Semantic HTNs: Incorporating Subsymbolic Hierarchical Temporal Networks and Symbolic Representation/Reasoning within Integrative Cognitive Architectures

Ben Goertzel
Novamente LLC, Rockville MD, USA
Fujian Province Key Lab for Brain-Like Intelligent Systems, Dept. of Cognitive Science, Xiamen University

Abstract

One approach to bridging the historic divide between "symbolic" and "subsymbolic" AI is to incorporate a subsymbolic system and a symbolic system into a synergetic integrative cognitive architecture. Here we consider various issues related to incorporating (subsymbolic) Hierarchical Temporal Networks (HTN's, a term introduced as a generalization of Hierarchical Temporal Memory networks) into an integrative cognitive architecture including symbolic aspects. The core conclusion is that for such integration to be meaningful, it must involve dynamic and adaptive linkage and conversion between HTN attractors spanning sensory, motor and goal hierarchies, and analogous representations in the remainder of the integrative architecture. We suggest the mechanism of "semantic HTNs", which maintain the general structure of HTNs but contain more abstract patterns, similar to those represented in more explicitly symbolic AI systems. This notion is made concrete by describing a planned integration of the DeSTIN HTN into the OpenCog integrative cognitive system (which includes a probabilistic-logical symbolic component).

Introduction

The AI field, since its early days, has been marked by a fairly sharp divide between explicitly symbolic methods (e.g. logic or production rules as typically used) and "subsymbolic" methods focused more on numerical pattern recognition (e.g. neural nets as typically used). "Neat versus scruffy" was one attempt to capture this dichotomy; and the identification of "soft computing" as distinct from symbolic AI methods was another. This conceptual divide (however one labels or formalizes it) has been bridged somewhat in recent decades, e.g. by use of neural nets for symbol processing, use of genetic programming to evolve logic expressions, etc. But even today there is no single AI approach capable of handling both symbol manipulation and scalable numerical pattern recognition in a compelling way.

This situation inclines one toward an integrative approach, but there also has been rather little work integrating symbolic and symbolic systems in a deep way that plays to both of their strengths. Work on neural-symbolic computing (Hammer and Hitzler 2007) has mainly focused on using symbolic systems to manipulate patterns emergent from neural nets, a valuable but limited approach. Here we suggest a novel approach to integrating symbolic and subsymbolic AI, via describing a mechanism for linking a powerful subsymbolic architecture (HTNs) into cognitive architectures with significant symbolic components.

The term "Hierarchical Temporal Network" (HTN) is introduced here as an extension of the term "Hierarchical Temporal Memory" (HTM) introduced by Jeff Hawkins to describe his AI architecture (Hawkins and Blakeslee 2006). HTNs, broadly, have a similar architecture to Hawkins' HTM, but may have very different dynamic algorithms and additional features. Current exemplifications of the HTN paradigm, as well as Jeff Hawkins' Numenta "HTM" system (Hawkins and Blakeslee 2006) [1], Itamar Arel's DeSTIN (Arel, Rose, and Coop 2009) and HDRN [2] systems, Dileep George's spin-off from Numenta [3], and work by Mohamad Tarifi (Tarifi, Sitharam, and Ho 2011), Bundzel and Hashimoto (Bundzel and Hashimoto 2010), and others. HTNs are reasonably well proven as an approach to intelligent sensory data processing, and have also been hypothesized as a broader foundation for artificial general intelligence (Goertzel and Pennachin 2005) at the human level and beyond (Hawkins and Blakeslee 2006) (Arel, Rose, and Coop 2009).

The specific form of HTN we will pursue here goes beyond perception processing, and involves the coupling of three separate hierarchies, for perception, action and goals/reinforcement (Goertzel et al. 2010). Abstract learning and self-understanding are then hypothesized as related to systems of attractors emerging from the close dynamic coupling of the upper levels of the three hierarchies.

HTNs embody a certain conceptual model of the nature of intelligence, and to integrate them appropriately with a broader architecture, one must perform the integration not only on the level of software code but also on the level of conceptual models. Here we focus here on the problem of integrating an extended version of the DeSTIN HTN system with the OpenCog integrative AGI (artificial gen-

[1] While the Numenta system is the best-known HTN architecture, other HTNs appear more impressively functional in various respects; and many HTN-related ideas existed in the literature well before Numenta's advent.

[2] `http:\\binatix.com`

[3] `http:\\vicarioussystems.com`

eral intelligence) system. The crux of the issue here is how to map DeSTIN's attractors into OpenCog's more abstract, probabilistic "weighted, labeled hypergraph" representation (called the Atomspace). The main conclusion reached is that in order to perform this mapping in a conceptually satisfactory way, a system of intermediary hierarchies is required. The DeSTIN perceptual hierarchy is augmented by motor and goal hierarchies, leading to a tripartite "extended DeSTIN"; and then three "semantic-perceptual" hierarchies are proposed, corresponding to the three extended-DeSTIN HTN hierarchies and explicitly constituting an intermediate level of representation between attractors in DeSTIN and OpenCog Atoms and Atom-networks. For simple reference we refer to this as the "Semantic HTN" approach.

