A Cognitive Architecture Based on Dual Process Theory

Claes Strannegård[1], Rickard von Haugwitz[2], Johan Wessberg[3], and Christian Balkenius[4]

[1] Department of Philosophy, Linguistics and Theory of Science, University of Gothenburg, Sweden and Department of Applied Information Technology, Chalmers University of Technology, Sweden
`claes.strannegard@gu.se`

[2] Department of Philosophy, Linguistics and Theory of Science, University of Gothenburg, Sweden
`rickard.von.haugwitz@gu.se`

[3] Institute of Neuroscience and Physiology, University of Gothenburg, Sweden
`johan.wessberg@gu.se`

[4] Department of Philosophy, Lund University, Sweden
`christian.balkenius@lucs.lu.se`

Abstract. This paper proposes a cognitive architecture based on Kahneman's dual process theory [1]. The long-term memory is modeled as a transparent neural network that develops autonomously by interacting with the environment. The working memory is modeled as a buffer containing nodes of the long-term memory. Computations are defined as processes in which working memory content is transformed according to rules that are stored in the long-term memory. In this architecture, symbolic and subsymbolic reasoning steps can be combined and resource-bounded computations can be defined ranging from formal proofs to association chains.

Keywords: cognitive architecture, dual process theory, computation, transparent neural network.

1 Introduction

Bridging the gap between symbolic and subsymbolic representations is a – perhaps *the* – key obstacle along the path from the present state of AI achievement to human-level artificial general intelligence. [2, p. 79]

This paper is concerned with artificial general intelligence (AGI). Our ultimate goal is to create a computational model that may operate in any environment and develop intelligence adapted to that environment in a fully automatic fashion. In particular, we would like our model to be powerful enough to handle both symbolic and subsymbolic reasoning, as per the distinction made in [2].

As the human brain is the only system known to us that fulfills the above criteria, we have turned to psychology and neuroscience for inspiration. The proposed model was inspired by developmental psychology in that it learns from its environment; by cognitive psychology in that it includes standard components of memory system models and dual process theory; and by neuroscience in that it is based on a network model. Our priority is problem solving and not biological realism. Therefore the proposed model does not strive to reflect all facts that have been established in the brain sciences.

This model consists of a long-term memory (LTM) structured as a developing transparent neural network [3] and a working memory (WM). The model is a generalization of some previously developed cognitive models for propositional logic [4], first-order logic [5], and sequence extrapolation [6]. As these models have been reported to perform above the average human level on the problem domains under study, the proposed model can be expected to do the same.

The remainder of the paper is organized as follows: Section 2 presents some brief remarks concerning cognitive modeling in general. Section 3 presents the cognitive architecture used herein. Section 4 discusses the computations employed in this model. Finally, Section 5 presents the conclusions of this work.

2 Cognitive Modeling

This section provides a brief background on some aspects of cognitive modeling that are relevant for the proposed model.

Memory plays a central role in the model. Wood *et al.* define *memory* as the capacity to use previous experience to inform subsequent behavior; *long-term memory* to be temporally indeterminate and independent of specific task demands; *working memory* to be a functionally distinct memory structure, finite in capacity and retention period, bounded by context and task demands, and used for retention of task relevant information [7].

Sometimes the "magic number seven" is used to refer to the limited capacity of working memory that can typically hold about seven items [8], but later studies suggest the capacity to be about four items in young adults and less than that in children and elderly [9]. In neuroscience memory formation has been studied in the Hebbian tradition ("neurons that fire together wire together") [10] and memory decay has been considered as the effect of synaptic decay ("use them or lose them") [11]. Memory decay was investigated experimentally by Ebbinghaus [12], who constructed a curve describing how forgetting is affected by the number of repetitions and the time interval between repetitions.

Traditional computational models include symbolic models such as automatic theorem provers [13], sub-symbolic models such as artificial neural networks (ANNs) [14], and probabilistic models such as Bayesian networks [15]. Computational models of particular interest with respect to the distinction between symbolic and subsymbolic processes include hierarchical temporal memory [16], long short-term memory [17], conceptual spaces [18], and neural-symbolic systems [19]. Cognitive architectures of particular interest in the present context include Soar [20], ACT-R [21], MicroPsi [22], Clarion [23], CHREST [24], and NARS [25]. Many of these model reasoning processes in the style of Newell and

Simon [26] and treat these processes as computations in abstract term-rewriting systems [27].

