G$ as discussed above), create a list $(A_1, ..., A_n)$ of *(inference rule, stored knowledge)* pairs that might be used to produce L

2. Using analogical reasoning to prior inferences, assign each A_i a probability of success

    - If some of the A_i are estimated to have reasonable probability of success at generating reasonably confident estimates of L's truth value, then invoke Step 1 with A_i in place of L (at this point the inference process becomes recursive)
    - If none of the A_i looks sufficiently likely to succeed, then inference has "gotten stuck" and another cognitive process should be invoked, e.g.
        - **Concept creation** may be used to infer new concepts related to A and B, and then Step 1 may be revisited, in the hope of finding a new, more promising A_i involving one of the new concepts
        - **MOSES** may be invoked with one of several special goals, e.g. the goal of finding a procedure P so that $P(X)$ predicts whether $X \rightarrow B$. If MOSES finds such a procedure P then this can be converted to declarative knowledge understandable by PLN and Step 1 may be revisited....
        - **Simulations** may be run in OCP's internal simulation engine, so as to observe the truth value of $A \rightarrow B$ in the simulations; and then Step 1 may be revisited....

The combinatorial explosion of inference control is combatted by the capability to defer to other cognitive processes when the inference control procedure is unable to make a sufficiently confident choice of which inference steps to take next. Note that just as MOSES may rely on PLN to model its evolving populations of procedures, PLN may rely on MOSES to create complex knowledge about the terms in its logical implications. This is just one example of the multiple ways in which the different cognitive processes in OCP interact synergetically; a more thorough treatment of these interactions is given in [23].

In the “new playmate” example, the interesting case is where the robot initially seems not to know enough about Bob to make a solid inferential judgment (so that none of the A_i seem particularly promising). For instance, it might carry out a number of possible inferences and not come to any reasonably confident conclusion, so that the reason none of the A_i seem promising is that all the decent-looking ones have been tried already. So it might then recourse to MOSES, simulation or concept creation.

For instance, the PLN controller could make a list of everyone who has been a regular visitor, and everyone who has not been, and pose MOSES the task of figuring out a procedure for distinguishing these two categories. This procedure could then used directly to make the needed assessment, or else be translated into logical rules to be used within PLN inference. For example, perhaps MOSES would discover that older males wearing ties tend not to become regular visitors. If the new playmate is an older male wearing a tie, this is directly applicable. But if the current playmate is wearing a tuxedo, then PLN may be helpful via reasoning that even though a tuxedo is not a tie, it’s a similar form of fancy dress – so PLN may extend the MOSES-learned rule to the present case and infer that the new playmate is not likely to be a regular visitor.

Further Cognitive Synergies We have reviewed one example of pairwise synergy (between PLN and MOSES), but the intent of the architecture is that *all* pairs of significant cognitive processes should display such synergies. So, for instance, if the quantities

$$msyn(MOSES, PLN)$$

$$msyn(MOSES, simulation)$$

$$msyn(simulation, PLN)$$

are all large in the context of the CogBot system’s goals in the preschool environment, then CogBot would possess a high degree of declarative/procedural/episodic cognitive synergy in the preschool context, which is one of the main goals of the proposed project. As we will discuss later, CogBot’s degree of cognitive synergy in the preschool context can be quantitatively estimated via applying CogBot to various tasks, and measuring its effectiveness as varying degrees of attention are allocated to various cognitive processes.

E.2 Synergy Between Human-Created Knowledge Resources and Experiential Data

Another form of synergy present in OCP, and not accessible in any direct way to purely emergentist architectures, is between knowledge gained via experience and knowledge gained from human-created knowledge bases. On the spectrum between knowledge-based systems like Cyc or SOAR, versus pure experience-based systems like developmental robotics architectures, OCP and CogBot are much closer to the latter. However, in the current OCP system we do also make use of some human-created knowledge bases; our language comprehension and generation systems use the WordNet [90] and FrameNet [91] knowledge bases as well as the dictionary and grammar implicit in the Carnegie-Mellon link parser [92]. FrameNet plays a particularly central role as nodes and links structured based on FrameNet are used to mediate between linguistic experience and sensorimotor experience. And there is a potential role in OCP and CogBot for use of further knowledge bases, such as the OpenCyc [4] or SUMO [93] ontologies.

