Distributed Representations and “Flexible” Modularity in Hybrid Architectures

Giovanni Pezzulo (giovanni.pezzulo@istc.cnr.it), Gianguglielmo Calvi (gianguglielmo.calvi@noze.it)
Istituto di Scienze e Tecnologie della Cognizione - CNR, via S. Martino della Battaglia, 44
00185 Roma, Italy

Abstract

Here we discuss the role of modules and representations into cognitive architectures by comparing the “unified” approach of SOAR and ACT-R with the “decentralized” one of the Society of Mind. We introduce AKIRA, an open-source hybrid architecture and show how to exploit its features, namely distributed representations and parallel, concurrent processing for agent based cognitive modeling.

Keywords: hybrid architectures; distributed representations.

Introduction

Cognitive architectures are claimed to be central for cognitive science: Newell (1990) introduces the “Unified” approach to cognition, that consists in integrating many cognitive capabilities instead of building models of limited aspects of cognition; in a similar way, Sloman (1999) argues for the opportunity of designing systems at the architectural level instead of assembling single, specialized components. The most notable examples are the generic, unified architectures SOAR (Rosenbloom et al., 1992) and ACT-R (Anderson et al., 1998); some other ones, such as the DUAL/AMBR (Kokinov, 1994) and Copycat (Hofstadter et al., 1994) models of analogy, do not address the generality of the cognitive processes, but try to capture the general underlying principles of high-level cognition.

In the following Sections we discuss the role of modules and representations in cognitive architectures. We introduce the architectural scheme of the Society of Mind (Minsky, 1986), that exploits “vertical” modules and distributed representations, showing how it realizes dynamic, context dependent computation, and why it is relevant for cognitive modeling. We also introduce AKIRA, an hybrid architecture mixing up Multi-Agent and Pandemonium (Jackson, 1987) features; agents (called Daemons) are managed by a server process (called Pandemonium). Differently from standard Multi-Agent architectures, Daemons are related to a central resource, the Energy Pool, and spread activation via an Energetic Network. These structures afford connectionist dynamics at the agent level; they permit to realize parallel, dynamic and emergent computation by distributing the operations between many simple, interacting processes, giving high versatility to modularization. Moreover, representations can be distributed among many partially active and partially available units.

Designing at the Architectural Level

Sloman (1999) furnishes strong arguments for thinking and designing models at the architecture level, rather than building single, independent cognitive functionalities. With respect to this objective, there are two strategies: the first one is well represented by some of the most known cognitive architectures, ACT-R (Anderson et al., 1998) and SOAR (Rosenbloom et al., 1992), implementing the “Unified Theory of Cognition” where a small set of mechanisms (e.g. production rules) are exploited as the substrate of all the cognitive functionalities.

The second possible strategy is more liberal: the architecture only provides a functional, high level “blueprint” of the relations and interdependencies of its parts; each one can be developed independently, providing that it matches the architectural constraints, and can exploit different, specialized mechanisms instead of a single, general-purpose one. However, there exist some underlying principles (e.g. concurrence between the components) that are shared by all the processes and are supposed to constraint all high-level cognition. This approach is exemplified by the Society of Mind model, where a number of narrow-minded, specialized agents interact and compete into the same environment. Cooperation and coordination are an emergent property of the system: each agent learns how to exploit other’s capabilities and activity for its purposes. The challenge is thus having many processes and functions working together without having a common ontology or a single computational mechanism.

Modularization

The second kind of architecture is more flexible even in terms of modules and hierarchies. Fodor (1981) asserts the relevance of modules into cognitive architectures; both contents and processes in a module are “opaque” to the other components. Each module influences the others only through its output (e.g. for an hypothetical “vision module”, it could be a symbolic representation): it is impossible to interact with the “private” processes of each module.

Sometimes (e.g. in ACT-R) modules also introduce a “serial bottleneck” into the processing: while many operations can be performed in parallel into a module, a single process is selected to be active for each moment. This introduces another rigidity element and makes it even more crucial the choice of which modules to implement and which is the format of the representations.

