Exploratory study of the interaction between action selection learning and goal selection using the PRIMs model

June 2020, Master Thesis

Author Information	Supervisor Information
Full Name: Alice Dauphin Title: Master Student Affiliation: Ecole Nationale Supérieure des Arts et Métiers Email Address: asedauphin@gmail.com	Full Name: Niels Taatgen Title: Professor, PhD Affiliation: University of Groningen - Faculty of Science and Engineering - Artificial Intelligence Bernoulli Institute - Cognitive Modeling Group Email Address: n.a.taatgen@rug.nl

Keywords: goal selection - production systems - reinforcement learning - PRIMs - transfert learning

Word Count: 10394

Arts et Métiers Sciences et Technologies

university of groningen

Abstract

The question of how humans (or more broadly, autonomous agents) learn to select the next action they are going to perform to achieve a given goal has been extensively studied. On the contrary, the question of how was the goal selected in the first place remains little explored. The present work aimed at exploring possible goal selection functions and their interaction with the action selection learning process. The PRIMs cognitive architecture was used to implement and test the goal functions I designed.

By using a local measure of the agent action selection learning progress to drive the goal selection process, an economical goal ordering emerged spontaneously. I compared the performance of this goal selection function with a random goal selection and showed that the learning performance of the model had improved thanks to a better distribution of the temporal resources.

Table of Contents

1. Introduction

In search of possible goal selection functions

Let's imagine a child facing the menu of a gaming app on a tablet computer. This is the first time he opens a gaming app and he has to choose among seven games represented with seven different icons. What motivates this child to choose one particular game instead of another? That is, on which criteria does he make the decision? Maybe the graphics? The music? One week later, the same child is once again facing the gaming app menu. Across the week, he played with the different games. This time, does he take his decision based on the same criteria he used one week ago? Certainly not. Which additional information does he take into account? How are the different pieces of information combined? What are the alternative ways to combine them? Why are they finally combined this way instead of another?

The above scenario illustrates the research problem I tackled for my master thesis. This problem stems from the three following questions:

{1} What is a goal? In a simplified version of the scenario, when the child selects a game, he selects a goal: to win the chosen game.[1] The current goal of an agent can be defined as a state, different from the current state, the agent tries to reach by its actions.

{2} What needs to be explained to understand goal processing? The child has to first generate different achievable goals before selecting a particular one. To understand goal-directed agents such as humans, one needs to explain "how are the goals produced in the first place?" (goal generation problem) and "what are the mechanisms and information the agent uses to select between conflicting goals?" (goal selection problem) (Hawes, 2011).

{3} How do action selection learning and goal selection interact? What mutual influences exist between these two processes? In the scenario, the child selects a game from which he can potentially learn. It seems probable that the information he uses to make his decision differs from the one he uses to decide whether he should go to the bathroom or to the kitchen to wash his hands. The goal selection process could help choosing the task from which learning will be the most effective for example.

I chose to investigate question {3} and simplified it as follows: **What goal selection processes can improve action selection learning?** My goal was to design possible goal selection functions and study their influence on action selection learning. To do so, I distinguished (artificially) between action selection and goal selection processes. To simplify even more the problem, I assumed that the agent I modeled could only learn by trial and error. Therefore, this study excluded completely situations in which social interactions could lead the agent to imitate someone or follow advice. By

[1] In fact, once a game is selected, different goals can be considered, the child might choose to beat his personal best score on that same game, to double his score or to end the game...

doing that, I restricted the domain of validity of my model to children under the age of six. This corresponds to the generally accepted limit in child development studies under which autonomous play represents an important part of learning, much more important than formal education. As I studied exclusively action selection learning, namely the acquisition of knowledge about how and under which circumstances to execute internal or external operations, throughout this document the term "learning" will refer exclusively to action selection learning[2].