An important point to make in this context is the need for synergy between symbolic and subsymbolic knowledge, in order for a system like CogBot to make real use of any of the knowledge from knowledge bases it incorporates. Simply loading in a piece of knowledge from a source like WordNet, FrameNet or Cyc doesn’t generally help OCP or CogBot to achieve its goals, unless the system knows how to connect this piece of knowledge with its experience in a specific way. For instance, if one takes a CogBot that has spent its life in a preschool, and feeds it knowledge about the behavior of animals in the jungle, it’s not likely to be able to make much use of this knowledge. It will be able to answer questions that pertain directly to the knowledge it’s been fed, but it won’t be able to make creative use of this knowledge or vary on it in intelligent ways. On the other hand, if this preschool CogBot is fed additional knowledge that pertains to things it sees in the preschool environment, then it will likely be able to make use of this, and the imported knowledge may allow it to learn faster and perform more intelligently.

[4]`http://opencog.org`

The current proposal does not involve using additional knowledge sources beyond the ones already incorporated in OCP, but, we anticipate that for the next phase of our work *after* the proposed robot preschool project is complete, such knowledge bases may play a considerable role. Once one has an AGI system with a basic commonsense understanding of a reasonably rich subset of the world (like a preschool), it may prove quite valuable to accelerate that system's knowledge of the rest of the world via importing appropriate knowledge. The possibility of doing this is one of the advantages of hybrid architectures over pure emergentist architectures.

F Example DeSTIN Experimental Results

PI Itamar Arel has been working in the field of deep machine learning for the past 4 years. During this time, he has evaluated the deep learning concepts underlying DeSTIN on numerous applications. In this evaluation, particular focus has been placed on the ability to capture both spatial and temporal information. We briefly describe here two experiments conducted, one involving temporal pattern recognition and the other involving face recognition.

F.1 Experiments with Temporal Pattern Recognition

Most state-based (i.e. memory-based) function approximation technologies, including shallow recurrent neural networks, suffer from the inability to efficiently represent temporal information, particularly in the context of capturing long-term temporal dependencies. We have conducted experiments aimed at evaluating deep learning on this very challenge, i.e. its ability to recognize long-term regularities in dynamic patterns. In these experiments, the input to the system was deterministic pulses (comprising of three consecutive 1's) interspersed with long sequences of noisy values (zero-mean independent, identically distributed random samples). The goal of the test was to establish the system's ability to correctly predict when the sequence of three 1's appears. Note that the learning here is entirely unsupervised: no one told the system to look for sequences of three 1's. The system simply scanned the mostly-random temporal data looking for any regularity it could find, and found the only one there.

The architecture comprised a linear hierarchical topology with three layers, each of which hosting a single node (RNN). The input to the lowest-layer node was the current sample from the input sequence, while its output was to be the predicted subsequent observation. The goal of the test was thus to have accurate values produced when predicting the sequence of three 1's. The second and third layer nodes received as input the state of their lower-layer node, and produced predictions of subsequent state values, as described above. Each of the three nodes consisted of 8 hidden (state) neurons. The system required 1000 presentations of the deterministic sequences (separated by the random ones) to achieve 100% accuracy (based on a threshold value of 0.98), for random sequence intervals ranging from 1 to 256 time steps – an achievement that clearly demonstrates

- the ability to capture long-term temporal dependencies
- its core capacity to model data pertaining to different modalities
- its inherent capability to overcome the short-term memory shortcoming characterizing most shallow architectures.