Of course, given the complexity of a cognitive system, a certain amount of modularization is mandatory; however, how to modularize is a complex design choice that strongly influences the kind of capabilities that can be obtained (for a review of modularization in AI systems see Bryson, 2004). Fodor argues in favor of “horizontal modularity”, i.e. implementing as modules the main cognitive processes such as perception, attention and memory. In this spirit, some architectures such as ACT-R exploit impenetrable perceptual-motor and memory modules. While inside each module there can be interaction and competition between the representations and the operations, there is less possibility for interactions between modules. Blendings, multi-modal interactions and other contextual and interference effects between modules are of course possible, but only at a coarse-grained time rate and only exploiting a certain format of the representations (for example, symbolic input and output). Moreover, each module has its private memory space and its computational resources; the processes in the different modules are independent and no module assumes as contextual parameters the current activity of the other ones. It remains an open issue if this modularization results to be too rigid: in fact, it could be claimed that in a cognitive architecture each process should have place thanks to, and in the context of, each other, at least to a certain extent.

On the contrary, more flexible and distributed cognitive architectures can take advantage *of modulating the modules*, i.e. having modules interfering with the content and the processing of other ones. This feature can be exploited e.g. for meta-reasoning, where some processes are supposed to have (at least a certain) access to the content and the processes of other modules. It can be also considered a general architectural principle; Cassimatis (2002) argues that *to an extent impossible to modular systems, inference schemes must share progress, exploit unforeseen opportunities, interrupt each other without hazard and be responsive to world knowledge.*

Knowledge in a module can be also used as *the context* of another one in order e.g. to reduce the problem or search space. In other cases a module or process has knowledge that is useful for another one; for example, in order to trace an object moving behind an obstacle, the attentional module should exploit knowledge produced by an hypothetical "ingenuous physics simulation" process.

One of the central requirements for adding versatility to the modules is synchronizing some of their representation (share progress); in a sense it means having a "deictic" representation of the context. This can be done by using shared variables, but this strongly constrains the content of all the modules, since a common ontology is needed; or by using an external element as a medium. A major claim of distributed cognition (Clark, 1997) is that the environment itself is such a medium: instead of using internal representations the processes can attune themselves to the environment, using it as an external memory.

A similar method is exploited by the Pandemonium, that can have a "common memory and work space" (e.g. a Blackboard): when they perform their operations (e.g. matching a pattern), the Daemons notify to the Blackboard (by "shrieking") that they are active. This notification can be accessed by other Daemons that can synchronize their representations (e.g. be aware that such a pattern is matched in the context). The Daemons have neither to share a common ontology, nor exchange explicit messages or share variables, but only learn to be sensible to the same information in the environment (such as "*Daemon x is active now*"). The same information can be of course interpreted in different ways by different Daemons.

There is another aspect of "modulating the modules": having processes interfering with the computational resources of other modules, e.g. assigning them more or less priority, activation or communication bandwidth. This allows developers to control the computational dynamics at the low level: e.g. representing *alarms* (Sloman, 1999) as urgent danger signals that stop all the computation; or the *attention focus* (see Baars, 1988). Resource management is strongly related to another kind of modularity exploitable by cognitive architectures: *modularity by time sharing*: the same resources are available to different processes, that can exploit them with different timings. For example, two concurrent modules can exploit information encoded in the same cortical area (there is evidence in neurobiological literature, see Bryson, 2004). The representations have thus different semantics in different times and contexts, depending from the function that exploits them: even if the representation is the same, it *means* and *is used for* different things. This has a more important consequence from the design point of view: some processes are mutually exclusive, because they exploit the same representation space; this is in contrast with the general assumption that all the modules should be always active. Instead of being interpreted only as a limitation to expressive power, this feature should be exploited for cognitive modeling. For example, SOAR and ACT-R have introduced *problem spaces*: each function can access only a part of the resources and representations of the system. Problem spaces actually serve as "framing" of situations in order to capture its context and to reduce computational load. Minsky (in preparation) argues that special agents called *selectors* are responsible for activating (only) a set of resources in response to a given context; this is even the ultimate role of emotions. Task specific agents (i.e. "vertical modules" in Fodor terms) can thus be recruited and made available to different extents in a context dependent way.

Last, the most crippling limitation of rigid (and horizontal) modular systems is that it is often necessary to explicitly pre-plan all their interactions and behaviors, thus it is difficult to design them. On the contrary, it is becoming popular (e.g. in behavior-based robotics, Brooks, 1991; Maes, 1990) to run a number of semi-independent, concurrent behavioral components and let them dynamically interact without a fixed control cycle or a central interpreter: a central theme of the Society of Mind model.