The research to understand action selection learning has been largely dedicated to the problem of how an agent selects its actions in order to achieve a given goal. The question of "how was the goal chosen in the first place?" remains little explored. To investigate this question could shed light on both the action selection and the goal selection processes by highlighting (or on the contrary discarding) differences between these two functions and by offering for both processes new mechanisms to look for in the brain[3]. A good understanding of the motivational system in humans would have powerful practical implications, among them the possibility to increase human performance (R. O'Reilly, 2018).

Requirements for goal management functions

As my goal was to design possible goal selection functions, I first looked for functional specifications. The CogAff project[4] on which Sloman and his collaborators worked from 1991 to 2011 provided me with an extensively developed conceptual framework. At the time, their goal was "to understand the types of architectures that are capable of accounting for the whole range of human (and non-human) mental states and processes". Among others, they investigated the motivational aspects of cognition. They suggested that emotional states are a consequence of design requirements and constraints in resource-limited intelligent agents (like humans), not an accident of animal evolution. From this perspective, they studied intensively goal processing in autonomous agents (Beaudoin, 1994; Hawes, 2011). They identified essential requirements for the goal selection function. I sampled three of them here:

[2] The notion of action selection knowledge overlaps with the notion of procedural knowledge. Procedural knowledge is often defined as information about how to execute behaviors (e.g. how to drive or how to do long division) and is contrasted with episodic and declarative knowledge, namely information about previous events that were experienced (or later interpreted as having been experienced) and factual information about the world.

[3] This idea that conceptual work is of primary importance is expanded by Minsky in his lectures. I sampled the following quotation to summarize his point: "You can't look for something until you have the idea of it." (Minsky, 2011)

[4] For additional information, see: https://www.cs.bham.ac.uk/research/projects/cogaff/

{1} Goal generation must be a process able to react to sudden changes, be it internal or external. An agent unable to adapt its goal to face the urgency of a situation would not survive. As a result, new goals can be generated at any time and the goal selection function must be able to engage its decision procedures similarly.

{2} The output of the goal selection function can be either to interrupt the current behaviour with a new goal or to allow the current behaviour to continue.

{3} The objective of the goal selection function is to identify the goal that is both achievable and the most beneficial for the agent. As a consequence, two processes have to be combined: one that rejects goals which cannot be achieved (through planning-like reasoning) and another that analyses the costs and benefits of the plan generated to achieve the goal.

As {1} requires to study the class of processes that can be interrupted at any time and produce sensible results (anytime algorithms, see Dean & Boddy, 1988) I did not try to meet this requirement in the present work. {2} was not directly tackled in this work and remains a challenging point for future research (see Discussion). {3} raises two questions: (a) how to check goal achievability? (b) how to measure the costs and benefits of achieving a goal? This thesis focuses on question (b). The literature presented below is a limited selection of inspirations to answer the question: what are the relevant signals to compare goals in order to maximise the learning performance of the agent?

Cognitive need(s) that may drive goal processing

Interestingly, a proposition of goal selection function can be found in the MicroPsi architecture[5] (Bach, 2009). The MicroPsi architecture also provides an example of goal generation function.

In MicroPsi, as in many theories (Sloman et al., 2005; Minsky, 2007), goals are derived from predefined needs. The agent has to satisfy its needs before an irrevocable point is reached (Bach, 2015). Three types of needs are distinguished: physiological needs, social needs, and cognitive needs. The needs are modeled as tanks. They have minimum and maximum values and the distance between the current value and the extreme values defines the urge of the need. In that scheme, goals are actions or situations that can either satisfy the urge (appetitive goals), or increase the urge (aversive goals).

Bach distinguishes between three types of cognitive needs: (1) the need for competence, (2) the need for exploration and (3) the need for aesthetics. A competence being either task-related, effect-related or general, (1) is made of: (a) the need for task-related improvement, that is the need for making progress at a given task, (b) the need for causing changes in the environment and (c) the need for general abilities to satisfy any type of needs. Exploration (2) is the need for uncertainty reduction,

[5] MicroPsi is a cognitive architecture derived from a framework for designing and simulating cognitive agents based on the Psi theory of Dietrich Dörner.

namely the need to acquire knowledge about objects and processes. Aesthetics (3) is made of (a) stimulus-oriented aesthetics, the need for predefined stimuli (e.g. specific tactile sensations such as softness) and (b) abstract aesthetics, the need for extracting structure in mental representations and improving existing mental representations.