F.2 Experiments with Face Recognition

As a more practical example, we now discuss how the perception processing technology described in this proposal was recently successfully applied to a real-time face recognition task. In this task, the goal of the system was to accurately classify an image as either being of a pre-defined owner (i.e. particular individual) or not. The process involved three phases: an exposure phase, during which the system was presented with a large number of unlabeled examples of both owner images and non-owner images, a training phase during which the system was trained (using a simple classifier) to recognize instances of owner images, and a testing phase in which the system was expected to correctly classify images not presented during the training stage.

For these experiments, all source images were converted to 64x64 pixel, gray-scale images to be used in during all phases; but a similar approach may be taken using color vision.

The deep learning topology consisted of four layers. Each of the nodes in the lowest layer received a tile of 4x4 pixels as input. To that end, 256 nodes at the lowest layer covered each input image. Every layer above the first comprised nodes which received as input complete state information from four lower-layer nodes, with connectivity as detailed in the body of this proposal. This resulted in a reduction of four-fold at each layer, yielding a network breakdown as follows: layer (1) - 256 nodes, layer (2) - 64 nodes, layer (3) - 16 nodes and layer (4) – 4 nodes and in layer (5) a single node.

In this particular realization of the technology, the top node (layer 4) functioned as a classifier. In such capacity, the top node received as input the states of the four nodes of layer 3, but was expected to produce as output the classification result, which in this case translated to a single output reflecting the owner/non-owner decision. There was no feedback provided by the top layer node to any of the layers below, since it became apparent that such feedback was unnecessary in this particular application.

Both training and testing consisted of presenting images in the form of random segments of motions. Every segment duration was randomly selected to be of 7 to 13 pixels long, and have a random direction of either top, bottom, left or right. By employing such segments of motion, dynamics are introduced to the inputs of the nodes of the first layer, thus enabling the entire architecture to represent spatiotemporal dependencies. The number of segments chosen for each new image was also randomly selected, ranging between 9 to 11. By applying such randomness to the segments of motion, one forces all nodes to focus on the informative content of each image rather than memorizing motion trends, as means of improving prediction performance. This resulted in typical overall presentation duration of approximately 100 steps for each image. As information propagated to the higher layers, a belief is formed of the content of the image observed.

Training was performed on images collected from over 50 short video sequences, containing both owner and non-owner facial images. The process involved selecting a random image, presenting it to the system as described above, updating the parameters and then repeating the process for a subsequent randomly selected image. As would be expected, the prediction error for each of the layers gradually decreased. The training process was stopped upon reaching 100,000 image presentations, at which point the system was ready for testing. The latter consisted of presenting the system with images that were not included in the training set. The presentation scheme was identical to the one used in training, with the exception of not applying any weight updates.

The results coincided with intuitive expectations, offering over 94% recognition accuracy under diverse real-world conditions (lighting, noise, scale variability, etc.). Due to the simplicity of the testing operation (which consists of feedforward functions only), the high operational speed of the system was also noted. In addition to off-line testing, a real-time testing setup was established in which still images from a web-camera producing streaming video were captured and applied to the system. The result was a successful real-time face recognition system, thus solidifying the framework and its ability to deliver robust pattern inference under distorted conditions.

G Detailed Task Breakdown

Here we break down the phases and tasks described in Table 2 in more detail.

G.1 Phase 1: Initial Integration

This phase lasts six months. Table 3 lists the major tasks, their duration, and timing within the project phase.

The main deliverables for this phase are:

1. Internal project and process documents describing practices and guidelines for the development-oriented activities.
2. DeSTIN code integrated within the OpenCogPrime codebase, with a pipeline architecture between the two subsystems.

Task	Duration	Execution Months
Detailed planning	1 week	1
Environment setup, development process training	2 weeks	1
Integration of Nao APIs with DeSTIN	1 month	1-2
Tuning DeSTIN action hierarchy to Nao actuators	6 weeks	2-3
Tuning DeSTIN perception hierarchy to preschool context	6 weeks	4-5
Code-level integration between DeSTIN and OCP	6 weeks	5-6
Phase conclusion workshop	2 days	6

Table 3: Major Phase 1 Tasks

3. Custom Ubuntu Linux installation CD including integrated system, Nao API libraries and other dependencies. This deliverable will be based on the existing CogBuntu customization of Ubuntu Linux for OpenCogPrime development.