It remains to be explored how much the modules have to be "flexible" and "penetrable" by other process; total penetrability is the opposite of the concept of module, but here we have individuated a number of reasons for designing some interactions (e.g. influencing the priority of their processes, letting them compete for resources, or synchronizing some inner states). Here we introduce a different approach to modularization that is well suited for this kind of fine-grained interactions.

The Society of Mind (SOM)

The SOM is composed of many *agencies* that contain many processes called *agents*. Agencies are different from Fodor-like modules: first of all, they are specialized processes for specific tasks (e.g. to build a bridge, to sum up, etc.) rather than large cognitive capabilities such as perception and reasoning. This is a different way of segmenting the cognitive functionalities: in Fodor terms they are "vertical" rather than "horizontal" modules. Moreover, they couple perceptual and motor elements; they are more flexible, tightly interconnected and less impenetrable; they do not communicate via an explicit inter-lingua (such as the "mentalese"), they do not share a common ontology: on the contrary, many of their interactions are simple activation exchange, or they can synchronize their representations by observing and reacting to the same event. They thus massively exploit *implicit communication* (Castelfranchi, 2004), an important feature even in Pandemonium (Jackson, 1987). Another crucial point in the SOM is that the organization and the control management is fully distributed: there is no central interpreter, and arbitrations within the same agency and between agencies are resolved on-line. Agents and agencies compete for activation, exploit the activity of each other for their own purposes (e.g. the agency *going home* exploits the activity of *sleep* for defeating *go running*), without centralized decision systems.

In the SOM agents and agencies embed perceptual and motor processes but no memory, that is instead represented by specialized agents called *K-lines*: they are neural-like, distributed structures, that involve many links between agents representing parts and patterns of experienced situations. Remembering is thus rebuilding what was active: K-lines are (partially or totally) reactivated when a new situation is encountered that resembles an old one; they in turn reactivate al the related agents by spreading activation through the links. Minsky (in preparation) extends the concepts of K-lines: they not only activate other "memory management" agents, but become "selector" of whichever kind of resource (or agency), including patterns of emotional states. K-lines provide a good substrate for many dynamics of context-sensitive cognitive phenomena such as graded or partial recollection, blendings and analogy.

There are many more agent kinds in SOM, (e.g. *nomes* and *nemes*), and they are mainly organized through frames and frame systems (a versatile formalism). Much of the SOM structure is compositional and hierarchical: there exist links and patterns of activations between the agents that implement in a distributed way a planned activity: for example, the agency for *building* learns to activate in sequence the agents for *picking*, *releasing* and so on. Special emphasis is given to a general learning principle: adapting and reusing some capabilities for other purposes, and organizing knowledge in a way that affords abstractions and analogy. Minsky (1986) calls it the *Papert's Principle: some of the most crucial steps in mental growth are based not simply on acquiring new skills, but on acquiring new administrative ways to use what one already knows*.

There is not a single, general purpose method for resolving problems, but on the contrary many inferences and problem solving systems (e.g. the *differences engine*) that interoperate and to a certain extent "emerge" as the pattern of response of the whole SOM to a situation. The SOM activity is thus dynamic, emergent, adaptive and highly context dependent. This versatility depends mainly from the fact that *knowledge is distributed* and, depending from the context (e.g. the active K-lines), *only a part of it is available* in a given moment. This is intended to address the frame problem: how to consider only *salient* information.

Distributed, Emerging Representations

Perhaps the first problem to address in the tentative of building a cognitive architecture is about the representations it exploits. The first and more traditional kind of representations, the symbolic ones, have been criticized in the recent years for many reasons: the most crucial and basic problem is their "groundedness" (e.g. Chalmers, 1992). Here we address only a limited point of the discussion, i.e. the fact that they are rigid and often they are pre-fixed by the developer: they are empty labels and not *active symbols* in the sense of Hofstadter et al. (1994).

