

A triple loop model of agent cognition

Mixel Kiemen Evolution, Complexity and Cognition

(ECCO), Vrije Universiteit Brussel,
Krijgskundestraat 33, B-1160 Brussels
Belgium
info@mixel.be

Abstract

We introduce an evolutionary-cybernetic control model for agent cognition, using programming modelling to go from conceptual design to implementation. We show how primitive instructions can be integrated via a bootstrapping network into higher-level cognition. The basic cognitive module loops between the semantic meaning of input and the syntactic aspects of the associated memory. The loop performs a context-dependent focus evaluation. Three context-focus modules, perception, motivation and reasoning, together create the agent cognition. The interaction leads to actions and to a constructive learning behaviour, where the learning will define the syntax.

1 Introduction

Present artificial intelligence (AI) is grounded in two competing paradigms: symbolic and embodied. While some may favour one approach over the other, we merely propose that each specific program model is good in expressing a specific cognitive capacity. From a historical perspective, symbolic AI received a lot of criticism because of its grounding problems [Searle, 1980]. This led to embodied AI as a reaction. Embodied AI is to some degree a return to *connectionist models to confront the grounding problem* by focusing on the low-level processing of sensory information.

But we shouldn't forget that symbolic AI focuses on higher-level cognition, a task for which connectionist models are clearly not very well suited. The symbolic approach used and developed languages, such as Lisp and Prolog, that are as abstract as possible, to express high level processes. So *language models frame higher-level cognition* and are better to express symbolic information processing. This leads us to the following research question: How do we get from sensory to symbolic information processing? And how can we go from connectionist models to language-evaluators?

By combining several software architectures we shall *frame complexity* to express the transition of information processing from lower to higher cognition. In

general, our model is related to the paradigm of Complex Adaptive Systems (CAS) [Holland, 1995] and we shall link CAS with general software models to understand the information processing in our model. In particular, we introduce a model based on evolutionary cybernetics [Heylighen, 1992b] and *a network of bootstraps to ground language*. The basic module is context-dependent and produces a *focus-evaluation*: the evaluation parses more semantics out of the input by syntax evaluation, while parsing more syntax out of the memory by semantic association.

With the basic architecture explained we continue to see how three modules (perception, motivation and reasoning) lead to cognition. The modules generate actions while the cognitive evaluation of the actions produce constructivist learning [Piaget, 1970]. We conclude the paper with a sketch of an operational simulation where our model is used to investigate the discovery and usage of artefacts [Simon, 1996].

2 The basic architecture of the model

It is commonly suggested that it is good to start with simple models and move to more complex ones later, but this is dangerous when complexity is a capability of its own. Consider situated and embodied cognition. Most implementations deal with specific sensory-motor aspects. However, when we look at the richness of real embodiment [Clark, 1996], it becomes unclear where the mind begins or ends, and impossible to see how simple connectionist models could model such shifting boundaries. To understand the mind, we have to consider the complexity of its process in time [Port and van Gelder, 1995], and the specific mechanisms through which the cortex produces higher level cognition [Hawkins and Blakeslee, 2004].

When we try to simulate complexity, we cannot just focus on the individual elements of code. A complex phenomenon is like a whirlpool: we cannot explain its structure on a local, molecular level, but only on a global, thermodynamic level where we see the dissipative structure emerge. Similarly, to express the emergence of cognition, we cannot just work on a local algorithmic level: we need a global software architectural level to express the emerging of cognition. This can be illustrated by relating CAS properties (flow, aggregation, and diversity [Holland, 1995]) to software

representation.

Flow refers here to *the continuous character of information processing*. The flow idea should direct the readers attention to the on-going process rather than to its outcome. Flow can produce *aggregation* via *parallel processing*: the individual processes can perform basic operations while the overall program only becomes visible when we look at their interaction. In our simulation, we have used the parallel programming method to create interaction between our modules.

Diversity the variety of forms we need to fill the different evolutionary niches is essential to learning. Therefore we created a tool to generate a diverse environment, using *object-oriented prototyping* [Lieberman, 1986]. The objects in the simulation (including the agent) are first cloned from a distributed prototype. We then randomize the values of the clones attributes according to a specific probability distribution to ensure variation.