A "tripartite semantic HTN" consisting of interlinked semantic perceptual, motoric and goal hierarchies could be coupled with DeSTIN or another HTN architecture to form a novel AGI approach; or (our main focus here) it may be used as a glue between an HTN and and a more abstract semantic network such as OpenCog's Atomspace, to form an integrative AGI approach including HTN along with other aspects.

The proposed "tightly linked integration" approach has significant implications in the context of the concept of "cognitive synergy" (Goertzel 2009), which hypothesizes particular properties that the interactions between components in an integrated AGI system should display, in order for the overall system to display significant general intelligence using limited computational resources. Simply piping output from an HTN to other components, and issuing control signals from these components to the HTN, is likely an inadequate mode of integration, incapable of leveraging the full potential of HTNs; what we are suggesting here is a much tighter and more synergetic integration. A previous paper (Goertzel and Duong 2009) explored integration of OpenCog with a neural net perception system; we believe the present approach enables tighter integration with more synergetic potential.

This is a conceptual paper and at time of writing (July 2011), most of the ideas described have not been implemented yet. Among other applications, OpenCog is currently being used to control virtual agents in virtual worlds (Goertzel et al. 2011), an application that does not require the kind of fine-grained perception at which HTNs currently excel. DeSTIN has recently been ported to CUDA, and is now being tried on robot vision data for the first time (although HDRN, a somewhat similar proprietary system built by Binatix Inc. has been used on such data for several years already). Assuming funding and other logistics permit, our intention is to pursue the hybridization discussed here in steps, beginning with the semantic-perceptual HTN, after we have made DeSTIN-CUDA more robust via enhancing some of its learning and memory mechanisms. However, we believe the ideas presented here have general value beyond any specific implementation, and indeed beyond any particular HTN or integrative cognitive system.

DeSTIN

DeSTIN [4] is an HTN architecture, aimed ultimately at human-level AGI. The general design has been described as comprising three crosslinked hierarchies, handling perception, action and reinforcement. However, so far only the perceptual hierarchy (also called the "spatiotemporal inference network") has been implemented or described in detail in publications. Here we will consider an "extended DeSTIN" consisting of all three hierarchies, with the caveat that our extension of DeSTIN to a tripartite system may differ in detail from the tripartite version intended by DeSTIN's creators.

The hierarchical architecture of DeSTIN's spatiotemporal inference network comprises an arrangement into multiple layers of "nodes" comprising multiple instantiations of an identical processing unit. 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).

The belief state is a probability mass function over the sequences of stimuli that the nodes learns to represent. DeSTIN's basic belief update rule governs the learning process and is identical for every node in the architecture, and is given in (Arel, Rose, and Coop 2009). Based on this rule, 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.

There are many possible ways to extend DeSTIN beyond perception into a tripartite system; here I will present one possibility that I call "extended DeSTIN" [5].

One may envision an extended-DeSTIN action hierarchy having a similar hierarchical structure to the perception hierarchy, but where each node contains a probability mass function over sequences of actions, rather than sequences of observations. The actions in the bottom-level nodes correspond to specific movements (e.g. in a robotics context, specific sequences of servomotor commands), and the actions in a higher-level node correspond to serial and/or parallel combinations of the actions in the child nodes. Rather than

[4] Some sentences in this section were pasted with minor modifications from (Goertzel et al. 2010), coauthored by Ben Goertzel, Itamar Arel and others

[5] barely resisting the urge to label it "DeSTIN-E"!

corresponding strictly to a partitioning of 2D or 3D space, the nodes in the action hierarchy are best understood to correspond to regions in configuration space (of the underlying system controlled by DeSTIN, e.g. a robot), where the region of configuration space referred to by a child node is a subspace of the region of configuration space referred to by the parent node. For instance, a node containing sequences of arm movements might have child nodes containing sequences of shoulder, elbow, wrist and finger movements respectively.

Similarly, one may envision an extended DeSTIN goal hierarchy, in which each node contains a cluster of sequences of percepts and/or actions, where each sequence terminates in a reward signal. Reward signals are percepts from the external world, or the system's wired-in motivational structure (e.g. the system could be intrinsically rewarded for learning new information according to some mathematical measure, or for keeping its robotic body's electrical power within reasonable bounds, etc.). Sequences in nodes below the top-level will consist of relatively localized perceptions that, when correlated with certain actions, have been found to lead to reward. This is subtler than standard reinforcement-learning hierarchies such as (**?**) where each node contains a reinforcement function that's a linear combination of the ones in its children.

The system's overall activity is then governed by a traditional reinforcement learning approach, where the goal is to maximize the weighted expected reward over the system's future. The hierarchical structure enables more sophisticated subgoaling than in traditional RL architectures.

