COGNITIVE ASSEMBLER

Michal Malý[1]

[1] Dept. of Applied Informatics, FMFI, Comenius University,
Mlynská dolina, 842 48 Bratislava, Slovakia, maly@ii.fmph.uniba.sk

ABSTRACT

The cognitive assembler is an analogy to the computer assembly language. It is a model to describe an elementary level of instructions in a brain. It analyzes the behavior of a brain but in a higher level than neuronal, it wraps details of neuronal computations into symbolic instructions.

We will try to use this cognitive assembler to more closely and finely describe the behavior of a brain observed in experiments. We formulate hypotheses about how the execution of basic instructions in a brain should be driven.

1. INTRODUCTION

The assembly language or sometimes known as an assembler is a low-level programming language describing instructions for the processor. Using symbolic abbreviations it expresses elementary operations that the processor is able to execute. During normal work we do not observe this level of operation. The executed instructions give rise to a high level user friendly interface.

Although the level of the assembler is low it is possible to go lower: we can observe the stream of bits in the memory, or even watch the electric signal measured on the given pin outs.

We use many methods to investigate cognition. On the one end of the pole we observe EEG, i.e. voltage outputs of real neuron cells, or we use neural networks to simulate these potentials. On the second end of the pole we observe behavior of animals or people, we investigate their reactions or we try to model them. Surely there exist many methods between those poles, but we think the part analogical to the computer assembler is not covered by the current research.

The task of the cognitive assembler is to describe the function of neural networks on a higher level, to cover details of neuronal computations into symbolic instructions and so provide a possibility to break the behavior description into smaller components.

2. MOTIVATION AND INSPIRATION

The knowledge of the processes of the brain can help and inspire us in artificial intelligence research. Our goal is to better describe these processes. Simultaneously the model will provide questions, which will be an inspiration for new cognitive experiments. Eventually when the cognitive assembler will be sufficiently described we can simulate its instructions and create an artificial mind up to the richness as that of the description of the instructions.

This article freely follows the ideas drawn in [2].

3. PRINCIPLES, RESOURCES AND ASSUMPTIONS

3.1. Biological plausibility

The brain and the nervous system work on the basis of neurons. The assembler instructions must be executable by a neural network or have to correspond to a chemical effect of hormones. An expert on neural networks has to declare that he is able to construct a neural network which could execute a given instruction, or there should be a neuroscientist who can declare that there is a part of brain which carries out what the instruction does.

3.2. Minimality of instructions and structures

It is desired that only a small number of instructions is used. If we would need a new instruction or change the structure of the data with which we work for each new type of behavior we want to describe, we cannot expect that our system would be complete and general. We would probably have to change it again for a new requirement.

However if we can cover many types of behavior by a small number of instructions, we are on the right truck.

The brain thinks in many ways but we are seeking relatively simple principles. We cannot afford to encode new incoming meanings by new ways. The representation of meanings has to be free: must not be explicitly provided by the programmer but has to be created in the model.

3.3. Autonomous control

Our model of the cognitive assembler has to describe the way of control of the execution, and input of instructions. We cannot expect that an external teacher will provide patters to learn from. An animal or a human brain finds these patterns by itself, so must our model. We will propose some general rules which our model will use.

3.4. The brain as a happiness machine

We will hold that the brain follows its own satisfaction, its own happiness. This principle might seem too egoistic or too biased, because a human does not follow only his own interests. But we can argue that a good man feels an adequate satisfaction when he acts right, does what he considers fair. And this satisfaction can be higher than an ephemeral or a bodily pleasure.

In this sense is this machine to be understood. It tries to reach the most satisfactory state in a given situation. The attractiveness of the state does not have to be stable. It can change during life as a result of nurture or experience.

3.5. Recognition

We assume that the brain can recognize a situation which has already occurred. We will recall the semantics of discrimination criteria, proposed in [4] and developed in [3].

The input for the discrimination is the sensory perception and the internal state. Let the inputs from sensors come as a vector and the internal state as a set of pairs (state, activation ratio).

The discrimination criteria is a function from the input into the interval <0,1> determining how the situation corresponds to the given state.

It is possible to use a large number of functions from a linear classifier (e.g. perceptron) to a complex qualitative criterion, e.g. a formal language, deciding whether the input word belongs to the specified language.

4. MODEL

The basis for the model is a varying valuated graph with vertices and two types of edges between them. We will describe the graph using the following language.

4.1. Graph of meanings, associations and actions

4.1.1. The states -- vertices of the graph

They represent meanings, which the brain has found and has assigned them a discriminating criterion.

Example: *a, b, c*

4.1.2. The emotional value of the state

For each state there is one real number, it is a preference of the state - how “good” is the state, how happy the brain feels when this state is activated.