Distributed, emergent representations are implemented and exploited in different ways in many kinds of models. The connectionist, PDP approach (Rumelhart, 1986) replaces symbolic and discrete representations with patterns of fully distributed representations, e.g. embedded in neural networks. Connectionist networks embody knowledge structures organized around *prototype-style* representations, i.e. multidimensional semantic space dotted with attractors (Churchland, 1989). In a similar way, Hofstadter claims that active symbols only emerge from distributed representations; even if he proposes more coarse-grained representations, the points is that a token only assumes a semantic where interacting in a network of related ones. Another interesting possibility in a similar direction is the model of Gärdenfors (2000) that bridges the symbolic and subsymbolic levels by a middle-level, geometric representation that treats concepts as vectors in a n-dimensional (features) space. Another family of models (including LISA: Hummel, 1996) exploits timing of activation of distributed patterns for dynamic binding of representations: elements oscillating in synchrony are bound together and form more complex representations.

As Chalmers (1992) points out, the advantage of distributed representations is that they are not carried on by atomic tokens (that can only be empty, ungrounded labels: this is the criticism to old-fashioned AI), but they have *meaningful structures*, distributed e.g. in patterns of activation of a network of tokens. We assume (even if this point is controversial) that this property is shared by all the previously cited examples, even if they use different levels of granularity: in all the cases the level of the tokens (syntactic) lies below the level of the representations (semantic). In the symbolic case, syntactic operations over the tokens are not assumed to have a direct semantic (e.g.

adding the symbol "A" to the symbol "CAT"), but each operation needs an external semantic interpretation. On the contrary, in the case of distributed representations, *their structures can be directly manipulated and exploited* by using operations that are directly meaningful; for example by performing arithmetic (the case of neural networks) or geometrical operations (the case of concepts in Gärdenfors, 2000); these operations modify the semantic of the representations (activating more a node or a concept means more relevance; crossing two concepts in a concept space produces a new concept) and not only their syntactic form.

The central capability provided by distributed representations is that *the description of a situation (or an object or a problem) emerges from patterns of activity of tokens at a lower level*, representing e.g. their parts or features. Systems exploiting distributed representations can build up their models of the current problem or situation instead of receiving as input pre-structured representations. These models are more versatile: the interaction between contingent phenomena and pre-existent knowledge (e.g. in a semantic network) leads to the emergence of new representations in a dynamic and context-sensitive way.

Here we introduce hybrid architectures, coupling the expressive power of symbolic modeling with the dynamics of distributed representations.

Hybrid Architectures

In hybrid representations both symbolic and connectionist elements are present; the claim of hybrid cognitive modeling is that many interesting cognitive phenomena emerge from the interplay of simple and narrow-minded processing units (in the sense of the SOM). The case of representations is similar: a meaningful representation only emerges from the activity of tokens that lie at a lower level of description. The point is how to couple connectionist dynamics and emergent representations with symbol/token manipulating systems.

There exist many kinds of hybrid architectures, depending on which feature is hybridized. For example, ACT-R is hybrid in the sense that it exploits two kinds of representations, *declarative* and *procedural*. Some other architectures (such as CLARION: Sun, 1997) are hybrid in the sense of layered: in the bottom layer (more reactive) connectionist dynamics make available symbolic, localist representations in the upper layer (more deliberative).

AKIRA (as well as DUAL) is *hybrid at the micro-level* (Kokinov, 1997): this means that *each agent/daemon is hybrid*, carrying on both symbolic content (e.g. in the form of a frame) and connectionist elements (activation level, energetic links); and the two aspects interact. Some connectionist features are used in the symbolic phase: for example, activation becomes the priority of the agent, and the semantic relations between the daemons are reflected by the topology of the links (e.g. an IS-A link spreads energy).

These systems thus exploit distributed representations and connectionist dynamics; instead of using simple and meaningless units such as nodes in the neural networks, these systems use agents, each having an activation level representing the availability of the (localist) representation they carry on. For example, in one of the main applications of the architecture DUAL, a model of analogical reasoning called AMBR (Kokinov, 1997), the granularity of the representations is one single object per agent: each agent includes a frame representing e.g. a cup, or a plate. Situations are fully distributed representations: e.g. "the cup is on the plate" is represented by a number of (partially) activated agents having links and exchanging activation; moreover, the patterns of activation change dynamically because of system evolution (e.g. decay) or because of the intervention of new elements or events (e.g. the cup is broken). By exploiting distributed and dynamic representation AMBR is able to perform analogical reasoning in a context dependent way: the result of the computation dynamically emerge from the parallel and concurrent activity of many (partially) active agents, carrying on (portions) of semantic information both in their symbolic and connectionist parts.