In general, symbolic systems are good for reasoning but much less useful for perceptual tasks. This can be solved by building hybrid systems, but many hybrid systems are limited by the difficulty of designing interfaces for complex interactions between their subsystems. This phenomenon holds true even if the basic concepts are shared among the subsystems. In general, a hybrid of n specialized programs can handle n specialized domains, but it is challenging to capture the deep interactions between these domains and to ensure useful generalization beyond these domains.

Hybrid-like systems have also been proposed within psychology. For example, several versions of dual process theory exist. The version introduced by Kahneman [1], that has been an inspiration for the present work, features two systems named System 1 (Intuition) and System 2 (Reasoning). System 1 is used for fast, associative thinking, whereas System 2 is used for slow, analytical thinking.

A computer program that has been developed by traditional means by a human engineer tends to be understandable in the sense that an informed human, *e.g.*, the author of the program or a peer, can explain how the program works, predict its input-output behavior, and predict the consequences of modifying the program in various ways. In contrast, artificial neural networks, which are represented by matrices of real-valued connection weights, tend to be difficult to understand. Except for trivial cases, it is virtually impossible for humans to understand what functions such neural networks compute. This holds true even for small feed-forward networks and more so for recurrent networks. It would thus be useful if there existed a transparent way to design these types of models.

Many of the above-mentioned computational models are problematic when considered in light of our present desiderata: some require human interaction to adapt to new problem domains (*e.g.* in the form of manual programming or manual selection of training sets); some are hybrids with unclear or insufficient interaction between subsystems; some are difficult to understand and are therefore problematic to use as foundations for more complex models; some are severely limited in their versatility or computational power; some have unclear interfaces for communication with the external world; some specialize exclusively in symbolic or sub-symbolic processing. The cognitive architecture presented in the next section was designed to avoid these problems.

3 Cognitive Architecture

This section introduces our novel cognitive architecture that solves some of the problems discussed above and combines different modes of operation within a unified transparent architecture. First, we describe the LTM, which is modeled as a network that develops according to certain rules.

Transparent Neural Networks. Let I be the set of real numbers in the interval $[0, 1]$. This set will be used to model signal intensity and connection weights.

Definition 1 (TNN). *A* TNN *is a structure* (V, E_1, E_2)*, where*

- E_1 *and* E_2 *are binary relations on* V*.*
- *Each element of* V *has a unique associated label from the following list:*
  $SENSOR_i$ *where* $i \in M$ *(fan-in 0)*
  $MOTOR$ *(fan-out 0)*
  MIN *(fan-in 2)*
  $DELAY$ *(fan-in 1)*
  $SPACE(\mu, \sigma)$ *(fan-in 1).*
  The fan-in and fan-out restrictions refer to the relation E_2*.* M *represents a set of modalities and* μ *and* σ *are real numbers.*
- *Each element of* E_1 *has a unique associated weight in* I*.*
- *The structure* (V, E_2) *is an acyclic directed graph.*

As we shall see, E_1 and E_2 are used for modeling System 1 and System 2 processes, respectively.

Activity. TNN stimuli are modeled as follows:

Definition 2 (Stimuli). *Let* T *be the set of integers (modeling time). A* stimulus *for a TNN with sensor set* $V' \subseteq V$ *is a function* $S : V' \times T \to I$*.*

Stimuli give rise to two types of activity that propagate through the TNN. Roughly stated, the two types of activity model System 1 and System 2 processes, respectively. The two types of activity propagate along the edges of E_1 and E_2, respectively. Therefore a TNN can be viewed as two subsystems that interact with each other. Our rationale for introducing two types of activity (and here we may depart from Kahneman's model) is that it enables us to make the fundamental distinction between perception and imagination. We believe this distinction to be crucial for many cognitive processes, including hypothetical reasoning. For instance, it enables us to distinguish between the perceived taste of an apple and the imagined taste of an apple or between a real lion and an imagined lion. It is not hard to imagine scenarios in which such distinctions can be critical to survival. Despite some undesired connotations, we shall call the two types of activity perception and imagination, respectively.

