Four Contemporary AGI Designs: a Comparative Treatment

September 26, 2006
Stan FRANKLIN, Ben GOERTZEL, Alexei SAMSONOVICH, Pei WANG

Introduction (by Ben Goertzel)

During his talk at the AGI Workshop, Stan Franklin suggested that his LIDA architecture might fruitfully be considered not only as a specific AGI design, but also as a general framework within which to discuss and compare various AGI designs and approaches. With this in mind, following the workshop itself, I (Goertzel) formulated a list of simple questions intended to be pertinent to any AGI software design, mostly based on the conceptual framework presented in Stan Franklin's workshop presentation (and represented in this volume by his article "A Foundational Architecture for Artificial General Intelligence") with a couple additions and variations. All individuals who presented talks on AGI architectures at the workshop were invited to respond to the questionnaire, giving answers appropriate to their own AGI design. Four individuals (myself, Wang, Franklin and Samsonovich) took up this offer, and their answers are reported here – without modification; exactly as they gave them. All of these designs are described to some extent elsewhere in the book.

I find this style of reportage an interesting one, and pleasantly different from the standard approach in which each AGI system is described in its own paper in terms of its own specialized terminology, and with its own peculiar focus. Potentially, one could consider this simple questionnaire as a first step toward the creation of a systematic typology and ontology of AGI approaches. The question of what are the key questions to ask about an AGI system, if one wishes to understand how it differs from other AGI systems and what are its truly essential aspects, is an interesting question unto itself, and asking it leads one straight into the heart of the AGI enterprise. In future I would find it interesting to repeat this sort of experiment with a more detailed questionnaire and a larger group of AGI system designers.

1. Questions and Responses

- What kind of **environment** is your AI system intended to interact with?

Franklin:

The external environment of the original IDA consisted of email messages from sailors, and of responses to database queries. The later LIDA (Learning IDA) is being developed in two domains, one to control robots in a University environment (ROBLIDA), and the other in an image database environment (ILIDA).\

Goertzel:

Novamente may interact with any sort of environment, including totally nonphysical environments such as pools of mathematical theorems. However, we are currently utilizing it in two contexts. One is language processing, in which the environment is simply text that passes back and forth between it and human users or human document repositories. And the other is the control of a simple humanoid agent in a 3D simulation world called AGISim. Actual control of physical robots is also of interest to us, but we haven't gotten there yet, for pragmatic rather than principled reasons.

Samsonovich:

Currently, it works in a simplistic 2-D virtual environment. However, the architecture is designed to be able to handle any environmental embedding (real or virtual), provided the interface capabilities are implemented and available to the architecture.

Wang:

NARS interacts with its environment in real time. The environment can feed the system with any task (new knowledge, question, or goal) that is expressible in Narsese, the knowledge representation language of the system. No restriction is made on the content and timing of the tasks.

- What mechanisms are used for "**perception**" (in the sense of mapping sensory data into a set of relationships that can be used by the AI system to make significant practical decisions)?

Franklin:

IDA and LIDA model perception using ideas from the Copycat Architecture of Hofstadter and Mitchell.

Goertzel:

There is a special perception subsystem called the Perceptual Pattern Miner. Basically, it recognizes repeated spatiotemporal patterns in sensory inputs. The same algorithm is applied to perceptual inputs, to records of actions taken, and also to records of cognitions taken – so perception spans sensation, action and thought, rather than just being perception of the external world. Algorithmically, the Perceptual Pattern Miner in its currently implemented version searches for patterns that are conjunctions of predicates, where the predicates it looks for are provided to it from two sources: some hard-wired predicates related to the structure of space and time, and predicates flagged as important by cognition. The current version doesn't take the hierarchical structure of perceptual patterns directly into account but a future version may.

Samsonovich:

A set of special functions called "primitives" that constitute the procedural memory are used to do sensory perception. We assume that these functions will be able to do signal-to-symbol conversion in any given environment / embodiment.

Examples of top-down control: the architecture may send a request to the sensory system to pay attention to a particular location, to look for a particular feature, or to re-examine acquired data.

Wang:

Sensors are optional parts of NARS. The system can accept any sensor that can be invoked by an operation in the format of (*operator, argument-list*), and can produce effects expressible in Narsese.

- What mechanisms are used for the execution of procedures embodying **external actions**? (e.g. procedures involving coordinated sets of specific commands to be sent to actuators)?