2.1 Flow and bootstrapping

Mutually defining concepts can be bootstrapped to produce the distinctions defining a structured language [Heylighen, 1992a; 2001]. In a bootstrapping procedure, component A is used to develop component B, while B is used to develop A. We use a similar method to make the evaluation process work without hardcoding the language (as would be done in symbolic AI). Our model is based on a network of bootstrapping processes, as illustrated in Fig.1 There are three types of bootstrap: one by information distinction (syntax and semantics), two by interaction (in cognition and in learning) and two between the flow speeds (cognition-action and cognition-learning).

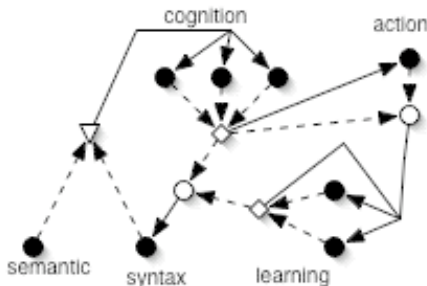


Figure 1: the bootstrap network: aspect (black dots) in relation to bootstraps types: distinction (triangle), interaction (squares) and flow (circles)

The bootstrap within each module results in fast cognition. The bootstrap between the modules produces actions. As there is more than one process speed, we can bootstrap between flows: the interaction between cognition and action leads to learning. The learning has two types that will bootstrap with each other to produce constructive learning (loop learning). The learning also creates syntax by abstracting concepts. Thus, syntax is the result of a bootstrap between cognitive evaluation which uses it and learning which defines it.

2.2 Context-focus modules

Our model has only one basic type of module, which is characterized by one novel aspect: focus evaluation. The focus evaluation functions with an internal and an external input which are connected via a loop passing through working memory (Fig. 2). The external input provides the raw, incoming data while the internal input contains syntactical information retrieved from working memory by associative matching of more abstract patterns processed earlier. The matching of a pattern happens in general through auto-association, but triggers an associated network of other patterns. While the pattern has semantic meaning (because it results from a measurement of an outside phenomenon), the associated network is structured by syntax. The syntax is used to produce specific evaluations, transforming the first abstract match to a detailed and concrete syntax evaluation system, via an internal feedback loop. The association is, by nature of the input, context-dependent. The output of the module is passed through a working memory pool in order to make the information available for other modules. Thus, working memory provides a bridge relating different mental states and actions.

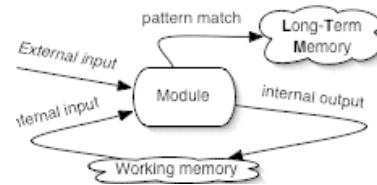


Figure 2: context-focus module

Let us illustrate this abstract process by considering the focus evaluation in vision. In the first few milli-seconds of visual processing, the perception will be fuzzy and general. For example, fast moving objects appear often only as vague spots in our visual field. A first association would be to distinguish light from dark. Next, our saccading eye will accentuate the boundaries between lighter and darker regions, so that it can distinguish shapes. The shape will trigger an associative pattern with a meaningful syntactical structure. Now, each module makes simple associative matching with this pattern. The perception module may associate the shape with the feature round, while the reasoning module may associate round with face. Via the working memory the perception module can reach syntactical information about the concept face. Some of this information will be about specific visual patterns, such as the presence and relative position of the components eye, nose, and mouth. This will trigger a visual action pattern specific for perceiving faces: our saccading eye will repeatedly look at locations close to the eyes and the mouth.

3 Agent cognition

Our model is built around three context-focus modules connected via their shared working memory (Fig. 3), so that higher-level cognition can emerge from their interactions. The modules can be classified by their

overall orientation, which is *intentional, external or internal*. Each orientation has its mode of evaluation: intentional by motivating, external by perceiving, and internal by reasoning. Each module has its own input data: needs for *motivation*, stimuli for *perception* and personality for *reasoning*.