Definition 3 (Perception). *The* perception $p : V \times T \to I$ *is defined as follows. If* $t \leq 0$ *then let* $p(a, t) = 0$ *and if* $t > 0$ *then let*

$$p(a,t) = \begin{cases} S(a,t) & \textit{if } a \textit{ is labeled SENSOR} \\ p(a',t) & \textit{if } a \textit{ is labeled MOTOR}, (a',a) \in E_2 \\ \min\{p(a',t) : (a',a) \in E_2\} & \textit{if } a \textit{ is labeled MIN} \\ N_{(\mu,\sigma)}(p(a',t)) & \textit{if } a \textit{ is labeled SPACE}(\mu,\sigma), (a',a) \in E_2 \\ p(a',t-1) & \textit{if } a \textit{ is labeled DELAY}, (a',a) \in E_2. \end{cases}$$

Here $N_{(\mu,\sigma)}(x) = \exp\{-(x-\mu)^2/\sigma^2\}$*. This is the Gaussian (similarity) function with mean* μ*, standard deviation* σ*, and max value 1.*

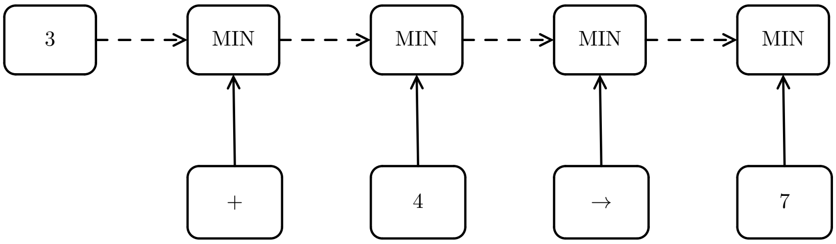

Fig. 1. Sequences. Solid arrows represent E_2-edges, and dashed arrows represent E_2-edges with a DELAY node inserted in the middle. This network represents the rewrite rule 3+4 → 7 as a sequence of symbols. The nodes with fan-in 0 can either be dedicated sensors or complex networks that recognize the corresponding symbols. The top right node becomes activated if and only if the sequence $3 + 4 \to 7$ is perceived.

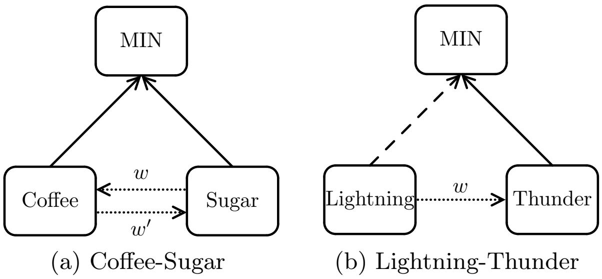

Fig. 2. Associations. Dotted arrows represent E_1-edges. Again, the nodes with fan-in 0 can either be dedicated sensors or top nodes of networks that recognize the corresponding concepts. In panel (a), the E_1-edges propagate imagination from Coffee to Sugar and also from Sugar to Coffee. In panel (b), the E_1-edge leads to the prediction of Thunder whenever Lightning occurs.

The SPACE nodes are used for storing values. The closer the input is to the stored value μ, the closer the output is to 1. DELAY nodes delay signals by one time unit.

Definition 4 (Imagination). *The* imagination $i : V \times T \to I$ *is defined as follows: if* $t \leq 0$ *then let* $i(a,t) = 0$ *and if* $t > 0$ *then let*

$$\begin{aligned} i(a,t) = \max\, &\{p(a',t) \cdot w(a',a,t) : (a',a) \in E_1\} \cup \\ &\{\zeta(i(a',t) \cdot w(a',a,t)) : (a',a) \in E_1\}), \end{aligned}$$

where $\zeta : I \to I$ *is a damping function and* $w(a',a,t)$ *is the label on the edge* $(a',a) \in E_1$ *at time* t.

Examples of TNNs are given in Figures 1 – 2.