Franklin:

In an essentially rule-based manner, IDA filled in the blanks in built-in scripts. ROBLIDA will likely use subsumption networks a la Brooks. I have no answer as yet for ILIDA.

Goertzel:

The Combo language, which represents procedures inside Novamente, contains primitives correcponding to specific external actions (such as, e.g.: "move this joint by this amount at this angle").

Samsonovich:

Again, special functions called "primitives" are used to execute actions. In particular, they are supposed to provide the higher symbolic level with a proprioceptive feedback.

Wang:

Effectors are optional parts of NARS. The system can accept any effector that can be invoked by an operation in the format of (*operator, argument-list*), and can produce effects expressible in Narsese.

- How are **procedures** stored in memory? (This has two aspects: what form do individual procedures take in memory; and how is the overall memory store of procedures organized?)

Franklin:

IDA used behavior codelets. LIDA uses a scheme net (schema a la Drescher). The links in the scheme net are "derived from" links.

Goertzel:

Executable procedures are stored in data objects which are equivalent to programs in a simple, LISP-like language called Combo. These Procedure objects are stored in a ProcedureRepository; and each of these objects is associated with a ProcedureNode that resides in semantic, declarative memory. Implicit knowledge about procedures is contained in declarative memory, and may be used to create new Procedure objects via a process called "predicate schematization." A key point is that Procedures, whenever possible, are broken down into small modules: this makes for effective real-time context-switching between various currently active procedures; and also makes procedure learning and procedural inference much easier.

Samsonovich:

Again, the lower-level (interface) procedures are stored as special function in a separate module called "procedural memory". Currently, there is no definite specification as to how the procedural memory should be organized. In any case, procedural memory is not the focus of our approach.

Overall, our architecture has four memory systems: procedural, working, semantic and episodic, plus the input-output buffer (in addition to the other 3 components; together it has 8 components).

Wang:

Procedures are represented as operations, which are statements under procedural interpretation. An atomic operation has the format of (*operator, argument-list*), and compound operations are formed from simpler operations and statements by logical operators.

- How are **episodes** (experienced by the system itself) stored in memory? (Again, this has two aspects: what form do individual episodes take in memory; and how is the overall memory store of episodes organized?)

Franklin:

Episodic memory is implemented by variants of Kanerva's sparse distributed memory. In IDA episodic structure was hand-crafted to fit the domain. In early experimentation for LIDA Philmore's case grammer was use. I don't yet know what will follow.

Goertzel:

Temporal relationships are represented using a special form of temporal logic, which extends Novamente's standard "probabilistic logic networks" variant of probabilistic logic, and integrates probabilistic knowledge representation with ideas from "event calculus." Episodes are then collections of knowledge bound together by temporal relationships -- and which form "maps" in Novamente's knowledge base, in the sense that an episode is a set of knowledge-items that the system frequently finds it useful to utilize as a whole.

Samsonovich:

Episodic memories are stored in our architecture as mental states. Each mental state is an instance of a Self of the agent taken together with all its current experiences (represented as instances of schemas).

Overall, episodic memory is clustered into episodes. A set of mental states may form a cluster (an episode) based on their logical and functional interdependence as well as proximity in time and space. An episode is comparable to a single snapshot of the entire working memory.

Wang:

Episodes are represented as events, which are statements with temporal attributes. Like all statements, they are stored in the related concepts.

- How are facts and conjectures ("**declarative knowledge**") stored in memory? (Again, this has two aspects: what form do individual facts/conjectures take in memory; and how is the overall memory store of facts/conjectures organized?)

Franklin:

Facts and conjectures are constructed in the workspace and are stored in declarative memory, perhaps eventually becoming part of it's subset, semantic memory.

Goertzel:

Declarative knowledge is represented using a semantic network type approach, involving types nodes and links. Some nodes represent abstract concepts, some represent particular objects, some represent percepts or actions, and some represent patterns not well described by any of these terms. Links, depending on type, may represent relationships such as logical inheritance, implication, equivalence, or else Hebbian-style association or ordered or unordered grouping, etc.

Finally, as well as knowledge stored explicitly in individual logical links, there is also implicit knowledge, stored in the pattern of linkages. For instance, if a set of nodes are tightly interlinked with Hebbian association links, then when a few nodes in the set are utilized, the others will tend to be utilized afterwards (due to the system's attention allocation dynamics) – so the set as a whole may be implicitly used to represent an item of knowledge.