G.2 Phase 2: Intelligence Testing Tasks 1-4

This phase lasts four months. Table 4 lists the major tasks, their duration, and timing within the project phase.

Task	Duration	Execution Months
Detailed planning	1 week	1
Detailed testing task specifications	4 weeks	1-2
Experiments and tuning for task 1	3 weeks	2
Experiments and tuning for task 2	3 weeks	3
Experiments and tuning for task 3	2 weeks	3-4
Experiments and tuning for task 4	2 weeks	4
Video recording and report writing	2 weeks	4
Phase conclusion workshop	2 days	4

Table 4: Major Phase 2 Tasks

The main deliverables for this phase are:

- Detailed specification for intelligence testing tasks 1 through 4, containing test cases, pre-conditions, and scoring metrics.
- Web pages with descriptions of each test case for each intelligence testing task, with accompanying videos showing CogBot performing each task.
- Technical reports for each intelligence testing task, describing experimental setup, learning and tuning procedure, and empirical results.
- *"Integrative intelligence" evidence of CogBot performing intelligence testing tasks 1 through 4 in sequence, without restarts or parameter adjustments.*

G.3 Phase 3: Implementation Improvements

This phase lasts eight months. Table 5 lists the major tasks, their duration, and timing within the project phase.

The main deliverables for this phase are:

1. Updated codebase containing the DeSTIN goal and action hierarchy, and the expanded PLN, ECAN, internal simulator and concept formation code for OpenCogPrime.
2. Experimental reports where pertinent, showing how the improvements to the codebase result in improved performance in comparison with the Phase 1 baseline.

Task	Duration	Execution Months
Detailed planning	1 week	1
Refinements to DeSTIN's goal and action hierachies	6 weeks	1-2
Concept formation dynamics in OCP	4 weeks	2-3
Tuning PLN inference in OCP	2 months	3-5
Tuning economic attention allocation in OCP	4 weeks	5-6
Improving OCP's world simulation module	4 weeks	6-7
Dedicated system-level testing	4 weeks	7-8
Phase conclusion workshop	2 days	8

Table 5: Major Phase 3 Tasks

G.4 Phase 4: Intelligence Testing Tasks 5-8

This phase lasts four months. Table 6 lists the major tasks, their duration, and timing within the project phase.

Task	Duration	Execution Months
Detailed planning	1 week	1
Detailed testing task specifications	4 weeks	1-2
Experiments and tuning for task 5	3 weeks	2
Experiments and tuning for task 6	2 week	3
Experiments and tuning for task 7	2 weeks	3-4
Experiments and tuning for task 8	3 weeks	4
Video recording and report writing	2 weeks	4
Phase conclusion workshop	2 days	4

Table 6: Major Phase 4 Tasks

The main deliverables for this phase are:

- Detailed specification for intelligence testing tasks 5 through 8, containing test cases, pre-conditions, and scoring metrics.
- Web pages with descriptions of each test case for each intelligence testing task, with accompanying videos showing CogBot performing each task.
- Technical reports for each intelligence testing task, describing experimental setup, learning and tuning procedure, and empirical results.
- *"Integrative intelligence" evidence of CogBot performing intelligence testing tasks* ***1 through 8*** *in sequence, without restarts or parameter adjustments.*

G.5 Phase 5: Deep Integration

This phase lasts six months. Table 7 lists the major tasks, their duration, and timing within the project phase.