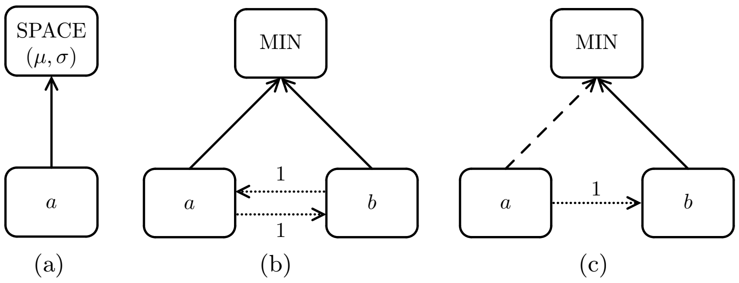

Fig. 3. The three memory addition rules. (a) Space memory addition. Starting from node a, this rule forms the structure shown in the figure with $\mu = p(a, t)$ and $\sigma = 0.25$. This rule can only be triggered if $p(a, t)$ differs sufficiently from all the values stored in the existing SPACE nodes that are connected to a. (b) Associative memory addition. Starting from the nodes a and b, this rule forms the structure shown in the figure. This rule can only be triggered if a and b"fire together." (c) Sequence memory addition. Starting from nodes a and b, this rule forms the structure shown in the figure. This rule can only be triggered if b and the delayed signal of a "fire together". Here, the rule applies to the special case when the delay is one time unit, but it also applies to arbitrary time intervals.

Development Rules. Now, we briefly survey the development rules. There are 15 development rules, some of which have rather involved triggering conditions. To conserve space, therefore, we only present partial descriptions of the most fundamental rules. An earlier set of rules is given in [3]. The development rules can be divided into rules for forming, updating, merging, and removing memories. The effects of the three memory addition rules are shown in Figure 3. The modalities of sensors, and by extension of other nodes, play a role in memory addition, as unimodal concepts are prioritized over polymodal concepts. The memory removal rule works as follows: for each node in V, a vitality score is stored and updated according to Ebbinghaus' forgetting curve. If the vitality score of $a \in V$ falls below a given threshold, then a is removed in an "avalanche" together with other structures that become detached upon the removal of a. Essentially, the memory removal rule serves as a filter that preserves memories of recurrent phenomena but removes memories of coincidences. The main rationale for this rule is to keep the number of memory structures on a manageable level.

The LTM is continuously developed using these rules, starting from a initial TNN at $t = 0$. The set of sensors are held constant throughout development. The WM is a list of nodes of the LTM. The WM can be regarded as a low-capacity memory buffer containing pointers to pieces of information (chunks) in the form of LTM nodes. An *information state* is specified by the stimuli, the WM content, and when applicable, an LTM node encoding a transition.

Actions. Actions are binary relations among information states. Section 4 provides several concrete examples of actions. There are three types of actions:

Attentional Actions. input information into the WM. These actions include Look, Listen, and Taste.

Computational Actions. manipulate information in the WM. These actions include Rewrite, which carries out the transition from a to b, provided that $a \to b$ is in the LTM, cf. Figure 1; the action Associate, which carries out the transition from a to b, provided that $w(a, b, t)$ is maximal, cf. Figure 2a; and the action Speculate, which carries out the transition from a to b, where b is chosen randomly from the E_1-successors of a, with probabilities proportional to their weights.

Motor Actions. output information from the WM to the MOTOR nodes. These actions include Speak, Write, and Move.

4 Computations

A *computation* is a sequence of information states, where each transition is generated by an action. Several examples of computations are given in Tables 1–6.

Table 1 shows an example of arithmetical calculation. Here, $17 \cdot 3$ is computed in a rewrite system similar to the one used in [6]. The rewrite rules are stored as nodes representing sequences in the LTM, cf. Figure 1. The symbol sequence $17 \cdot 3$ is given as visual input. Look puts this sequence into the WM, and a series of Rewrite applications transforms it into 51. Finally, Write invokes the write module, which outputs 51.