Samsonovich:

Semantic memory in our architecture stores general knowledge of the agent. It consists of a set of schemas organized by a global semantic net (each schema corresponds to a node in this net). A schema is a very general form of representation that allows us to represent any given category or a class of categories. It can be conceived as a graph or as a 2-D table of nodes. All nodes have one and the same standard set of attributes (about 20 attributes). Nodes are connected to each other via bindings, etc.

In addition, semantic memory includes a part called "reference memory" which represents agent's beliefs about the most current actual state of the world. E.g., reference memory can be implemented as a spatial map with instances of schemas allocated on it. We still have debates regarding the implementation of reference memory.

Wang:

All declarative knowledge are represented as Narsese statements, and stored in the related concepts.

- How does "**attention**" work? How is it quantified and what mechanisms control its change over time? (Here attention is considered as the process that brings information from perception and long-term (episodic/declarative) memory into the attentional focus.)

Franklin:

Attention is the work of attention codelets who gather coalitions that compete for access to consciousness, the global workspace. The winner is broadcast throughout the cognitive system.

Goertzel:

Each node or link in Novamente is labeled with two numbers collectively comprising an "attention value," called the short-term and long-term importance. Roughly, the former governs processor time allocation and the latter governs whether the node/link is allowed to stay in memory or not. Importance levels are dynamically updated using a set of equations based on "artificial economics", involving two currencies, one for short term and one for long term importance. So, for instance, the "moving bubble of attention" is an emergent dynamical phenomenon, which at any point in time consists of the nodes/links with the highest short-term importance.

Samsonovich:

Each schema node has an attribute "attention" that determines the probability of selecting this node for processing, etc. Special rules implemented in our "driving engine" (one of the 8 components of the architecture) control the dynamics of attention.

Wang:

Attention works in NARS as a dynamic resources allocation mechanism. Each task has a priority indicating its share of processing time, and each belief has a priority indicating is accessibility.

- How does "**filtering**" work – i.e., how does the attentional mechanism cope with filtering the vast number of items that may be seeking attention at any given time?

Franklin:

Filtering occurs in the sensory mechanism, by the selection of the sensors. It occurs again during perception as the percept is formed. It occurs when local associations are cued from the episodic memories. It occurs during the competition of coalitions for consciousness. It occurs during instantiation of schemes from procedural memory. Finally, it occurs during the selection of a single action at the end of a cognitive cycle. Filtering can also occur during higher-level cognitive processing, say during deliberation or volition. Filtering seems to be ubiquitous during cognition.

Goertzel:

Filtering of sensory data occurs implicitly via the attention allocation mechanism (the dynamics of short-term importance levels). Low-level percepts are assigned a low-level of long-term importance, and in most cases never get a high level, so they're almost always forgotten shortly after they arise. The patterns recognized among these low-level percepts generally are assigned a higher level of long-term importance, and are more likely to be remembered.

Samsonovich:

There are several mechanisms that can be called "filtering mechanisms" in our architecture; most of them do not involve attention control. First, there are "syntactic" rules of binding schemas and their instances to each other. Then, filtering of candidates for binding is done with the help of our neuromorphic cognitive maps (one of the 8 components). Finally, there are special attributes (activation, attention) that in effect do filtering.

Wang:

Filtering works in NARS as the consequence of resources competition among tasks, beliefs, and concepts. Only the items with high priority get time-space resources needed for their processing.

- How does "**action selection**" work? For example, how is the choice made between two tasks serving two different concurrent goals?

Franklin:

Action selection employs an enhancement of Maes' behavior net. It deals effectively with concurrent goals, dangers, unexpected opportunities, etc.

Goertzel:

Action selection is carried out using an application of economic attention allocation, which may (taking some liberties) be thought of as an extension/modification of Maes' behavior net. Actions are represented by Procedure objects that are broken down into modules, and these modules are connected both by probabilistic logical implication relationships (representing "precondition" relationships between modules) and by activation relationships along which currency values representing "short term importance" pass between modules. The dynamics of action selection then emerge as a consequence of the dynamics of probabilistic reasoning and economic attention allocation. This dynamics can pass action back and forth between different procedures in the "active procedure pool" of currently active procedures, via the rule that, when multiple modules in the active procedure pool have their preconditions met but cannot be simultaneously executed without interfering with each other, then the one with the largest short-term-importance currency is selected for execution.