The obvious question that comes to mind, when considering an architecture like this as a general AGI architecture, is how such phenomena as mathematical reasoning or recursive phrase-structure grammar might emerge from the network via the system's world-interactions. The general answer proposed for this is that it involves complex attractors ensuing from the three-way interactions between the perceptual, motor and goal hierarchies. While this is a plausible hypothesis, it is far from convincingly demonstrated. Further, in spite of some of Hawkins' (Hawkins and Blakeslee 2006) arguments, the analogy to neuroscience is not a strong argument for the AGI potential of HTNs, because at best HTNs are a model of a specific pattern of neural organization that occurs only in a moderate-sized subset of the brain. There is a fair analogy between HTN and the hierarchical structure of the visual cortex, but many other portions of the brain lack this marked hierarchy, and many portions of the cognitive cortex owe as much to the primarily combinatory connectivity pattern of olfactory cortex as to the predominantly hierarchical connectivity pattern of visual and auditory cortex.

We agree that HTNs may be a viable route to human-level AGI on their own, but presently are more concerned to hybridize them with other AGI paradigms.

OpenCog

Now we briefly review OpenCog – an open-source AGI software framework, which has been used for various practical applications, and also for implementation of the OpenCog-Prime design aimed ultimately toward AGI at the human level and beyond. OpenCog has been used for commercial applications in the area of natural language processing and data mining; e.g. see (Goertzel et al. 2006). It has also been used to control virtual agents in virtual worlds, at first using an OpenCog variant called the OpenPetBrain (Goertzel and Et Al 2008), and more recently in a more general way using a Minecraft-like virtual environment (Goertzel et al. 2011).

Conceptually founded on the "patternist" systems theory of intelligence outlined in (Goertzel 2006), OpenCog-Prime combines multiple AI paradigms such as uncertain logic, computational linguistics, evolutionary program learning and connectionist attention allocation in a unified architecture. Cognitive processes embodying these different paradigms interoperate together on a common neural-symbolic knowledge store called the Atomspace. The interaction of these processes is designed to encourage the self-organizing emergence of high-level network structures in the Atomspace, including superposed hierarchical and heterarchical knowledge networks, and a self-model network enabling meta-knowledge and meta-learning.

OCP relies on multiple memory types (all intersecting via the AtomSpace, even when also involving specialized representations), including the declarative, procedural, sensory, and episodic memory types that are widely discussed in cognitive neuroscience (Tulving and Craik 2005), plus attentional memory for allocating system resources generically, and intentional memory for allocating system resources in a goal-directed way. Declarative memory is addressed via probabilistic inference; procedural memory via probabilistic evolutionary program learning; episodic memory via simulation; intentional memory via a largely declarative goal system; attentional memory via an economics-based dynamical system similar to an attractor neural network. Sensorimotor memory is not handled thoroughly within OCP itself, part of the reason for integrating an HTN into OpenCog.

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, the operative hypothesis being that this will yield cooperative intelligence going beyond what could be achieved by an architecture merely containing the same structures and processes in separate "black boxes." That is, 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: so-called **cognitive synergy**

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 "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. In a robot or virtual "AGI school" context, for example, the top-level goals are simple things such as pleasing the teacher, learning new information and skills, and protecting the agent's body.

Integrating HTNs with Other AI Frameworks

HTNs represent knowledge as attractor patterns spanning multiple levels of hierarchical networks, supported by nonlinear dynamics and (at least in the case of the overall DeSTIN design) involving cooperative activity of perceptual, motor and control networks. These attractors are learned and adapted via a combination of methods including localized pattern recognition algorithms and probabilistic inference. Other AGI paradigms represent and learn knowledge in a host of other ways. How then can HTNs be integrated with these other paradigms?

A very simple form of integration, obviously, would be to use an HTN as a sensorimotor cortex for another AI system that's focused on more abstract cognition. In this approach, the HTN would stream state-vectors to the abstract cognitive system, and the abstract cognitive system would stream abstract cognitive inputs to the HTN (which would then consider them together with its other inputs). One thing missing in this approach is the possibility of the abstract cognitive system's insights biasing the judgments inside the HTN. Also, abstract cognition systems aren't usually well prepared to handle a stream of quantitative state vectors (even ones representing intelligent compressions of raw data).

An alternate approach is to build a richer intermediate layer, which in effect translates between the internal language of the HTN and the internal language of the other AI system involved. The particulars, and the viability, of this will depend on the particulars of the other AI system. What we'll consider here is the case where the other AI system contains explicit symbolic representations of patterns (including patterns abstracted from observations that may have no relation to its prior knowledge or any linguistic terms). In this case, we suggest, a viable approach may be to construct a "semantic HTN" to serve as an intermediary. The semantic HTN has the same hierarchical structure as an HTN, but inside each node it contains abstract patterns rather than numerical vectors. This approach has several potential major advantages: the other AI system is not presented with a large volume of numerical vectors (which it may be unprepared to deal with effectively); the HTN can be guided by the other AI system, without needing to understand symbolic control signals; and the intermediary semantic HTN can serve as a sort of "blackboard" which the HTN and the other AI system can update in parallel, and be guided by in parallel, thus providing a platform encouraging "cognitive synergy".