The main deliverables for this phase are:

1. Updated codebase reflecting deep integration between DeSTIN and OpenCogPrime.
2. Scientific publication detailing the conceptual design for deep integration between DeSTIN and OpenCog-Prime, describing the major issues and key decisions, as well as expected result

Task	Duration	Execution Months
Detailed planning and conceptual design	4 weeks	1
Software-level integration	6 weeks	2-3
DeSTIN parameter tuning	4 weeks	3-4
Economic attention allocation parameter tuning	6 weeks	4-5
Report writing	3 weeks	6
Phase conclusion workshop	2 days	6

Table 7: Major Phase 5 Tasks

Task	Duration	Execution Months
Detailed planning	1 week	1
Detailed testing task specifications	6 weeks	1-2
Experiments and tuning for task 9	3 weeks	2-3
Experiments and tuning for task 10	4 weeks	3-4
Experiments and tuning for task 11	4 weeks	4-5
Experiments and tuning for task 12	3 weeks	5
Experiments and tuning for task 13	3 weeks	6
Experiments and tuning for task 14	4 weeks	6-7
Video recording and report writing	6 weeks	7-8
Phase conclusion workshop	2 days	8

Table 8: Major Phase 6 Tasks

G.6 Phase 6: Intelligence Testing Tasks 9-14

This phase lasts eight months. Table 8 lists the major tasks, their duration, and timing within the project phase.

The main deliverables for this phase are:

- Detailed specification for intelligente testing tasks 9 through 14, containing test cases, pre-conditions, and scoring metrics.
- Web pages with descriptions of each test case for each intelligence testing task, with accompanying videos showing CogBot performing each task.
- Technical reports for each intelligence testing task, describing experimental setup, learning and tuning procedure, and empirical results.
- *"Integrative intelligence" evidence of CogBot performing intelligence testing tasks **1 through 14** in sequence, without restarts or parameter adjustments.*
- Scientific publications describing the experimental results, highlighting details and conclusions from empirical studies on cognitive synergy.
- Project conclusion report, summarizing achievements and describing future work directions.

References

[1] I. Arel, D. Rose, and T. Karnowski, “A deep learning architecture comprising homogeneous cortical circuits for scalable spatiotemporal pattern inference.” *NIPS 2009 Workshop on Deep Learning for Speech Recognition and Related Applications*, Dec 2009.

[2] B. Goertzel, “Opencog prime: A cognitive synergy based architecture for embodied artificial general intelligence,” in *Proceedings of ICCI-09, Hong Kong*, 2009.

[3] B. Goertzel and C. Pennachin, *Artificial General Intelligence.* Springer, 2005.

[4] B. G. Stan Franklin and P. W. (Editors), *Proceedings of First Conference on Artificial General Intelligence.* IOS Press, 2008.

[5] P. H. Marcus Hutter and B. G. (Editors), *Proceedings of Second Conference on Artificial General Intelligence.* Atlantis Press, 2008.

[6] N. C. (Editor), *Human-Level Intelligence (Special Issue of Artificial Intelligence Magazine).* AAAI, 2006.

[7] P. Hayes and K. Ford., “Turing test considered harmful,” *IJCAI-14*, 1995.

[8] R. French., “Subcognition and the limits of the turing test’,” *Mind*, 1990.

[9] N. Alvarado, S. S. Adams, S. Burbeck, and C. Latta, “Beyond the turing test: Performance metrics for evaluating a computer simulation of the human mind,” *Development and Learning, International Conference on*, vol. 0, p. 147, 2002.

[10] A. Turing, “Computing machinery and intelligence,” *Mind*, vol. 59, 1950.

[11] D. M. Neilsen, *Teaching Young Children, Preschool-K: A Guide to Planning Your Curriculum, Teaching Through Learning Centers, and Just About Everything Else.* Corwin Press, 1998.

[12] B. Goertzel, *The Hidden Pattern.* Brown Walker, 2006.

[13] L. S. Gottfredson, “The general intelligence factor,” *Scientific American Presents*, vol. 9.

[14] S. Legg and M. Hutter, “A collection of definitions of intelligence,” in *Ben Goertzel and Pei Wang (eds), Advances in Artificial General Intelligence.* IOS, 2007.

[15] B. Goertzel, “Toward a formal definition of real-world general intelligence,” in *Proceedings of AGI-10, Lugano*, 2010.