In a similar way, Copycat performs analogical reasoning on the basis of a problem space that is not pre-fixed, but changes dynamically during the analogical processing. Representation-building and mapping (e.g. of two situations) run in parallel and influence each other.

In AKIRA we have implemented (Pezzulo, 2005) a parallel BDI (Rao et al., 1995) where Goals and Plans are implemented using Daemons; their usual relations (e.g. a Plan can satisfy a Goal) are represented by using energetic links that are the carriers of activation e.g. from Goals to Plans. The control cycle is thus parallel and distributed among the concurrent processes without central interpreters.

All these systems exploit hybrid representations (hybrid agents for AKIRA and DUAL; a semantic network and a procedural memory for Copycat). Of course, the symbolic part can be set at whichever level of granularity (e.g. at the level of objects, or features); normally these representations are referred as "localist" and contrasted to "fully distributed" ones. Their advantages and disadvantages are a matter of discussion; localist ones are usually preferred in hybrid architectures, because they are more manageable and understandable, apparently without losing expressive power. In fact, even if all the systems here described could in principle exploit fully distributed representations (such as neural networks), there are many problems with them: for example, they make it difficult to design at an high level, mainly because is not possible to interpret them (e.g. in symbolic terms) once they are modified, since the structural operations they exploit make the resulting structures opaque. The point here is having at least two levels of description, one for the tokens and one for the representations: thus, representations of situations emerge from patterns of activation over simpler structures (such as objects and events, the tokens), and the emergence is dynamic and context aware. New, more complex representations can thus be formed on-line by the system either by procedural operations running concurrently and modifying the problem space (e.g. the *codelets* in Copycat), or by dynamic binding (such as in LISA), or by operations whose availability is directly coupled to the most active representations (the case of AKIRA and DUAL).

AKIRA and its Main Components

AKIRA is a C++ platform fully developed in open-source. It is not a cognitive architecture, but a framework for building cognitive models at different levels of complexity and integration. AKIRA does not commit to a single model, but many design choices are inspired by the Society of Mind (Minsky, 1986), the Pandemonium (Jackson, 1987), DUAL (Kokinov, 1994) and Copycat (Hofstadter et al., 1994).

Here we briefly introduce AKIRA's components: the kernel (called Pandemonium); the Agents (called Daemons); the Blackboard, the pool of resources (Energy Pool). For full reference, see Pezzulo (2005) or www.akira-project.org.

The kernel is called Pandemonium; it is the management structure of a set of Daemons, performing a number of routine actions such as Blackboard management, garbage collection, monitoring the system, etc.

The Daemons are the basic kind of agents in AKIRA. Each Daemon has a thread of execution; a functional body where its behavior is specified; and a concurrent access to a central resource called the Energy Pool, that is the core of the *AKIRA Energetic Model*. The Pool is limited and gives an upper bound to the available resources: differently from multi-agent systems, the priority of Daemons' threads are directly linked to their current energy. This feature (the *Energetic Metaphor*: Kokinov, 1994) forces the Daemons to compete: only a limited amount of processes can run, and only a limited amount of representations are available in a given moment, because only some Daemons are active in a certain moment. This models dynamic context-sensitive activation of salient resources, in the spirit of the SOM.

Daemons can: tap energy from the Energy Pool; spread activation to the other ones via an Energetic Network; form on-the-fly assemblies (called Coalitions) for resolving complex tasks; exchange explicit messages via the Blackboard; exploit *implicit* communication, consisting in monitoring the activity of another Daemon ("hearing its shriek"), that is routinely notified to the Blackboard; and execute their symbolic operation, but only if they have enough energy (because operations have a cost).

Fig. 1 provides an intuitive picture of the concurrency model. The activation of each Daemon is calculated by an energetic network affording energetic exchanges. Activation becomes priority of the Daemons' threads, driving their sequences of activation: more active Daemons can act more. Thus, Daemons are represented both as *nodes* (circles in the grid on the bottom), that exchange activation (via the links), and as *agents* (circles in the cloud on the top). The *priority* of the agents (their height in the cloud) depends on the activation of the correspondent nodes. Daemons also share a limited amount of energy (summing up to the Energy Pool).