Example: *motivation(a) = 0.8*

4.1.3. Valuated non-oriented association edges

The value represents the measure of the association binding between those two meanings and it is a real number.

Example: assoc(a ,b) = 0.75

4.1.4. Valuated oriented action edges

The orientation represents the starting point and the goal of the action. The action itself is defined as an output vector sent on effectors, e.g. muscles. The value represents the probability perceived by the model that the given action will result in the goal state.

Example: *act(catch_mouse , has_mouse) = 0.5*

4.2. Model functionality

We can imagine the behavior of the model approximately as a finite-state automaton. Our model resides in many states simultaneously, and simultaneously moves between them. When the model moves, it executes chosen actions between the states. The structure of the graph can evolve by the time.

The selection of states and actions depends on which states is the model in, on their valuation and their existence and the valuation of edges.

The states which the model resides in express the scope of its thoughts and attention. We will use the following notation.

4.3. Focus and control of the attention, recognition of the situation

The focus to the current states we denote by a set of pairs (state, real number)

Example:

focus = {(a,0.9),(b,0.5),(c,0.1)}

This means that three states are activated, first of them strongly, the situation experienced is very likely *a*, second expresses that the situation can be described as "partly holds *b*" and the third is a non-important, but existing association with *c*.

4.4. Selection of goal and action

Our model selects goal states according to their motivation. It uses two simple searches. First, it finds adjacent vertices via associative edges and tests their discrimination criteria. It resembles thinking about the current situation.

Then we search action edges. We examine each possible action from the current set of states. The action we can execute will be evaluated according to which states it results. The resulting set of states is expanded by the first type of search. Finally, the action is evaluated according to the motivation of the set states.

The best action will be executed.

4.5. Changes in the graph structure

Our model can change the structure of graph ("learn") during the movement between states and experiencing external stimuli. These are possible operations:

- create a new state
- create or increase an association
- delete a state
- decrease or delete an association

The modification of association bindings will be automatic according to the Hebb's rule. Each time when two states are excited (focused) concurrently, their association increases, and vice versa.

It is practical to have a threshold here. If the association value exceeds this threshold, we will consider the edge as created; in the case that the value is less than the threshold we will consider the edge as deleted.

A new state will be created when the model registers something new, which is different from previously recognized meanings. The newly created state is empty, only a discrimination criterion is set according to the new fact in the current situation.

Let's imagine a familiar room. We see and recognize a table, which has a state assigned to it. The room itself also has an assigned state. This state is associated with the state representing the table and states representing other pieces of furniture which we see in the room.

Suddenly somebody puts a new vase on the table. Other objects are not changed; the only thing which is new here is the vase. The visual system can compare the image of the room without the vase and with the vase. It separates a new object and its discrimination criterion will be a mask created from the difference between those images.

At this time we do not deal with the risk of too many new states.

4.5.1. Creation of abstractions

By the term abstraction we mean a mechanism which extracts common properties and ignores non-important differences. It groups many observed events into one and so it enables adaptation and the advantage of prediction of the future. It can also serve as a novelty detector when creating new states.

1. There are states $a_1, a_2,..., a_i$.
2. We try to generalize them into a new state a'.
3. The new state is valuated according to the states $a_1, a_2,..., a_i$ so that it represents an abstraction of these states.

The simplest example of this abstraction mechanism is a simple comparison. If there is a complete equivalence, the states will be grouped together. We can improve it so it ignores a given number of differences.

Another well-known mechanism is a neural network.

Another example can be a compression algorithm. It can distinguish static and dynamic properties. Static properties have to be written only once, and dynamic properties are generated according to the formula found. Again we see an ability to predict: if the data do not contain unpredicted events it does not have to code the change and the resulting compressed file will be smaller in size.

We see that in a different situation a different mechanism can be used. We will not prescribe now which is to be used. We plan to test our model with the increasing complexity of these abstraction mechanisms.

5. TESTING OF THE MODEL

Our model, even in the case that it satisfies our intuition about biological processes in the brain is still only a theoretical construction. It is desirable to compare it to the reality. We can do this in two steps:

1. Try to use the model to explain real experiments, where the mapping from a real behavior to the model is done by hand.
2. Simulate the model and compare its behavior to reality.

The first step is appropriate in the beginning. It enables us to bridge over doubts and to find out whether the model is in principle viable for following research. Here we test, if the model has a sufficient expression power to express the behavior. We observe the holes in the model and we try to help it, here we have to improve the model.

This brings a risk of making the model vague, the help being too intelligent and not being able to reproduce it later in a program.

If we are done with the first phase, we can proceed to the second step. All help is here prohibited: the algorithm has to run automatically.

We are now in the first phase. We present an example, how the experiment description could look like.