[16] M. Hutter, *Universal AI.* Springer, 2005.

[17] N. Nilsson, *Hierarchical Bayesian Optimization Algorithm: Toward a New Generation of Evolutionary Algorithms.* Springer, 2005.

[18] M. Richardson and P. Domingos, “Markov logic networks,” *Machine Learning*, 2006.

[19] B. Goertzel, I. G. M. Iklé, and A. Heljakka, *Probabilistic Logic Networks.* Springer, 2008.

[20] M. Williams and J. Williamson, “Combining argumentation and bayesian nets for breast cancer prognosis,” *Journal of Logic, Language and Information*, 2006.

[21] L. S. Zettlemoyer, H. M. Pasula, and L. P. Kaelbling., “Logical particle filtering,” *Proceedings of the Dagstuhl Seminar on Probabilistic, Logical, and Relational Learning*, 2007.

[22] S. Thrun and et al., “The robot that won the darpa grand challenge.” *Journal of Robotic Systems*, vol. 23-9, 2006.

[23] B. Goertzel, "Cognitive synergy: A universal principle of feasible general intelligence?" in *Proceedings of ICCI-09, Hong Kong*, 2009.

[24] ——, "The embodied communication prior," in *Proceedings of ICCI-09, Hong Kong*, 2009.

[25] E. Tulving and R. Craik, *The Oxford Handbook of Memory.* Oxford University Press, 2005.

[26] G. Fauconnier and M. Turner, *The Way We Think: Conceptual Blending and the Mind's Hidden Complexities.* Basic, 2002.

[27] M. Looks, *Competent Program Evolution.* PhD Thesis, Computer Science Department, Washington University, 2006.

[28] B. Goertzel and C. P. et al, "An integrative methodology for teaching embodied non-linguistic agents, applied to virtual animals in second life," in *Proc. of AGI-08*, 2008.

[29] B. Goertzel, J. Pitt, M. Ikle, C. Pennachin, and R. Liu, "Glocal memory: a design principle for artificial brains and minds," *Neurocomputing, Special Issue of Artificial Brain*, to appear.

[30] J. Bach, *Principles of Synthetic Intelligence.* Oxford University Press, 2009.

[31] B. Goertzel, C. Pennachin, and N. Geisweiller, *Building Better Minds: Engineering Beneficial General Intelligence.* In preparation, 2010.

[32] B. Goertzel, H. Pinto, and C. Pennachin, "Using dependency parsing and probabilistic inference to extract relationships between genes, proteins and malignancies implicit among multiple biomedical research abstracts," in *Proceedings of the BioNLP Workshop/HLT-NAACL*, 2006.

[33] J. Hawkins and S. Blakeslee, *On Intelligence.* Brown Walker, 2006.

[34] T. Serra, A. Oliva, and T. Poggio, "A feedforward architecture accounts for rapid categorization," *Proceedings of the National Academy of Sciences*, vol. 104-15, 2007.

[35] J. Albus and A. Meystel, *Engineering of Mind: An Introduction to the Science of Intelligent Systems.* Wiley and Sons, 2001.

[36] G. E. Hinton, S. Osindero, and Y. Teh, "A fast learning algorithm for deep belief nets." *Neural Computation*, vol. 18, pp. 1527–1554, 2006.

[37] Y. LeCun, B. Boser, J. S. Denker, and et al., "Handwritten digit recognition with a back-propagation network." *Advances in Neural Information Processing Systems*, vol. 2, 1990.

[38] D. Wechsler, *WPPSI-III Technical and Interpretive Manual.* The Psychological Corporation, 2002.

[39] B. Goertzel and S. Bugaj, "Stages of cognitive development in uncertain inference based ai systems," in *Advances in Artificial General Intelligence*, 2007.

[40] ——, "Stages of ethical development in uncertain inference based ai systems," in *Proc. of AGI-08*, 2008.

[41] A. Newell, *Unified Theories of Cognition.* Harvard University press, 1990.

[42] R. Caruana, "Multitask learning," *Machine Learning*, 1997.