The remainder of this paper goes into more detail on the concept of semantic HTNs. The discussion mainly concerns the specific context of DeSTIN/OpenCog integration, but the core ideas would apply to the integration of any HTN architecture with any other AI architecture involving uncertain symbolic representations susceptible to online learning.

Semantic HTN for Perception Processing

In the standard perceptual HTN hierarchy, a node N on level k (considering level 1 as the bottom) corresponds to a spatiotemporal region S with size s_k (s_k increasing monotonically and usually exponentially with k); and, has children on level $k-1$ corresponding to spatiotemporal regions that collectively partition S. For example, a node on level 3 might correspond to a $16x16$ pixel region S of 2D space over a time period of 10 seconds, and might have 4 level 2 children corresponding to disjoint $4x4$ regions of 2D space over 10 seconds, collectively composing S.

This kind of hierarchy is very effective for recognizing certain types of visual patterns. However it is cumbersome for recognizing some other types of patterns, e.g. the pattern that a face typically contains two eyes beside each other, but at variable distance from each other.

One way to remedy this deficiency is to extend the definition of the hierarchy, so that nodes do not refer to fixed spatial or temporal positions, but only to *relative* positions. In this approach, the internals of a node are basically the same as in an HTN, and the correspondence of the nodes on level k with regions of size s_k is retained, but the relationships between the nodes are quite different. For instance, a variable-position node of this sort could contain several possible 2D pictures of an eye, but be nonspecific about where the eye is located in the 2D input image.

Figure 1 depicts this "semantic-perceptual HTN" idea heuristically, showing part of a semantic-perceptual HTN indicating the parts of a face, and also the connections between the semantic-perceptual HTN, a standard perceptual HTN, and a higher-level cognitive semantic network like OpenCog's Atomspace. [6]

More formally, in the suggested "semantic-perceptual HTN" approach, a node N on level k, instead of pointing to a set of level $k-1$ children, points to a small (but not necessarily connected) *semantic network* , such that the nodes of the semantic network are (variable-position) level $k-1$ nodes; and the edges of the semantic network possess labels representing spatial or temporal relationships, for example *horizontally_aligned*, *vertically_aligned*, *right_side*, *left_side*, *above*, *behind*, *immediately_right*, *immediately_left*, *immediately_above*, *immediately_below*, *after*, *immediately_after*. The edges may also be weighted either with numbers or probability distributions, indicating the quantitative weight of the relationship indicated by the label.

So for example, a level 3 node could have a child network

[6] The perceptual HTN shown is unrealistically small for complex vision processing (only 4 layers), and only a fragment of the semantic-perceptual HTN is shown (a node corresponding to the category face, and then a child network containing nodes corresponding to several components of a typical face). In a real semantic-perceptual HTN, there would be many other nodes on the same level as the face node, many other parts to the face subnetwork besides the eyes, nose and mouth depicted here; the eye, nose and mouth nodes would also have child subnetworks; there would be link from each semantic node to centroids within a large number of perceptual nodes; and there would also be many nodes not corresponding clearly to any single English language concept like eye, nose, face, etc.

of the form horizontally_aligned(N_1, N_2) where N_1 and N_2 are variable-position level 2 nodes. This would mean that N_1 and N_2 are along the same horizontal axis in the 2D input but don't need to be immediately next to each other. Or one could say, e.g. on_axis_perpendicular_to(N_1, N_2, N_3, N_4), meaning that N_1 and N_2 are on an axis perpendicular to the axis between N_3 and N_4. It may be that the latter sort of relationship is fundamentally better in some cases, because $horizontally\_aligned$ is still tied to a specific orientation in an absolute space, whereas $on\_axis\_perpendicular\_to$ is fully relative. But it may be that both sorts of relationship are useful.

Next, development of learning algorithms for semantic HTNs is an open problem, but we do have some basic ideas about how it may be done (to be refined through practical exploration). First of all, it would seem that, for instance, the DeSTIN learning algorithms could straightforwardly be utilized in the semantic HTN case, once the local semantic networks involved in the network are known. So at least for some HTN designs, the problem of learning the semantic networks may be decoupled somewhat from the learning occurring inside the nodes. DeSTIN nodes deal with clustering of their inputs, and calculation of probabilities based on these clusters (and based on the parent node states). The difference between the semantic HTN and the traditional DeSTIN HTN has to do with what the inputs are.

Regarding learning the local semantic networks, one relatively straightforward approach would be to *data mine them from a standard HTN*. That is, if one runs a standard HTN on a stream of inputs, one can then run a frequent pattern mining algorithm to find semantic networks (using a given vocabulary of semantic relationships) that occur frequently in the HTN as it processes input. A subnetwork that is identified via this sort of mining, can then be grouped together in the semantic HTN, and a parent node can be created and pointed to it.