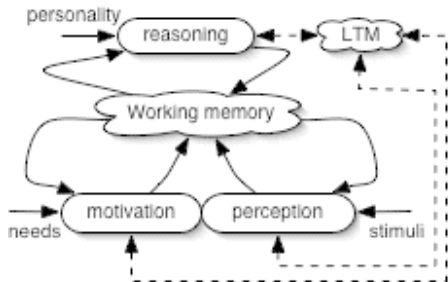


Figure 3: interaction between the three basic modules

The input always is a whole experience leading to an input state. Stimuli measurements (images, sounds, smells, tastes...) lead to perceptual states (shapes, intonations, flavours...). Needs measurements (internal temperature, energy level, blood sugar level...) lead to motivational states (cold, hunger, tiredness, boredom). Reasoning inputs are less obvious: memory will contain information about the persons abilities, such as mental and physical strength, ways of problem-solving (social, creative), etc. We can see these memories as the agents personality. Measurements of them define the reasoning state (introvert or extrovert, sensitive or forceful, etc).

3.1 Perception

Let us use general Gestalt phenomena as a case study to illustrate perception. In *reification*, we experience (by association) more information than is actually present in the external stimuli. For example, the triangle in Fig. 4.A is clearly experienced, but not part of the sensory data. It appears as if perception is not about what we (physically) see, but about what we see it related with.

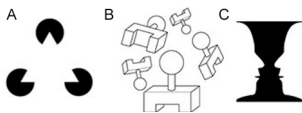


Figure 4: A) reification B) invariance C) multistability

We can try not to see the triangle, just like we can choose which of two figures to see in a multistable picture (Fig. 4.C), but that is hard. The effortful action of not seeing what we normally see is driven by motivation and by reasoning: if we have enough knowledge of how we actually got tricked into seeing a triangle (reasoning) and keep on focusing to the task (motivation), we can suppress automatic visual perception. The result of suppression is the disappearance of the experience: it will be removed from the internal input, but it was never in the external input.

Simple geometric evaluations are part of our perceptual control. They create basic visual recognition

phenomena like reification and invariance (Fig. 4.B). More complex phenomena like multistability are created by an interaction between the modules. Interaction can result in the sudden emergence of a picture (see Fig. 5). Once we recognize a Dalmatian dog (reasoning) in the scattered black and white dots of Fig. 5, we can see the dogs spots in an invariant way. What is more, the interpretation will be stored in memory: if you look at the picture again a few days (or even years) later, you are likely to immediately recognize the dog. The storing of the image has little influence on the module evaluation, but it will be important for learning.



Figure 5: context-focus module

Notice that what we might call higher cognitive recognition is not a part of perception; it does not take place at a higher level either. We should better call it an interactive pattern. Reasoning and motivation are as basic as perception: the pattern matching is merely performed in a different type of context.

3.2 Motivation

In the motivation module, the external input has a degree of *relevance*: if the motivational state has high priority or is urgent, it will determine the whole process. For example, while diving under water your need for oxygen will gradually increase and eventually force you to go up again. If the need is a more vague state of mind, it will merely modulate or refine your evaluation of internal input. For example, if you are tired this will influence the way you behave, making you prefer less energetic forms of acting. Let us demonstrate an interactive pattern: while you were reading about suppressing the triangle perception (Fig. 4.A), you may have found it relevant to test this mechanism (reasoning). The concept of testing involves a lot of motivation syntax.

The motivation is intentional and the focus evaluation will accordingly filter the working memory. The internal input retrieved information from the working memory. In the case of perception, the information would be put back into the memory pool, with extra associations. In the case of motivation, the information may be inconsistent with the intention and therefore get thrown out. Suppressing the triangle is an example. On the other hand, the input may be very relevant to the intention, and therefore be pushed to further exploration. Motivation will drive the intention to intentional action, and from there all the way to action-in-progress [Searle, 2001]. While the action is in progress the motivation will ensure that it stays in line with the intention.

Although the motivation module has intentions, it works toward its goals without a rigid direction. This

indirectness make the intention sensitive to change: the other modules may bring in new information that changes the intention drastically. For example, assume you intend to cross a playground as quickly as possible. However, while crossing, you see a ball coming in your direction. When you notice that the ball will hit you on the head, you change your intention, now aiming at a side-way move to avoid the collision.

3.3 Reasoning

The internal input will be even more important for reasoning than it was for motivation. In contrast to perception, which is driven by external input, reasoning will be driven by internal input. The most basic operations produced may give a close conceptual association to another concept. For example, the agent could have the need to drink and the perception of a glass. Reasoning would make the match between drinking, glass, and the fact that a glass may contain water which can be drunk. An interactive pattern can solve holistic concept by associating it with unusual associations (like metaphors). For example, you could have a set of concepts that are strongly related, but without being able to point out the relation. The other modules may (by their presence) add some other concepts to the group and thus pinpoint the relation between the other associations (an Aha! experience).