[43] S. Thrun and T. Mitchell, "Lifelong robot learning," *Robotics and Autonomous Systems*, 1995.

[44] S. Ben-David and R. Schuller, "Exploiting task relatedness for learning multiple tasks," *Proceedings of the 16th Annual Conference on Learning Theory*, 2003.

[45] M. Taylor and P. Stone, "Cross-domain transfer for reinforcement learning." *Proceedings of the 24th International Conference on Machine Learning*, 2007.

[46] M. Rosenstein, Z. Marx, T. Dietterich, and L. P. Kaelbling, "Transfer learning with an ensemble of background tasks," *NIPS workshop on inductive transfer*, 2005.

[47] A. Laud and G. Dejong, "The influence of reward on the speed of reinforcement learning," *Proceedings of the 20th International Conference on Machine Learning*, 2003.

[48] W. Duch, R. Oentaryo, and M. Pasquier, "Cognitive architectures: Where do we go from here?" *Proceedings of the Second Conference on AGI*, 2008.

[49] N. Nilsson, "The physical symbol system hypothesis: Status and prospects," in *50 Years of AI,*, 2007, p. 917.

[50] J. Laird, P. Rosenbloom, and A. Newell, "Soar: An architecture for general intelligence." *Artificial Intelligence*, vol. 33, 1987.

[51] D. E. Meyer and D. E. Kieras, "A computational theory of executive cognitive processes and multiple-task performance: Part 1," *Psychological Review*, vol. 104, 1997.

[52] P. Langley, "An adaptive architecture for physical agents." *Proc. of the 2005 IEEE/WIC/ACM Int. Conf. on Intelligent Agent Technology*, 2005.

[53] S. Shapiro and et al., "Metacognition in sneps," *AI Magazine*, vol. 28, 2007.

[54] D. Lenat and R. V. Guha, *Building Large Knowledge-Based Systems: Representation and Inference in the Cyc Project.* Addison-Wesley, 1990.

[55] I. Arel, D. Rose, and R. Coop, "Destin: A scalable deep learning architecture with application to high-dimensional robust pattern recognition." *Proc. AAAI Workshop on Biologically Inspired Cognitive Architectures*, Nov 2009.

[56] R. C. O'Reilly, T. S. Braver, and J. D. Cohen, "A biologically-based computational model of working memory." in *Models of Working Memory*, A. Miyake and P. Shah, Eds., 1999, pp. 375–411.

[57] J. Fleischer, J. Gally, G. Edelman, and J. Krichmar, "Retrospective and prospective responses arising in a modeled hippocampus during maze navigation by a brain-based device," *PNAS*, vol. 104, 2007.

[58] J. Schmidhuber, "Curious model-building control systems.." *Proc. International Joint Conference on Neural Networks*, 1991.

[59]

[60]

[61]

[62] J. Han, S. Zeng, K. Tham, M. Badgero, and J. Weng, "Dav: A humanoid robot platform for autonomous mental development,," *Proc. 2nd International Conference on Development and Learning*, 2002.

[63] J. Weng, W. S. Hwang, Y. Zhang, C. Yang, and R. Smith, "Developmental humanoids: Humanoids that develop skills automatically,," *Proc. the first IEEE-RAS International Conference on Humanoid Robots*, 2000.

[64] J. Weng and W. S. Hwangi, "From neural networks to the brain: Autonomous mental development," *IEEE Computational Intelligence Magazine*, 2006.

[65]

[66] F. Kaplan, "Neurorobotics: an experimental science of embodiment," *Frontiers in Neuroscience*, 2008.

[67] P. Oudeyer and F. Kaplan, "Discovering communication," *Connection Science*, 2006.

[68] J. Modayil and B. Kuipers, "Autonomous development of a grounded object ontology by a learning robot." *AAAI-07*, 2007.

[69] J. Mugan and B. Kuipers, "Towards the application of reinforcement learning to undirected developmental learning." *International Conference on Epigenetic Robotics*, 2008.