Also, the standard HTN can be searched for frequent patterns involving the clusters (referring to DeSTIN here, where the nodes contain clusters of input sequences) inside the nodes in the semantic HTN. Thus, in the "semantic DeSTIN" case, we have a feedback interaction wherein: 1) the standard HTN is formed via processing input; 2) frequent pattern mining on the standard HTN is used to create subnetworks and corresponding parent nodes in the semantic HTN; 3) the newly created nodes in the semantic HTN get their internal clusters updated via standard DeSTIN dynamics; 4) the clusters in the semantic nodes are used as seeds for frequent pattern mining on the standard HTN, returning us to Step 2 above.

After the semantic HTN is formed via mining the perceptual HTN, it may be used to bias the further processing of the perceptual HTN. For instance, in DeSTIN each node carries out probabilistic calculations involving knowledge of the prior probability of the "observation" coming into that node over a given interval of time. In the current DeSTIN version, this prior probability is drawn from a uniform distribution, but it would be more effective to draw the prior probability from the semantic network – observations matching things represented in the semantic network would get a higher prior probability. One could also use subtler strategies such as using imprecise probabilities in DeSTIN (Goertzel 2011), and assigning a greater confidence to probabilities involving observations contained in the semantic network.

Finally, we note that the nodes and networks in the semantic HTN may naturally be linked into the nodes and links in a semantic network such as OpenCog's AtomSpace. This allows us to think of the semantic HTN as a kind of bridge between the standard HTN and the cognitive layer of an AI system. In an advanced implementation, the cognitive network may be used to suggest new relationships between nodes in the semantic HTN, based on knowledge gained via inference or language.

Semantic HTN for Motor and Sensorimotor Processing

Next we consider a semantic HTN that focuses on movement rather than sensation. In this case, rather than a 2D or 3D visual space, one is dealing with an n-dimensional *configuration space* (C-space). This space has one dimension for each degree of freedom of the agent in question. The more joints with more freedom of movement an agent has, the higher the dimensionality of its configuration space.

Using the notion of configuration space, one can construct a *semantic-motoric HTN hierarchy* analogous to the semantic-perceptual HTN hierarchy. However, if one does this using a completely naive approach, one finds a significant problem – the curse of dimensionality. A square of side 2 can be tiled with 4 squares of side 1, but a 50-dimensional cube of side 2 can be tiled with 2^{50} 50-dimensional cubes of side 1. Clearly, if one is to build a hierarchy in configuration space analogous to the HTN hierarchy in perceptual space, some sort of sparse hierarchy is necessary.

There are many ways to build a sparse hierarchy of this nature, but one simple approach is to build a hierarchy where the nodes on level k represent motions that combine the motions represented by nodes on level $k-1$. In this case the most natural semantic label predicates would seem to be things like *simultaneously*, *after*, *immediately_after*, etc. So a level k node represents a sort of "motion plan" corresponded by chaining together (serially and/or in parallel) the motions encoded in level $k-1$ nodes. Overlapping regions of C-space correspond to different complex movements that share some of the same component movements, e.g. if one is trying to slap one person while elbowing another, or run while kicking a soccer ball forwards. Also note, the semantic HTN approach reveals perception and motor control to have essentially similar hierarchical structures, more so than with the traditional HTN approach and its fixed-position perceptual nodes.

Just as the semantic-perceptual HTN is naturally aligned with a traditional perceptual HTN, similarly a semantic-motoric HTN may be naturally aligned with a "motor HTN". A typical motoric hierarchy in robotics might contain a node corresponding to a robot arm, with children corresponding to the hand, upper arm and lower arm; the hand node might then contain child nodes corresponding to each finger, etc. This sort of hierarchy is intrinsically spatiotemporal because