Samsonovich:

There is a special higher-level procedure of a voluntary action implemented as a part of our driving engine. (There is a set of higher-level procedures, including perception, understanding, voluntary action, checking the result, conflict resolution, and many more; however, these are not parts of procedural memory, as they are executed in full awareness of the system of what and how is done).

Wang:

Usually "selection" becomes "distribution", in the sense that all the tasks will be processed, but at different speeds. As a special case, new goals are created by a decision-making mechanism from desirable and plausible events.

- What **learning mechanisms** exist in your AI system? For each mechanism, specify: what kinds of learning problems is it specialized for, and how does it interact with other learning mechanisms?

Franklin:

IDA did not learn. LIDA implements perceptual, episodic, and procedural learning, and later will include attentional learning. They interact only indirectly.

Goertzel:

Very broadly speaking, there are three main learning mechanisms in Novamente: probabilistic inference, (probabilistic) evolutionary learning, and greedy stochastic pattern mining. Each of these can appeal to the others internally to help it out, quite directly. Furthermore, each of these can be used in multiple ways. For instance, probabilistic inference and evolutionary learning can both be used for both declarative and procedural learning; and, conjunctive pattern mining is used for both perceptual pattern mining and for "map formation" (recognition, and explicit representation, of concepts that exist only implicitly as collections of commonly-utilized nodes/links).

Samsonovich:

There is a number of learning mechanisms, ranging in complexity from almost trivial episodic memory creation (by transferring mental states from working memory to episodic memory) to new schema creation, off-line self-analysis and cognitive map self-organization (involved in, e.g., building a system of values). This is a long story.

Wang:

All object-level knowledge in NARS can be learned, by several mechanisms:

a) New tasks/beliefs/concepts can be accepted from the environment;
b) New tasks and beliefs can be derived from existing ones by inference rules;
c) The truth-value of beliefs can be modified by the revision rule;
d) New concepts can be formed from existing concepts by compound-term composition/decomposition rules;
e) The priority values of tasks/beliefs/concepts can be adjusted by the feedback evaluation mechanism.
f) Existing tasks/beliefs/concepts can be removed by the forgetting mechanism.

Since each of the above mechanisms works on a different aspect of the memory, it does not need to directly interact with the others.

- How is **meta-cognition** (reflective cognition about cognition, leading potentially to goal-directed self-modification of cognitive processes) implemented in your system?

Franklin:

We implemented two versions of meta-cognition in IDA, each as a Minsky style B-brain, and each using a different mechanism. This was the wrong way to approach the problem. In LIDA we'll implement meta-cognition as a collection of behavior streams that will affect the system internally over multiple cycles.

Goertzel:

In principle, there should be no strict distinction between meta-cognition and cognition in the Novamente system. All the system's cognitive processes may be represented as Procedure objects in the same format as procedures that the system learns; and ultimately, the system should be able to learn new cognitive processes, and modify its existing ones, using the same techniques it uses for general procedure learning. At the moment, however, for reasons of computational efficiency, many of the system's cognitive processes are represented as C++ code, which prevents them from being adaptively modified as the system grows and learns. It is intended to change this situation in future when a more efficient interpreter of learned (Combo) procedures is implemented.

At the moment, metacognition exists in the system in a more limited way, however. For instance, the heuristics used for pruning of inference processes involve utilization of patterns mined from prior inferences. And, the probabilistic evolutionary learning method used in Novamente (MOSES) looks for patterns across multiple attempts at solving the same problem, and even attempts at solving related problems, thus learning new representations of problem space.

Samsonovich:

The system has access to all its higher-level internal representations: mental states and their parts. Mental states can process other mental states, as they would process sensory information or intended behavioral actions. Some mental states do just this. For example, a mental state "I-Meta" makes sure that the rest of working memory form a consistent working scenario.

Wang:

Certain type of meta-cognition can be achieved as self-perception and self-control with certain internal sensors and effectors. More radical changes to the system design is left for an evolution process, which is separated from the intelligence mechanism that works within the life-cycle of a single system.

- How does **deliberation** occur in your AI system? (Deliberation is taken to refer to activities such as planning, deciding, scheduling, etc. that require "conscious thinking" about some issue.)

Franklin:

Deliberation was implemented in IDA, and will be in LIDA, as a higher-level process operating over multiple cycles by means of behavior streams.

Goertzel:

"Deliberation" in Novamente is directed by (forward or backward chaining) probabilistic inference, supported by other cognitive processes as it finds necessary.