Table 2 shows a propositional proof. The tautology $p \vee (p \Rightarrow q)$ is proven. The proof system here is similar to that used in [4]. The formula is provided as visual input and then rewritten to True in a goal-driven process using several

Table 1. Arithmetical calculation

Stimuli	**WM**	**LTM**	**Action**
$17 \cdot 3$			Look
	$17 \cdot 3$	$17 \to (10 + 7)$	Rewrite
	$(10 + 7) \cdot 3$	$(x + y) \cdot z \to x \cdot z + y \cdot z$	Rewrite
	$10 \cdot 3 + 7 \cdot 3$	$10 \cdot 3 \to 30$	Rewrite
	$30 + 7 \cdot 3$	$7 \cdot 3 \to 21$	Rewrite
	$30 + 21$	$30 + 21 \to 51$	Rewrite
	51		Write

Table 2. Propositional proof

Stimuli	**WM**	**LTM**	**Action**
$p \vee (p \Rightarrow q)$			Look
	$p \vee (p \Rightarrow q)$	$(x \Rightarrow y) \to (\neg x \vee y)$	Rewrite
	$p \vee (\neg p \vee q)$	$((x \vee (y \vee z)) \to ((x \vee y) \vee z)$	Rewrite
	$(p \vee \neg p) \vee q$	$(x \vee \neg x) \to$ True	Rewrite
	True $\vee\, q$	(True $\vee\, x) \to$ True	Rewrite
	True		Write

Table 3. Sequence generation

Stimuli	**WM**	**LTM**	**Action**
$1, 2$			Look
	$1, 2$	$x, y \rightarrow x, y, x + y$	Rewrite
	$1, 2, 1 + 2$	$1 + 2 \rightarrow 3$	Rewrite
	$1, 2, 3$	$x, y \rightarrow x, y, x + y$	Rewrite
	$1, 2, 3, 2 + 3$	$2 + 3 \rightarrow 5$	Rewrite
	$1, 2, 3, 5$		Write

Table 4. Classification

Stimuli	**WM**	**LTM**	**Action**
Apple			Taste
	Apple	Apple $\twoheadrightarrow$ [æpl]	Associate
	[æpl]		Speak

Table 5. Association

Stimuli	**WM**	**LTM**	**Action**
Coffee			Taste
	Coffee	Coffee $\twoheadrightarrow$ Sugar	Associate
	Sugar	Sugar $\twoheadrightarrow$ Brazil	Associate
	Brazil	Brazil $\twoheadrightarrow$ Football	Associate
	Football		

Table 6. Speculation

Stimuli	**WM**	**LTM**	**Action**
	Beach	Beach $\twoheadrightarrow$ Ocean	Speculate
	Ocean	Ocean $\rightarrow$ Water	Rewrite
	Water	Water $\twoheadrightarrow$ Drown	Speculate
	Drown		

applications of Rewrite. The output True is then sent to a write module, which outputs True.

Table 3 illustrates sequence generation. A Fibonacci sequence is generated by repeated applications of Rewrite using the sequence $x, y \rightarrow x, y, x + y$. This example illustrates how the model can extrapolate from examples.

Table 4 shows a classification. In computations, we use the symbol $\twoheadrightarrow$ for E_1-edges. Here, an apple is given as the stimulus. The action Taste identifies the topmost active taste-node, which is inserted into the WM. Then, Associate replaces this node by the phonetic sequence [æpl].

The formation of an association is shown in Table 5. Here, Coffee taste begins a chain of associations. The transitions follow the E_1-edges with the highest weights.

Finally, Table 6 shows how the model can speculate. Speculations are modeled as random walks along the E_1-edges with random start nodes.

5 Conclusion

A cognitive architecture was constructed with transparency as one of the major design goals. A partial implementation exists in Haskell. Dual process theory was used in order to maintain the crucial, but often neglected, distinction between reality (perception) and imagination. The architecture is based on two memory systems: (i) a long-term memory, which is an autonomous system that develops automatically through interactions with the environment, and (ii) a working memory, which is a memory system used to define the notion of (resource-bounded) computation. This notion of computation is general enough to model arithmetical calculations, propositional proofs, sequence generation, classification, association, and speculation. Thus, symbolic and subsymbolic processing can coexist and interact with each other in this monolithic architecture.