[70] J. Mugan and B. Kuipers., "Autonomously learning an action hierarchy using a learned qualitative state representation," *IJCAI-09*, 2009.

[71] G. Metta, G. Sandini, D. Vernon, L. Natale, and F. Nori, "The icub humanoid robot: an open platform for research in embodied cognition," *Performance Metrics for Intelligent Systems Workshop (PerMIS 2008)*, 2008.

[72] J. Schmidhuber, "Developmental robotics, optimal artificial curiosity, creativity, music, and the fine arts," *Connection Science*, 2006.

[73] B. Bakker and J. Schmidhuber, "Hierarchical reinforcement learning based on subgoal discovery and subpolicy specialization," *Proceedings of the 8-th Conference on Intelligent Autonomous Systems*, 2004.

[74] B. Bakker, V. Zhumatiy, G. Gruener, and J. Schmidhuber, "Quasi-online reinforcement learning for robots," *Proceedings of the International Conference on Robotics and Automation*, 2006.

[75] J. Simsek and A. Barto, "An intrinsic reward mechanism for efficient exploration," *Proceedings of the Twenty-Third International Conference on Machine Learning*, 2006.

[76] S. Singh, A. Barto, and N. Chentanez, "Intrinsically motivated reinforcement learning," *Proceedings of Neural Information Processing Systems 17*, 2005.

[77] M. H. Lee, Q. Meng, and F. Chao, "Developmental learning for autonomous robots," *Robotics and Autonomous Systems*, 2007.

[78] ——, "Staged competence learning in developmental robotics," *Adaptive Behavior*, 2007.

[79] D. Campolo, F. Taffoni, G. Schiavone, D. Formica, E. Guglielelli, and F. Keller, "Technology for ecological assessment of sensori-motor coordination in infants," *International Journal of Social Robotics.*, 2007.

[80] N. Nilsson, "The physical symbol system hypothesis: Status and prospects," *50 Years of AI, Festschrift, LNAI 4850*, vol. 33, 2009.

[81] J. R. Anderson and C. Lebiere, "The newell test for a theory of cognition." *Behavioral and Brain Science*, vol. 26, 2003.

[82] W. Gray, M. Schoelles, and C. Myers, "Meeting newells other challenge: Cognitive architectures as the basis for cognitive engineering." *Behavioral and Brain Sciences*, 2009.

[83] R. Sun and X. Zhang, "Top-down versus bottom-up learning in cognitive skill acquisition." *Cognitive Systems Research*, vol. 5, 2004.

[84] A. Nestor and B. Kokinov, "Towards active vision in the dual cognitive architecture." *International Journal on Information Theories and Applications*, vol. 11, 2004.

[85] S. Franklin, "The lida architecture: Adding new modes of learning to an intelligent, autonomous, software agent," *Int. Conf. on Integrated Design and Process Technology*, 2006.

[86] N. Cassimatis, "Adaptive algorithmic hybrids for human-level artificial intelligence," 2007.

[87] L. Shastri and V. Ajjanagadde, "From simple associations to systematic reasoning: A connectionist encoding of rules, variables, and dynamic bindings using temporal synchrony," *Behavioral Brain Sciences*, vol. 16-3, 1993.

[88] D. Hofstadter, *Fluid Concepts and Creative Analogies.* Basic Books, 1996.

[89] ——, *Fluid Concepts and Creative Analogies.* Basic Books, 1995.

[90] C. Fellbaum, *WordNet: An Electronic Lexical Database.* Addison-Wesley, 1990.

[91] C. Baker, C. Fillmore, and J. Lowe, "The berkeley framenet project." *Proceedings of the COLING-ACL*, 1998.

[92] D. Sleator and D. Temperley, "Parsing english with a link grammar," *Third International Workshop on Parsing Technologies.*, 1993.

[93] I. Niles and A. Pease, "Towards a standard upper ontology," *Proceedings of the 2nd International Conference on Formal Ontology in Information Systems*, 2001.