Samsonovich:

As mentioned above, our architecture has a standard procedure for a voluntary action (and most actions that the agent does are voluntary). This procedure includes such elements as generation of ideas (i.e., instances f schemas representing feasible actions in the current situation), selection of those that fit best into the working scenario, making an intent, scheduling and performing its execution, checking results, and so on.

Wang:

Many forms of deliberation occur in NARS as reasoning on operations. In NARS, the conscious thinking and subconscious thinking are carried out by the same mechanism, and their difference is mainly quantitative, not qualitative.

- How does **creativity** occur in your AI system? Specifically, "creativity" is taken to refer to the creation of surprising new concepts or conjectures that were not obviously implicit in the knowledge given to the system.

Franklin:

There's little, if any, creativity in IDA. LIDA should be capable of creativity via perceptual learning, which is done on a generate-and-test basis. Attentional learning may also play a role.

Goertzel:

Creativity occurs in multiple ways, including within the probabilistic inference module (e.g. through inductive and abductive inference), evolutionary learning (finding new solutions to problems posed, and finding new surprising combinations of existing ideas), stochastic pattern mining (finding new surprising combinations...), and blending (exploratory fusion of existing concepts and relationships to form new ones).

Samsonovich:

There is a number of mechanisms of creation of new schemas in our architecture. A process of new schema creation starts with a prototype for a schema (called a "dream" or a "hypothesis"). The prototype may be functional in imagery even at the stage when the schema is not available. One possibility to complete a new schema is to combine existing schemas. When created, a new schema needs to be tested.

Wang:

Creativity occurs in NARS when the derived task/belief/concept does not exists in the system previously. There is no separate mechanism responsible for it, and whether an item is "creative" is sometimes a matter of degree.

- Does your AI system involve any analogue of the human notion of "f**eelings**." If so, what are the similarities and differences?

Franklin:

IDA doesn't learn and has handcrafted motivations in the form of drives attached to the behavior net. No feelings. LIDA, on the other hand, has feelings implemented in the perceptual module that are carried, as part of the common currency, throughout the whole system. In particular, feelings modulate learning and provide motivation.

Goertzel:

Novamente has "internal sensors" that serve as the roots of feelings – e.g. an internal sensor measuring the amount of knowledge recently gained; one measuring the amount of reward given to the system by its teachers; etc. "Feelings" in a more macro-level psychological sense are then broader patterns of activation, triggered by and/or closely associated with these internal sensors.

Samsonovich:

Yes, our architecture has an emotional cognitive map inspired by the human system of emotional values. The primary source for emergence of feelings in our architecture is the reward and punishment system (one of the 8 components). It includes a set of internal stimuli that are associated with selected schemas.

Wang:

In NARS, feeling and emotions comes out of the system's desirability on both external entities/events and internal status/situations. They will influence the internal control and the external communication of the system. As human feelings, they will be experience-grounded and context-sensitive, but since the system's experience will be different from human experience, we cannot expect the system to have the same feeling about a given event or status as a typical human being.

- How will your AI system give rise to a dynamic "**self-model**"? How will this model be represented and what learning methods will create and update it?

Franklin:

IDA has no self-model. In LIDA a self-model can be expected to emerge from autobiographical memory, a part of declarative memory.

Goertzel:

This has not been observed yet in practice, but a robust and complex self-model is expected to emerge in a Novamente system via the combined action of multiple cognitive mechanisms. The system must observe its own actions (e.g. in the AGISim simulation world), and the reactions of other systems to its actions, and construct a model of itself accordingly. Based on this self-model, it directs its ongoing actions, and hence obtains new information with which to improve its self-model. Inference and evolutionary learning both play key roles here, in terms of solving the cognitive problem of guessing "what compact model might describe what I am, in order to explain why I act the way I do." "Self" will not be a single concept represented by a single node, but rather a large set of nodes and links coordinated by a number of overlapping habitual patterns of activation.

Samsonovich:

The self is implemented in our architecture in multiple instances, as mental states. In a sense, each mental state can be called a self-model. The dynamics of mental states conforms to a set of fundamental beliefs (self axioms) that are hard-coded in the driving engine. E.g., the self must be consistent over time. Finally, the personal system of values developed by the cognitive map module is an essential aspect of the Self.

Wang:

The system will have a self-concept, with beliefs about what its status, needs, abilities, and so on. This concept will be handled as the other concepts.