5.1. Description of an real experiment: Monkeys and dogs

Experimenters [1] were investigating cognition of chimpanzee and dogs. After a hint from the experimenter, the animal should choose from two offered containers. One of the container contained hidden food.

The dogs were more successful when the hint was in a social form, e.g. pointing.

The monkeys were better when the hint was causal, e.g. a shaking container.

5.2. Our description of the experiment

1. *motivation(food)=1*
   motivation(no_food)=0

   The food is a stable motivation for an animal.

2. *act(bucket_with_food,open, food)*

   If a bucket contains food, we gain food.

3. *act(bucket_without_food,open,no_food)*
If a bucket does not contain food, we do not gain food.

4. *motivation(bucket_with_food)=1*
motivation(bucket_without_food)=0
The valuation of the container is similar to what it contains. The valuation is transferred from the food to the container.

5. *focus = {(pointed_bucket, 1), (bucket_contains_food, 1)}* occurs often → *assoc(pointed_bucket, bucket_contains_food)* will increase
We learn that the container which has been pointed to is the container with food.

6. *focus={(not_pointed_bucket, 1), (bucket_does_not_contain_food, 1)}* occurs often → *assoc(not_pointed_bucket, bucket_does_not_contain_food)* will increase
We learn that the container which has not been pointed to is not the container with food.

7. *motivation(pointed_bucket) = 1*
motivation(not_pointed_bucket) = 0
We learn that the container which has been pointed to is the good one.

8. *focus = {(pointed_bucket, 0.5), (not_pointed_bucket, 0.5)}*
We see both containers, one is pointed to.

9. That leads to *focus={(pointed_bucket, 1)}*
We transfer attention to the good one.

10. This leads to "*open*" action.

11. *Food* is a reward.

We can draw a similar scheme not only for a container which has been pointed to, but also for the container which was shaken. The difference between dogs and monkeys can we assign to the motivation to observe the gesticulation, which is important for association in the fifth and the sixth step.

6. CONCLUSION

6.1. General questions

What is the minimal starting knowledge to begin a reasonable simulation? What do we need to boot the cognitive process? Which meanings have to be given – innate? Which meanings is the model able to learn and which not? And what are the conditions for it?

6.2. Implementation questions

How to implement the abstraction – generalization? What differences in learning do we gain for a different abstraction mechanism? How to set the constants for states and operations with them? What results do we obtain for different constants?

6.3. Related work

There exist a handful of cognitive architectures, which we have not thoroughly analyzed up to this time. We offer a short comparison with one of them.

CLARION [5] distinguishes between implicit and explicit processes. This distinction is not present in our model. Its action decision making algorithm can be compared to ours. CLARION incorporates action-centered and non-action centered subsystems, contrary to our model, where general knowledge inference and decision making is integrated in one structure. A more detailed comparison with this architecture could be a source of inspiration for us.

6.4. Summary

We presented a model inspired by our belief that neural structures in a biological brain are able to create associations between meanings, enable the model to create an abstract generalization and can recognize learned facts. We tried to capture their work in a propositional symbolic form. We ignored the details of neurons and the details of an implementation of concrete neural networks which would be capable to realize these operations.

Instead of that we proposed a dynamic data structure which enables a parallel run of cognitive threads. It also enables a fuzzy evaluation. We were motivated by a desire to create a simple but general structure, able to create states representing new meanings. We thus avoided the need to define them explicitly. Planning of the next action is based on a motivational component of meanings. The structure - graph with parallel states valuated by a motivation enables short term and long term planning and can abandon the original plan. We illustrated the model in a short example.

Our model provides a lot of questions and opens possibilities for a future work, the clarification of the model and its testing. This is the subject of a future research.

7. REFERENCES

[1] J. Bräuer et al. Making inferences about the location of hidden food: Social dog, causal ape. In Journal of Comparative Psychology, volume 120, pages 38-47, 2006.

[2] Dana Retová, Jarmila Šilliková, Ján Šefránek. Opice, psy, sémantika a logika. In J. Kelemen, V. Kvasnička, J. Pospíchal, editors, Kognice a umelý život VII. Slezská univerzita, Opava, 2007.

[3] M. Takáč. Cognitive semantics for dynamic environments. In Contributions to ICCS 2006 – 14th International conference on conceptual structures. Aalborg, Denmark: Aalborg University Press., 2006.
[4] Ján Šefránek. Cognition without mental processes. In J. Rybár, Ľ. Beňušková, V. Kvasnička, editors, Cognitive science. Kalligram, 2002.
[5] Ron Sun, The motivational and metacognitive control in CLARION. In: W. Gray (ed.), Modeling Integrated Cognitive Systems. Oxford University Press, New York. 2007.