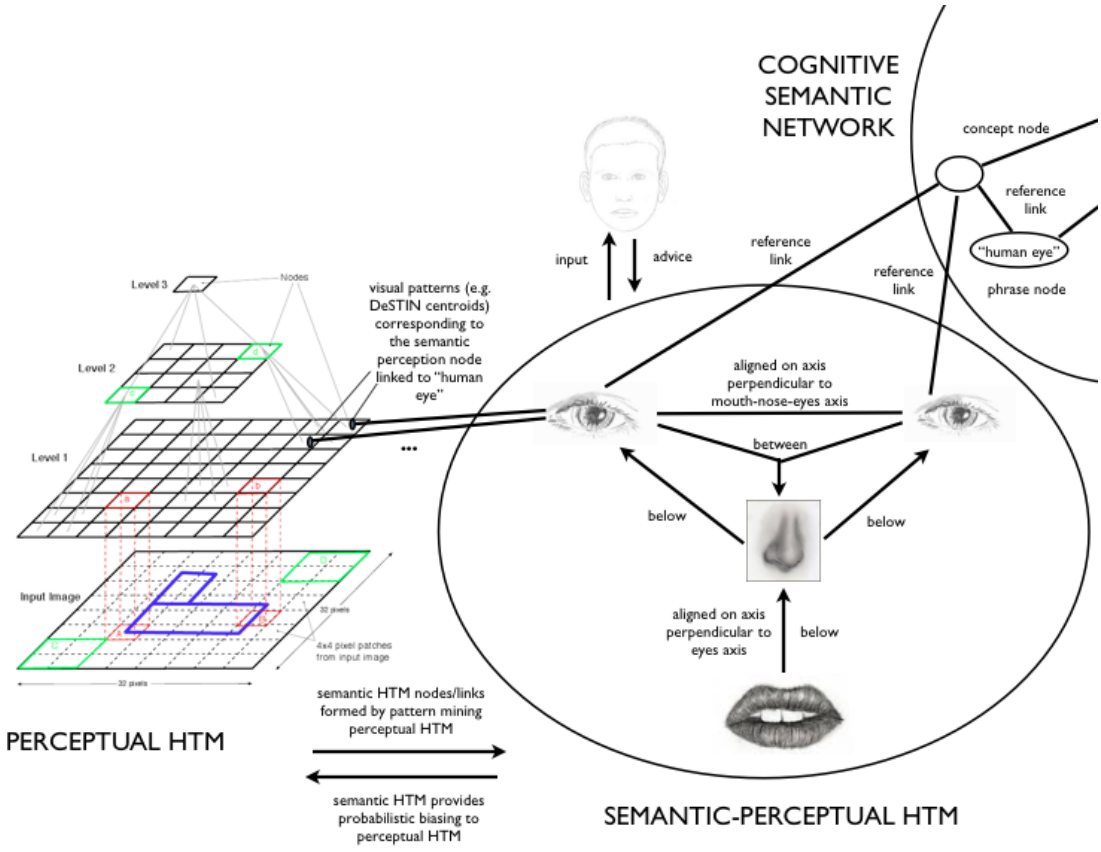

Figure 1: Simplified depiction of the relationship between a semantic-perceptual HTN, a traditional perceptual HTN (like DeSTIN), and a cognitive semantic network (like OpenCog's AtomSpace).

each individual action of each joint of an actuator like an arm is intrinsically bounded in space and time. Perhaps the most ambitious attempt along these lines is (Albus and Meystel 2001), which shows how perceptual and motoric hierarchies are constructed and aligned in an architecture for intelligent automated vehicle control.

Figure 2 gives a simplified illustration of the potential alignment between a semantic-motoric HTN and a purely motoric hierarchy (like the one posited above in the context of extended DeSTIN). [7] In the figure, the motoric hierarchy is assumed to operate somewhat like DeSTIN, with nodes corresponding to (at the lowest level) individual servomotors, and (on higher levels) natural groupings of servomotors. The node corresponding to a set of servos is assumed to contain centroids of clusters of trajectories through configuration space. The task of choosing an appropriate action is then executed by finding the appropriate centroids for the nodes. Note an asymmetry between perception and action here. In perception the basic flow is bottom-up, with top-down flow used for modulation and for "imaginative" generation of percepts. In action, the basic flow is top-down, with bottom-up flow used for modulation and for imaginative, "fiddling around" style generation of actions. The semantic-motoric hierarchy then contains abstractions of the C-space centroids from the motoric hierarchy – i.e., actions that bind together different C-space trajectories that correspond to the same fundamental action carried out in different contexts or under different constraints. Similarly to in the perceptual case, the semantic hierarchy here serves as a glue between lower-level function and higher-level cognitive semantics.

Connecting the Perceptual and Motoric Hierarchies with a Goal Hierarchy

Finally, we turn to the notion of connecting perceptual and motoric hierarchies directly (rather than, say, via means of a separate cognitive network). One way to do this is using a "semantic-goal HTN" bridging the semantic-perceptual and semantic-motoric HTNs. The semantic-goal HTN would be

[7] In the figure, only a fragment of the semantic-motoric HTN is shown (a node corresponding to the "get object" action category, and then a child network containing nodes corresponding to several components of the action). In a real semantic-motoric HTN, there would be many other nodes on the same level as the get-object node, many other parts to the get-object subnetwork besides the ones depicted here; the subnetwork nodes would also have child subnetworks; there would be link from each semantic node to centroids within a large number of motoric nodes; and there might also be many nodes not corresponding clearly to any single English language concept like "grasp object" etc.