The cognitive needs described above suggest different dimensions to compare goals[6]:

- (1.a) Competence within a task: Which goal maximizes (learning) progress within a task?
- (1.b) Change in the environment: Which goal maximizes change in the environment?
- (1.c) General competence: Which goal maximizes (learning) progress across tasks?
- (2) Exploration: Which goal maximizes uncertainty reduction/acquisition of new knowledge?
- (3.b) Abstract aesthetics: Which goal maximizes the creation of new (useful) mental representations and the optimisation of the existing ones?

The above descriptions need to be turned into quantifiable measurements to be suitable for use. The section below presents a formalism from the reinforcement learning literature to the dimension (1.a).

How to measure the learning progress: the Intelligent Adaptive Curiosity (IAC)

The reinforcement learning paradigm has been extensively developed in artificial intelligence and developmental robotics to build agents able to learn, from their interaction with the environment, regularities between their perceptions and actions (sensorimotor mapping). In this research field, tools have been designed to measure the learning progress of agents. These tools were created to solve the problem of "how to collect data efficiently in an open environment?" or, in other words, "how to explore both extensively and with limited resources the sensorimotor space?" Indeed, the agent's sensorimotor space is often unbounded and high- dimensional meaning that there are much more potentially learnable skills than what the agent can discover in its lifetime. As a result, the question of how to explore and what to learn is essential (Baldassarre et al., 2010). I took direct inspiration from the work done in this domain to design a "progress driven" goal selection function (see Methods).

The Formal Theory of Creativity, Fun and Intrinsic Motivation (Schmidhuber, 2010) first introduced the idea of a reward proportional to the prediction progress of the agent. This theory assumes that children and scientists understand a new phenomenon when they find an efficient way to compress

[6] One can argue that physiological and social needs also drive goal selection in learning contexts (e.g. mentor) which is certainly true. Even so, time being a limited resource, the above hypotheses could not be investigated.

data related to this phenomenon[7]. As a consequence, an (artificial) intelligent agents should be rewarded each time it compresses data, that is each time it finds a shorter program to encode data[8]. The amount of reward should be proportional to the amount of compression performed so that the agents maximizes the compression rate.

Inspired by the above theory, Oudeyer and his collaborators designed an agent able to learn complex skills autonomously (in comparison with what could be done at the time) from its environment (Oudeyer et al., 2007). They biased the agent's exploration of the environment by introducing a reward proportional to the prediction progress of the agent using a motivational system called Intelligent Adaptive Curiosity (IAC). IAC aimed at computing a measure of the learning progress (LP) in predictions of the agent. It was made of a learning machine M that learnt to predict the sensorimotor consequences when a given action is executed in a given context and a meta-learning machine metaM that learnt to predict the errors that machine M made in its predictions (Fig 1). LP was the difference between <E(t+1)>, the expected mean error rate of the predictions of M in the close future predicted by metaM, and <E(t)> the mean error rate in the close past. As a result, LP was an estimate of the local derivative of the error rate curve of M. When LP was positive, the error rate increased meaning that the situation did not help the agent to learn to predict the consequence of its actions. On the contrary, if LP is negative the agent was making progress.