Because it can find analogies in concept contexts an agent with reasoning will be fitter to survive in its environment. A simple example is improper use of an object when reasoning about alternatives for a task. E.g., you can use a glass to keep your pencils together on your desk. While most associations to the concept glass would be about liquids, we can see how the feature keeping things together can be associated with the problem of pencils. The new association would also be stored. Again, the storing would have little influence for the current evaluation but it is vital for learning.

4 Learning and behaviour

Our model was designed with the aim of simulating an agent who is capable of constructing mental (concepts) and physical tools (artefacts) that may help in problem-solving. Construction is the result of learning, i.e. constructions grow by the selective storing of perceived and reasoned information (see 3.1 and 3.3). The construction is based on the cognitive state during acting. Each module is responsible for a specific type of process (see Fig. 6): motivation produces actions, perceived actions produce experience, and reasoning about experience produces abstract concepts.

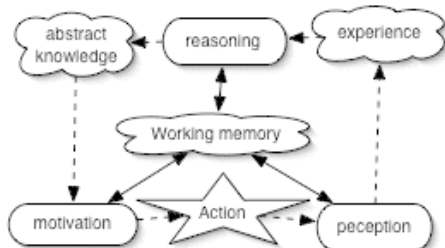


Figure 6: agent behaviour

We distinguish two types of learning: *behaviour mastery* and *reflection*. Their main difference resides in what we have called the orientation of the modules. During behaviour mastery motivation will focus on experiencing; during reflection it will focus on abstracting. But before we can delve deeper into learning, we must explain the agents general behaviour or action performance.

4.1 Behaviour

The cognitive architecture as we have sketched it will be used to plan, monitor and if necessary correct the actions that an agent performs. For example, to produce the action pick-up we need to consult our memory about the pick-up process: what stimuli can we expect (shape, surface roughness, weight, hardness, etc. of the object to be manipulated) and how will they influence the required patterns of muscle movement? Also the relation between the object and the action process is important, as we will need different action patterns to handle different types of objects, such as a sack of potatoes, a crystal vase, or a bucket of water. Depending on the relevant variables the intended action determines an initial pattern of motion. The perception of the action-in-progress will then replace the abstract values by concrete ones. For example, the weight of the object can be anticipated using vision first and updated once the sensations from the muscles kick in. Well-known actions can be automated so that they only require a few check-up tests. For example, when riding a bike the complex and dynamic causal relations between muscle activities can be learned so well that the basic movement can be automated and only the overall balance needs to be checked.

4.2 Behaviour mastery

Behaviour mastery starts from an abstract concept without immediate physical or concrete grounding. For example, you may be used to ride a bike in the normal way, while having the abstract concept of riding without using your hands. The abstract concept may be indirectly related to other concepts that are based in physical facts, but the new action would require a wholly different pattern of muscle activity. During the action-in-progress the difference between assumptions and reality will become clear as the perception gets that information. The motivation will drive the focus towards to experiencing the information that is relevant to the intention. The action-in-progress will be a rough approximation of the intended action. The reasoning module will acquire a lot of new experience to process, and make a coarse conceptualization. The action is refined through repetition while physical associations get linked into the concept. For example, the attempt to ride the bike with no hands will initially produce a very clumsy movement. After a few attempts, the control of balance, which is normally performed by the hands will shift to the hips. In that way, the abstract concept will get implemented as a concrete action pattern, being grounded in newly experienced physical facts.

4.3 Reflection

Reflection is similar in process to behaviour mastery but goes in the opposite direction: from concrete facts to abstract concepts. Assume that a lot of facts are known about a family of concepts, but that there exist no direct association between them. For example, an agent has gathered experience trying to lift various larger and smaller objects, and developed for each type of object a specific expectation of the amount of muscular force that needs to be applied. The reflection will create an abstraction of such diverse associations. For example, by reflecting on these experiences, the agent may come up with the concept of weight to organize the whole.