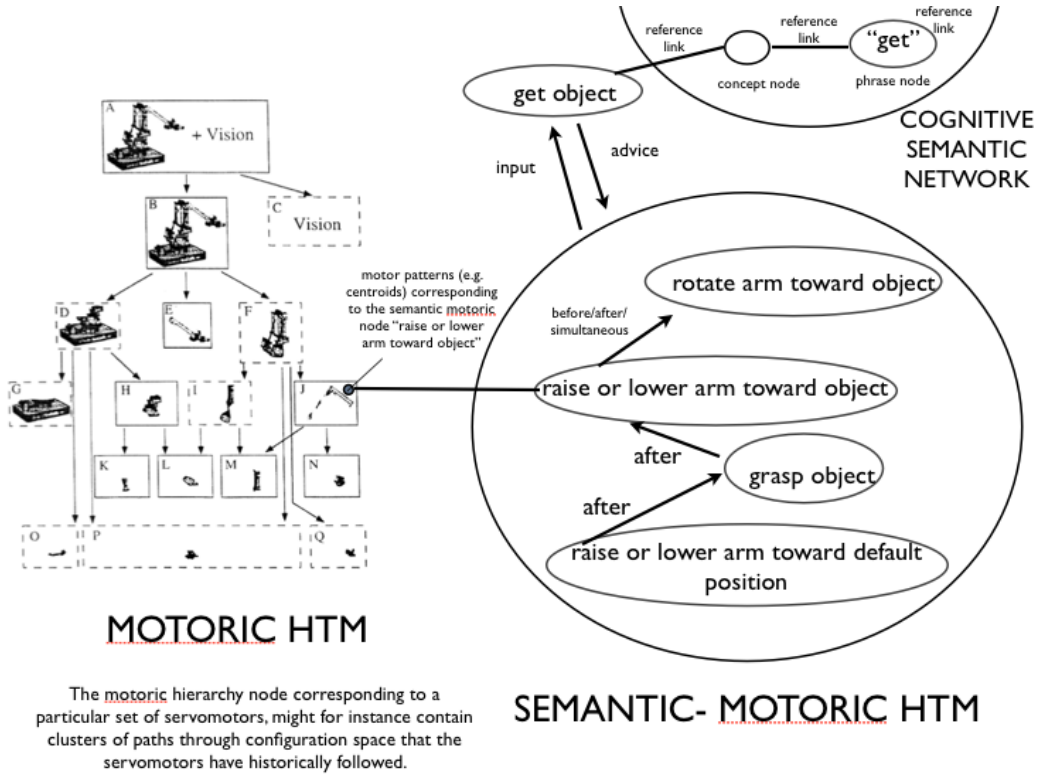

Figure 2: Simplified depiction of the relationship between a semantic-motoric HTN, a motor control hierarchy (illustrated by the hierarchy of servos associated with a robot arm), and a cognitive semantic network (like OpenCog's AtomSpace).

a "semantic HTN" loosely analogous to the perceptual and motor semantic HTNs – and could optionally be linked into the reinforcement hierarchy of a tripartite HTN like extended DeSTIN. Each node in the semantic-goal HTN would contain implications of the form "Context & Procedure → Goal", where Goal is one of the AI system's overall goals or a subgoal thereof, and Context and Procedure refer to nodes in the perceptual and motoric semantic HTNs respectively.

For instance, a semantic-goal HTN node might contain an implication of the form "I perceive my hand is near object X & I grasp object X → I possess object X." This would be useful if "I possess object X" were a subgoal of some higher-level system goal, e.g. if X were a food object and the system had the higher-level goal of obtaining food.

To the extent that the system's goals can be decomposed into hierarchies of progressively more and more spatiotemporally localized subgoals, this sort of hierarchy will make sense, leading to a tripartite hierarchy as loosely depicted in Figure 3. [8] One could attempt to construct an overall AGI approach based on a tripartite hierarchy of this nature, counting on the upper levels of the three hierarchies to come together dynamically to form an integrated cognitive network, yielding abstract phenomena like language, self, reasoning and mathematics. On the other hand, one may view this sort of hierarchy as a portion of a larger integrative AGI architecture, containing a separate cognitive network, with a less rigidly hierarchical structure and less of a tie to the spatiotemporal structure of physical reality. The latter view is the one we are primarily taking within the OpenCog AGI approach, viewing perceptual, motoric and goal hierarchies as "lower level" subsystems connected to a "higher level" system based on the OpenCog AtomSpace and centered on its abstract cognitive processes.

Learning of the subgoals and implications in the goal hierarchy is of course a complex matter, which may be addressed via a variety of algorithms, including online clustering (for subgoals or implications) or supervised learning (for implications, the "supervision" being purely internal and provided by goal or subgoal achievement).

Acknowledgement: the ideas given here emerged via conversations with Itamar Arel, Shuo Chen and Michel Drenthe.

[8] The diagram is simplified in many ways, e.g. only a handful of nodes in each hierarchy is shown (rather than the whole hierarchy), and lines without arrows are used to indicate bidirectional arrows, and nearly all links are omitted. The purpose is just to show the general character of interaction between the components in a simplified context.