A Daemon can be used for modeling a specific function, or a module, or embed a piece of semantic information, in a flexible way. According to SOM, Daemons should be specialized for simple tasks; complex ones should be managed by high-level Daemons exploiting the results of low-level ones, or by more complex structures (as in the case of the complex "agency for building") or by the whole Pandemonium. The behavior of the whole system emerges thus from energetic dynamics (the *AKIRA Energetic Model*) without centralized control. All the Daemons can participate (totally or partially) to a given computation, and in principle each function, module or activity can influence each other, even without a common language or ontology.

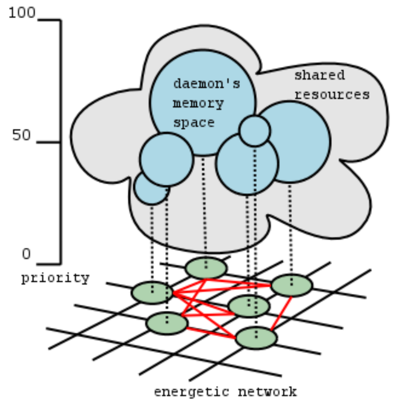

Figure 1. Daemons as nodes and as agents.

Thus, *AKIRA realizes dynamic computation by exploiting distributed representations and processes*. This scheme allows developers to design any kind of cognitive architecture, by defining any kind (and degree) of modularization and any granularity of the representations.

AKIRA does not define any specific agent architecture but provides a set of prototypes that can be extended for realizing and customizing agents having different design (e.g. reactive, deliberative, layered) and capabilities. The *AKIRA Macro Language* provides a rich toolkit of resources, including, BDI (Rao et al., 1995); Behavior Networks (Maes, 1990); Neural Networks, Fuzzy Logic and Fuzzy Cognitive Maps (Kosko, 1997), etc. Our aim is to allow cognitive modelers to compare different solutions for the same problem and to integrate different models. The Macro Language permits to abstract from implementation details, designing at the level of the cognitive functions.

Conclusions and Current Work

We have discussed the role of distributed representations and "flexible" modules in cognitive architectures. We have introduced hybrid architectures, where connectionist and symbolic elements are merged and influence each other. Hybridization permits to model *accessibility* and *saliency* of processes and representations: for example, in AKIRA each agent has an associated priority depending on its contextual relevance. As in the Global Workspace Theory (Baars, 1998) this "attentional focus" influences not only representations but even processes (and possibly modules): focused processes communicate more, perform more operations and influence more the other ones. The main features of AKIRA are inspired by SOM and Pandemonium: implicit communication, concurrency and cooperation between the agents (i.e. even without explicit messaging); hierarchies and coalitions of agents; lack of centralization; the Blackboard as a shared workspace[1].

Cognitive modelers can take advantage of these features, that can be used as common functionalities for developing a broad range of models. For example, we used AKIRA for modeling decision under uncertainty (Pezzulo et al., 2004) and goal-oriented processing (Pezzulo, 2005). In both cases, the knowledge and control structures are distributed: the most salient beliefs, goals and actions are more active, influencing more the overall state of the computation.

(Pezzulo, 2005b) describes a Pandemonium-like model where many specialized Daemons (mainly color and shape detectors) realize a visual search task in a collaborative way, by jointly influencing the "focus controller" Daemon in a measure that is proportional to their current activation and saliency. Thus, while in the original Pandemonium model a central decision process was used, we used a distributed schema. Some Daemons also monitor the progresses of the others: for example, line detectors are sensitive to the activity of point detectors and exploit their activity. As a design rule, we preferred monitoring activity to explicit message passing. The model also includes top-down influences, with goal processes (e.g. "Find the Red T") pre-activating some feature recognizers (e.g. red recognizers) and inhibiting others (e.g. green recognizers); we are now improving the model by adding Daemons that learn regularities and anticipate interesting locations to search.

Acknowledgments

Work funded by the EU project MindRACES, FP6-511931.

References

Anderson, J. R. & Lebiere, C. (1998). *The atomic components of thought*. Mahwah, NJ: Erlbaum.

Baars, Bernard J. (1988). *A Cognitive Theory of Consciousness*. New York: Cambridge University Press

Brooks, R.A.. Intelligence without Representation. *Artificial Intelligence*, Vol.47, 1991, pp.139-159

Bryson, J. J. Modular Representations of Cognitive Phenomena in AI, Psychology and Neuroscience. *Visions of Mind.* Darryl Davis ed. Final, 2004.