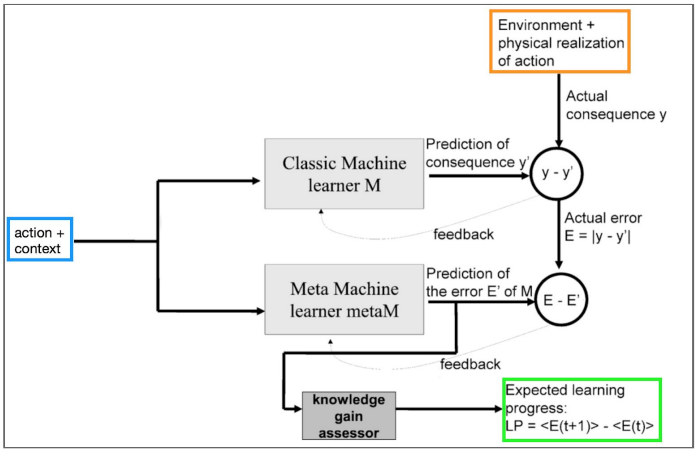

Figure 1. Motivational system of a reinforcement learning agent, IAC (adapted from Oudeyer et al., 2007). A learning machine M learns to predict the sensorimotor consequences when a given action is executed in a given context (input, blue box). A meta-learning machine metaM learns to predict the errors that machine M makes in its predictions. The signal of interest here is the learning progress (LP, green box) computed from the predicted error generated by metaM.

The agent learnt using reinforcement learning. At each timestep, it was rewarded by the additive inverse of LP: the reward was positive when the error rate decreased, and negative when it increased. As a result, the agent selected the action that would lead to the greatest decrease of the mean error rate of M (according to the predictions of metaM). By doing that, the agent could not stay stuck in front of

[7] It is assumed that the mechanisms that produce new discoveries on the scale of a lifetime are the same mechanisms that produce new discoveries on the scale of human history (Boden, 2007; Sloman, 1978).

[8] Not all data are compressible, for example, white noise on a screen does not exhibit regularities and therefore it is not possible to predict it whatever the amount of data collected beforehand.

unlearnable situations as it could not decrease its errors in prediction in these situations. From time to time a random selection of action was done to avoid ignoring a situation in which the agent could have made better progress (ε-greedy action selection rule, Sutton & Barto, 2014).

From IAC, I imagined a progress driven goal selection function, namely a function that would choose the next goal so that it maximizes the learning progress of the agent. To do so, an estimate of the learning progress was necessary. In IAC, the learning progress LP in fact corresponded to a learning progress in prediction which required many modules to be possible. In order to first test a simplified version of it, the progress driven goal selection function was conceived to select the goal for which the agent made the most progress in the close past. For each goal, the opposite of the derivative of the past agent performance was used to measure the learning progress of the agent (a formal mathematical description can be found in the Methods). By doing so, the agent should first select easy goals, that is goals for which it made a lot of progress and then, as the performance stagnates and the derivative tends towards zero, easy goals would be less and less often selected. The knowledge that the agent acquired when completing easy goals could then be transferred to achieve harder goals. The learning curriculum of the agent would organise spontaneously from easy to harder goals. To test this hypothesis I implemented the scenario of the child facing a gaming app menu using the PRIMs cognitive architecture (Taatgen, 2013). The following section explains why.

Cognitive Modeling & PRIMs

Cognitive architectures are models of the mind developed to build testable theories of mental processes (Bach, 2009). Within an architecture pieces from different fields of cognitive science are assembled in a formal and coherent framework, then translated into a computer program. The integration does not merely consist in concatenating what is already known, it also implies delving into the additional requirements (and the necessary functions to meet these requirements) of such an assembly. As a result, the process can produce back new insights and predictions.

To study the continuous interaction between goal management and action selection across different tasks I naturally headed towards cognitive architectures. Precisely, I headed towards PRIMs (Taatgen, 2013), a symbolic architecture derived from the ACT-R architecture of John Anderson (Anderson, 2007). Symbolic architectures assume that the knowledge used in (humans) brains can be modeled in a relevant and efficient way at a particular level of abstraction in which important features are captured and details removed. At that level of abstraction, knowledge is captured in the form of symbols (Goel & Joyner, 2019). PRIMs belongs to a particular type of symbolic architecture: production systems. Production systems were called in reference to the set of "productions" they use (Boden, 2006). Productions are IF–THEN rules. Each rule specifies (one or more) actions to be taken in response to (one or more) particular cues, or conditions.