Reflection normally relies on the reasoning module, which will be driven by motivation to produce an integrating framework for the information gathered via the perception module. However, reflection can also indirectly use the perception module to help it with its complex, information-processing task. To achieve that it can use the simple trick of exteriorizing abstract concepts in the form of concrete physical representations, e.g. by talking, writing or drawing. This external representation can then again be processed via the quick and efficient perceptual module. Realizing the concept in the physical world allows the agent to embed the perception in the reflection process. This is the perspective of the external mind [Clark and Chalmers, 1998].

Note that there is no guarantee that reasoning will be able to abstract the relevant associations. An example is the phenomenon of change blindness [Rensink *et al.*, 2003] in which clear visual changes in a picture are not noticed because they are not expected to happen in the given context. It is simply difficult to detach oneself from the specific context in order to formulate a more general principle. To illustrate this we could look at science in action [Latour, 1987]: most of the important scientific innovations are based on a reframing of concepts developed in one context or discipline to apply them in another one (e.g. using radioactive decay (nuclear physics) for absolute dating (archaeology)).

5 The operational simulation

While we have outlined our model, it remains at a high level of abstraction that may raise question about its concrete implementation. The present section will show that the model can be operationalized by sketching a simulation that we made of an agent learning to use a stone as a tool.

One of the reasons that the simulation is only introduced at the end is because of the syntax creation nature of our model. In the simulation we only illustrate a very small step in the syntax creation [Kiemen, 2003]. Still, the code behind our simulation is quite large, because it is based on the whole architectures that supports our model (see 2). The simulation uses threads to simulate parallel processing, prototyping to create the components of the environment, and a logical evaluator for the syntax parsing in the modules.

It is programmed in Squeak, a variant of the object-oriented language SmallTalk, that allows us to make use of morphics. But once we go into the module, we use computational reflection [Maes, 1987] to go from Smalltalk to Prolog.

The simulation is inspired by apes using objects such as sticks and stones to open a nut. In our simulation the agent has an abstract notion of action, that can take on five variables: do, get, on, at, with, representing respectively the type of action, its result, its object, its position and the instrument used to perform the action. do and get are labels used purely for the agents cognitive appraisal of its situation; the other variables are linked (by perception) to objects in the environment. The agent contains a starting memory of four primitive actions:

- do: move get: newPosition
- do: pickup get: Object at: Object
- do: eat get: stopHunger on: food
- do: break get: fragment on: stone with: stone

If a variable has no value, it is assumed that the action can take place without this attribute. E.g. move is executable without any condition. The pickup action is more abstract than the others as it uses a variable Object (the capitalization refers to a variable), which can be replaced on both positions by one of the concrete objects (food, stone and nut). The agent has a fixed goal or intention (stop hunger). The motivation module will try to realize that intention, producing a stack containing the actions: eat, pickup, and move. The last, abstract action (break) is not part of the intentional context, but will be needed later for a concept context.

Let us now consider the objects in the agents environment. The nut object has a variable representing the hardness of the nuts shell. If the value is very low, the nut will be perceived as food and it will most likely produce the action break when picked up. The agent will start moving once the motivation has created the stack and the perception module has replaced the label food with a concrete object (in the simulation it is the pointer to the object). Note the aspect of flow in the running of the process threads. For example, 1 action step and 10 evaluation steps will be produced in the same time frame. Pickup requires 3 action steps, giving the perception 30 evaluation-steps to notice the breaking of the nut during this action. After a few pickups, the agent learns about the break, i.e. the similarities between experienced action-patterns of breaking will be recognized and the behaviour mastery will learn precisely which action pattern can break the nut. In our simulation we see the addition of an abstract action:

- do: break get: food on: nut

This experience produces a shift in the agents behaviour, since now nut too will be an option for satisfying the stop hunger intention. Still, certain hardness values for the nut-object will be so high that the agents physical properties (expressed as finger strength) will

be insufficient to break the nut. This is where reflection becomes necessary. The agent now knows two break actions, one applied to stones (initially given), one applied to nuts (learned). In the initial action it was a stone that produced the breaking (with:stone). This association can now be applied to produce a new, intentional break action which will then be tested. If the test is successful, the following new action is stored in memory:

- do: break get: food on: nut with: stone

Note how physical and mental retrieved associations grow onto the break concept, which may result in a more abstract, encompassing action category.