Cassimatis N. L. (2002). *Polyscheme: A Cognitive Architecture for Integrating Multiple Representation and Inference Schemes.* Doctoral Dissertation, Media Laboratory, MIT, Cambridge, MA.

Castelfranchi C., Silent Agents: From Observation to Tacit Communication, In: *Proceedings of MOO 2004*

Chalmers, D., 1992, Subsymbolic Computation and the Chinese Room, in J. Dinsmore (ed.), *The Symbolic and Connectionist Paradigms: Closing the Gap*, Hillsdale, NJ: Lawrence Erlbaum

Churchland, P. M. *The Neurocomputational Perspective*. Cambridge: MIT/Bradford Books, 1989

Clark, A. (1997) *Being There: putting brain, body, and world together again* MIT Press.

Fodor J. *Representations.* Cambridge, MA: MIT Press 1981

Gärdenfors, P. *Conceptual Spaces - The Geometry of Thought*, Cambridge: MIT Press, 2000

Hofstadter, D. R., and M. Mitchell. 1994. The Copycat Project. In *Advances in connectionist and neural computation* theory, Vol. 2: logical connections, ed. K. J. Holyoak, and J. A. Barnden. Norwood N.J.: Ablex.

Hummel, J. E., & Holyoak, K. J. (1996). LISA: A computational model of analogical inference and schema induction. *Proceedings of the Eighteenth Annual Conference of the Cognitive Science Society* (pp. 352-357). Hillsdale, NJ: Erlbaum.

Jackson J. V., Idea for a Mind. *Siggart Newsettler*, 181 1987

Kokinov B. N., The context-sensitive cognitive architecture DUAL, in *Proceedings of the Sixteenth Annual Conference of the Cognitive Science Society*, Lawrence Erlbaum Associates, (1994).

Kokinov, B. N. (1997). Micro-level hybridization in the cognitive architecture DUAL. In R. Sun & F. Alexander (Eds.), *Connectionist-symbolic integration: From unified to hybrid approaches* (pp. 197-208). Hilsdale, NJ: Lawrence Erlbaum Associates.

Kosko B. *Fuzzy Engineering* Prentice-Hall, 1997.

Maes P., Situated Agents Can Have Goals. *Robotics and Autonomous Systems*, 6 (1990).

Minsky M. *The Society of Mind*. Simon and Schuster, N. Y. 1986

Minsky, M. *The Emotion Machine*. (In preparation)

Newell, A. (1990). *Unified Theories of Cognition*. Cambridge, MA: Harvard University Press.

Pezzulo G., Calvi G. Designing and Implementing MABS in AKIRA P. Davidsson et al. (Eds.) *MABS 2004, LNAI 3415*, pp. 49–64, 2005

Pezzulo G., Calvi G. (2005b) Dynamic Computation and Context Effects in the Hybrid Architecture AKIRA *Proceedings of CONTEXT 2005*

Pezzulo G., Lorini E., Calvi G. (2004). How do I Know how much I don't Know? *Proceedings of COGSCI 2004.*

Rao A., Georgeff M., BDI Agents from Theory to Practice, *Tech. Note 56, AAII*, 1995.

Rosenbloom, P. S., Laird, J. E. & Newell, A. (1992) *The Soar Papers: Research on Integrated Intelligence*. Volumes 1 and 2. Cambridge, MA: MIT Press.

Rumelhart D. E., Mc Clelland J. L. and the PDP Research Group, *Parallel distributed processing: explorations in the microstructure of cognition*. Vol. I, 1986.

Sloman, A. 1999. What Sort of Architecture is Required for a Human-like Agent? In *Foundations of Rational Agency*, ed. M. Wooldridge, and A. Rao. Dordrecht, Netherlands: Kluwer Academic Publishers.

Sun R., *Learning, action, and consciousness: a hybrid approach towards modeling consciousness*. Neural Networks, 10 (7), pp.1317-1331. 1997.

[1] Not all the elements of SOM are followed to the letter. For example, SOM K-lines are quite separated from other agencies. However, it is possible to represent K-lines in AKIRA by using Daemons having semantic content and no functional body.