The specificity of PRIMs is to offer two levels of granularity for productions: “operators” and “primitive operations”. Primitive operations are a set of basic productions that can be combined to make operators. Primitive operations have been successfully associated with a learning process enabling the model to build its operators autonomously instead of having them manually built by the modeler. Together, they are an attempt to answer the question “how were the operators built in the first place?” and provided a proof of concept that new operators in production systems can be created over interactions between the agent and its environment using reinforcement learning (Ji et al., 2019). However, PRIMs does not offer any mechanism for goal selection making it an ideal blank slate to investigate possible goal selection functions.

2. Methods

The results of this study were generated using the PRIMs architecture developed by Taatgen upstream from my internship. The two following subparts present the general functioning of the model and the processes through which new operators are created and associated with a goal. Readers who want to skip these parts can jump directly to the experiments I designed and conducted along with their analyses.

The PRIMs cognitive architecture

The Primitive Information Processing Elements (PRIMs) architecture (Taatgen, 2013) was built on the top of the Adaptive Control of Thought–Rational (ACT-R) architecture (Anderson, 2007). As a consequence, it shares most of the ACT-R assumptions:

{1} The organization of the mind is modular. PRIMs has specific modules for vision, motor control (manual module), goal maintenance, declarative memory and working memory (Fig. 2, right).

{2} Modules can operate in parallel, but can only do one operation at a time.

{3} The communication between modules is carried out through their buffers (white ring in Fig. 2, right). Each buffer is made of slots. Each slot can hold a single piece of information. All the buffers together comprise the (global) workspace of the system.

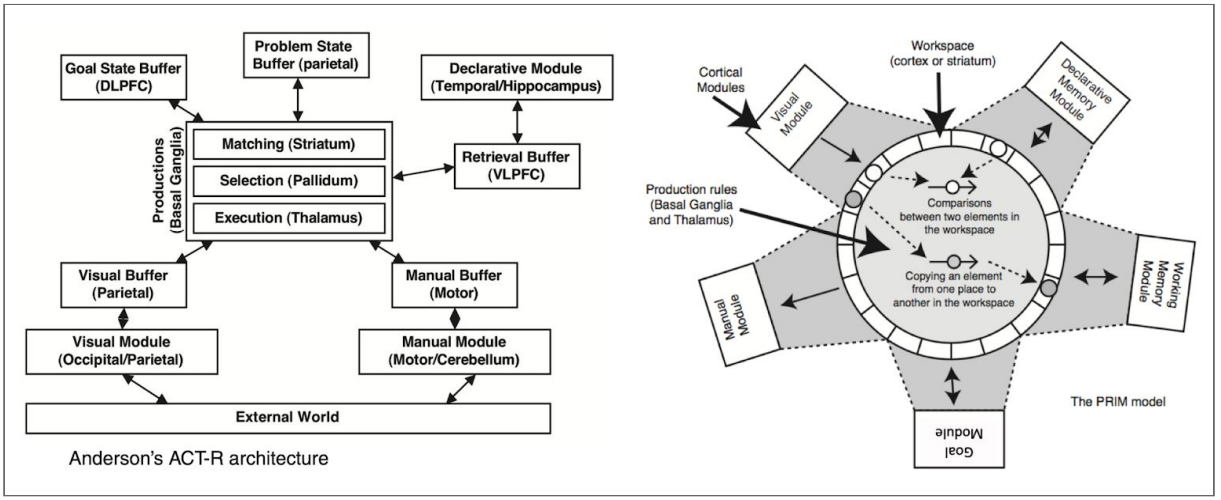

Figure 2. The PRIMs architecture (right) is directly inspired from the ACT-R architecture (left), DLPFC: dorsolateral prefrontal cortex; VLPFC: ventrolateral prefrontal cortex (reproduced from Taatgen, 2013)

{4} A piece of knowledge (declarative or procedural) is a chunk. The probability to retrieve a chunk from memory is computed from its activation A_i, the chunk with the highest activation being retrieved (to know more about how activations are computed, see Appendix A).