References

1. Kahneman, D.: A perspective on judgment and choice: mapping bounded rationality. American Psychologist 58, 697 (2003)
2. Goertzel, B.: Perception processing for general intelligence: Bridging the symbolic/subsymbolic gap. In: Bach, J., Goertzel, B., Iklé, M. (eds.) AGI 2012. LNCS, vol. 7716, pp. 79–88. Springer, Heidelberg (2012)
3. Strannegård, C., Häggström, O., Wessberg, J., Balkenius, C.: Transparent neural networks: Integrating concept formation and reasoning. In: Bach, J., Goertzel, B., Iklé, M. (eds.) AGI 2012. LNCS, vol. 7716, pp. 302–311. Springer, Heidelberg (2012)
4. Strannegård, C., Ulfsbäcker, S., Hedqvist, D., Gärling, T.: Reasoning Processes in Propositional Logic. Journal of Logic, Language and Information 19(3), 283–314 (2010)
5. Strannegård, C., Engström, F., Nizamani, A.R., Rips, L.: Reasoning about truth in first-order logic. Journal of Logic, Language and Information, 1–23 (2013)
6. Strannegård, C., Amirghasemi, M., Ulfsbäcker, S.: An anthropomorphic method for number sequence problems. Cognitive Systems Research (2012)
7. Wood, R., Baxter, P., Belpaeme, T.: A review of long-term memory in natural and synthetic systems. Adaptive Behavior 20(2), 81–103 (2012)
8. Miller, G.: The magical number seven, plus or minus two: some limits on our capacity for processing information. Psychological Review 63(2), 81 (1956)
9. Cowan, N.: Working memory capacity. Psychology Press, New York (2005)
10. Baars, B., Gage, N.: Cognition, brain, and consciousness: Introduction to cognitive neuroscience. Academic Press (2010)
11. Wixted, J.: The psychology and neuroscience of forgetting. Annu. Rev. Psychol. 55, 235–269 (2004)

12. Ebbinghaus, H.: Memory: A contribution to experimental psychology. Number 3. Teachers college, Columbia university (1913)
13. Harrison, J.: Handbook of practical logic and automated reasoning. Cambridge University Press (2009)
14. Rumelhart, D., McClelland, J.: Parallel distributed processing: Psychological and biological models, vol. 2. The MIT Press (1986)
15. Pearl, J.: Probabilistic reasoning in intelligent systems: networks of plausible inference. Morgan Kaufmann (1988)
16. Hawkins, J., George, D.: Hierarchical temporal memory - concepts, theory, and terminology. Technical report, Numenta, Inc. (2006)
17. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Computation 9(8), 1735–1780 (1997)
18. Gärdenfors, P.: Conceptual Spaces. MIT Press (2000)
19. d'Avila Garcez, A.S., Lamb, L.C.: Cognitive algorithms and systems: Reasoning and knowledge representation. In: Cutsuridis, V., Hussain, A., Taylor, J.G. (eds.) Perception-Action Cycle. Springer Series in Cognitive and Neural Systems, pp. 573–600. Springer, New York (2011)
20. Laird, J., Newell, A., Rosenbloom, P.: Soar: An Architecture for General Intelligence. Artificial Intelligence 33(3), 1–64 (1987)
21. Anderson, J., Lebiere, C.: The atomic components of thought. Lawrence Erlbaum, Mahwah (1998)
22. Bach, J.: The MicroPsi agent architecture. In: Proceedings of ICCM-5, International Conference on Cognitive Modeling, Bamberg, Germany, pp. 15–20 (2003)
23. Sun, R.: The importance of cognitive architectures: An analysis based on Clarion. Journal of Experimental & Theoretical Artificial Intelligence 19(2), 159–193 (2007)
24. Gobet, F., Lane, P.C.: The CHREST architecture of cognition: the role of perception in general intelligence. Kitzelmann, E (2010)
25. Wang, P.: From nars to a thinking machine. In: Proceedings of the 2007 conference on Advances in Artificial General Intelligence: Concepts, Architectures and Algorithms: Proceedings of the AGI Workshop 2006, pp. 75–93. IOS Press, Amsterdam (2007)
26. Simon, H., Newell, A.: Human problem solving: The state of the theory in 1970. American Psychologist; American Psychologist 26(2), 145 (1971)
27. Huet, G.: Confluent reductions: Abstract properties and applications to term rewriting systems: Abstract properties and applications to term rewriting systems. Journal of the ACM (JACM) 27(4), 797–821 (1980)