References

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

Arel, I.; Rose, D.; and Coop, R. 2009. Destin: A scalable deep learning architecture with application to high-

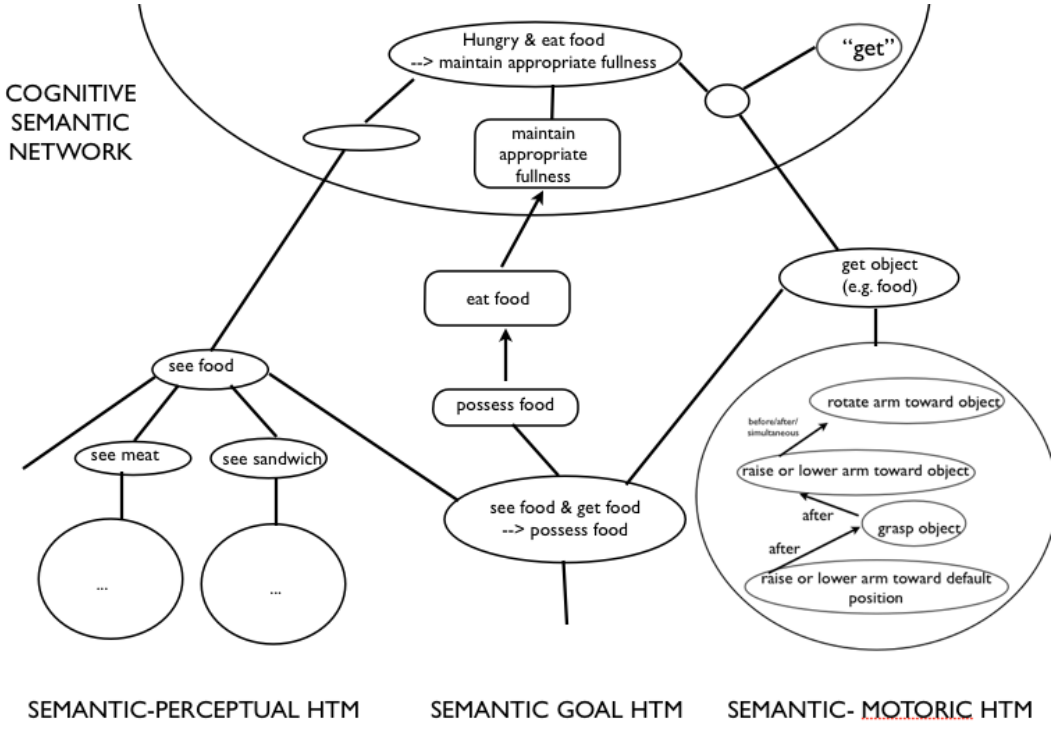

Figure 3: Simplified illustration of the proposed interoperation of perceptual, motoric and goal semantic HTNs.

dimensional robust pattern recognition. Proc. AAAI Workshop on Biologically Inspired Cognitive Architectures.

Bundzel, and Hashimoto. 2010. Object identification in dynamic images based on the memory-prediction theory of brain function. Journal of Intelligent Learning Systems and Applications *2-4*.

Goertzel, B., and Duong, D. 2009. Opencog ns: An extensible, integrative architecture for intelligent humanoid robotics. In Proc. of BICA 2009.

Goertzel, B., and Et Al, C. P. 2008. An integrative methodology for teaching embodied non-linguistic agents, applied to virtual animals in second life. In Proc.of the First Conf. on AGI. *IOS Press.*

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

Goertzel, B.; Pinto, H.; Pennachin, C.; and Goertzel, I. F. 2006. Using dependency parsing and probabilistic inference to extract relationships between genes, proteins and malignancies implicit among multiple biomedical research abstracts. In Proc. of Bio-NLP 2006.

Goertzel, B.; Lian, R.; DeGaris, H.; Chen, S.; and Arel, I. 2010. World survey of artificial brains part 2: Biologically integrative cognitive architectures. Neurocomputing.

Goertzel, B.; Pitt, J.; Cai, Z.; Wigmore, J.; Huang, D.; Geisweiller, N.; Lian, R.; and Yu, G. 2011. Integrative general intelligence for controlling game ai in a minecraft-like environment. In Proc. of BICA 2011.

Goertzel, B. 2006. The Hidden Pattern. *Brown Walker.*

Goertzel, B. 2009. Opencog prime: A cognitive synergy based architecture for embodied artificial general intelligence. In ICCI 2009, Hong Kong.

Goertzel, B. 2011. Imprecise probability as a linking mechanism between deep learning, symbolic cognition and local feature detection in vision processing. In Proc. of AGI-11.

Hammer, B., and Hitzler, P., eds. 2007. Perspectives of Neural-Symbolic Integration. Studies in Computational Intelligence, Vol. 77. *Springer.*

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

Tarifi, M.; Sitharam, M.; and Ho, J. 2011. Learning hierarchical sparse representations using iterative dictionary learning and dimension reduction. In Proc. of BICA 2011.

Tulving, E., and Craik, R. 2005. The Oxford Handbook of Memory. *Oxford U. Press.*