6 Conclusion

We started with a theory of how a network of bootstrapping process could provide a foundation for language. The language implies a syntax evaluation, parsing and defining. The evaluation is the core process of our module where the parsing creates behaviour (cognition and action) and the behaviour (via learning) creates the syntax. The phenomenon is well-known as constructive learning and we proposed a design that would make it possible to engineer such behaviour. The most central aspect in our design is the transformation from association to syntax. The most general syntax can be embedded as hard syntax, defining a hardwired, reflex level of cognition. Soft syntax defines the learned level of cognition. The soft syntax emerges only after a long period of acting with existing syntax, from the structure of the learned associations. (we view this structure as syntax because it is used by evaluation). This model is but a start in the investigation of the complex interplay between perception, learning, and action that defines cognition, and we need to further explore the possibilities for implementation and design. The model should examine syntax creation in more depth, which may require communication and particularly awareness. A promising approach to model such conscious information processing that we plan to explore is the idea of the mind as a virtual machine [Sloman, 1994; Sloman and Chrisley, 2003].

References

- [Clark and Chalmers, 1998] Andy Clark and David J. Chalmers. The extended mind. *Analysis*, 58:10–23, 1998.
- [Clark, 1996] Andy Clark. *Being There: Putting Brain, Body, and World Together Again*. MIT Press, Cambridge, MA, USA, 1996.
- [Hawkins and Blakeslee, 2004] Jeff Hawkins and Sandra Blakeslee, editors. *On Intelligence*. Times Books, 2004.
- [Heylighen, 1992a] Francis Heylighen. From complementarity to bootstrapping of distinctions: A reply to Ijffren’s comments on my proposed ’structural language. *International Journal of General Systems*, 21(1):99, 1992.
- [Heylighen, 1992b] Francis Heylighen. Principles of systems and cybernetics: an evolutionary perspective. *Cybernetics and Systems*, 8:368–373, 1992.
- [Heylighen, 2001] Francis Heylighen. Bootstrapping knowledge representations: from entailment meshes via semantic nets to learning webs. *Kybernetes*, 30(5/6):691–722, 2001.
- [Holland, 1995] John H. Holland. *Hidden order: how adaptation builds complexity*. Addison Wesley Longman Publishing Co., Inc., Redwood City, CA, USA, 1995.
- [Kiemen, 2003] Mixel Kiemen. Het appropriatiegedrag. Master’s thesis, Vrije Universiteit Brussel, 2003.
- [Latour, 1987] Bruno Latour. *Science in action: How to follow scientists and engineers through society*. Harvard University Press, Cambridge, MA, 1987.
- [Lieberman, 1986] Henry Lieberman. Using prototypical objects to implement shared behavior in object-oriented systems. In *OOPSLA ’86: Conference proceedings on Object-oriented programming systems, languages and applications*, pages 214–223, New York, NY, USA, 1986. ACM Press.
- [Maes, 1987] Pattie Maes. Concepts and experiments in computational reflection. In *OOPSLA ’87: Conference proceedings on Object-oriented programming systems, languages and applications*, pages 147–155, New York, NY, USA, 1987. ACM Press.
- [Piaget, 1970] Jean Piaget, editor. *Structuralism*. Harper & Row, New York, 1970.
- [Port and van Gelder, 1995] Robert F. Port and Timothy van Gelder, editors. *Mind as motion: explorations in the dynamics of cognition*. Massachusetts Institute of Technology, Cambridge, MA, USA, 1995.
- [Rensink et al., 2003] R.A. Rensink, J.K. O’Regan, and J.J. Clark. To see or not to see: The need for attention to perceive changes in scenes. *Psychological Science*, 8:368–373, 2003.
- [Searle, 1980] John Searle. Minds, brains and programs. *Behavioral and Brain Sciences*, 3(3):417–457, 1980.
- [Searle, 2001] John Searle. *Rationality in action*. Harvard University Press, Cambridge, 2001.
- [Simon, 1996] Herbert A. Simon. *The sciences of the artificial (3rd ed.)*. MIT Press, Cambridge, MA, USA, 1996.
- [Sloman and Chrisley, 2003] Aaron Sloman and Ron Chrisley. Virtual machines and consciousness. *Journal of Consciousness Studies*, 10(4/5):113–172, 2003.
- [Sloman, 1994] Aaron Sloman. Semantics in an intelligent control system. *Philosophical Transactions of the Royal Society Series A: Physical Sciences and Engineering*, 349(1689):3–57, 1994.