The core PRIMs theory focuses on how information is exchanged within the workspace. A rolled-out version of the workspace can be seen in figure 4 (a). The workspace is made of the basal ganglia and

the thalamus. Basal ganglia control the routing of information between cortical areas and are able to bind pairs of source and destination regions (abstraction built on the top of Stocco et al., 2010).

Bottom-up learning in PRIMs: operator creation and goal-operator association

Within the workspace a set of elementary production rules (primitive operations) copy and compare the information from different slots as illustrated by the dotted arrows in the center in figure 2, right. Primitive operations can be combined to form operators and operators can be concatenated to achieve a given goal (Fig. 3).

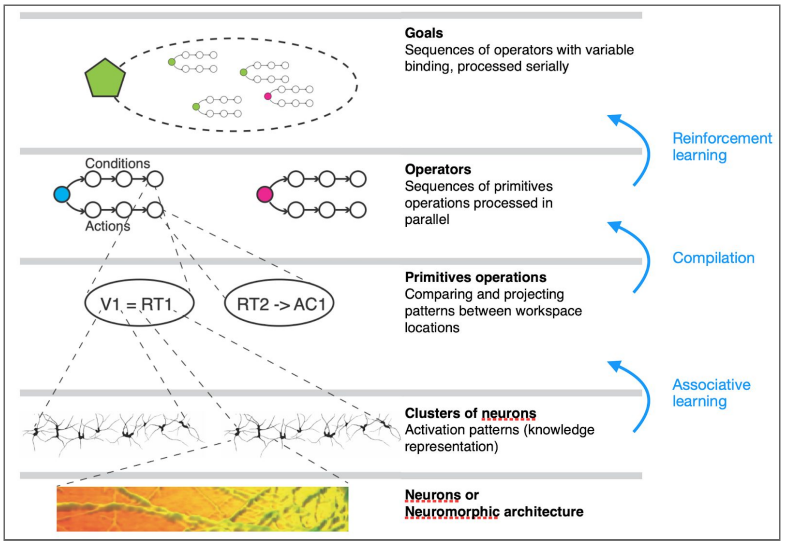

Figure 3. The knowledge hierarchy in PRIMs (adapted from N. Taatgen, 2019). Each layer represents a different level of abstraction from the neuronal implementation (bottom) to the goal representation (top). The representations within a level of abstraction can be learnt on the top of representations from lower levels of abstractions through different mechanisms (blue arrows).

Each time a succession of primitive operations succeeds at achieving a goal a new operator is created (compilation). In addition, the activation between the goal j and the newly created operator i is strengthened (reinforcement learning). The equations for the update are:

$$S_{ji} = S_{ji\,(previous\ trial)} + \Delta S_{ji} \qquad (1)$$

$$\Delta S_{ji} = \beta * (payoff - S_{ji\,(previous\ trial)}) \qquad (2)$$

$$payoff = default\ association * \frac{(maximum\ expected\ time - actual\ time)}{expected\ time} \qquad (3)$$

In these equations, the parameters β , default association, expected time are fixed.

The strength of the association between the goal and the operator, Sji, converges toward the default association parameter over the trials.

To illustrate the entire process, let's look at the paired-associate task. In this task, words are associated with different fingers. The model is shown a word and has to press the corresponding finger to be rewarded. For example, the word "vanilla" is associated with the thumb. In his declarative memory,

the model has at his disposal different pieces of knowledge about “vanilla” it had previously learnt such as: {f1} associate, vanilla, thumb; {fa1} taste, vanilla, sweet; {fb1} color, vanilla, yellow. To succeed at the paired-associate task, the model has to retrieve {f1} when cued by the word “vanilla” and copy the “thumb” element to the manual buffer for the model to execute the correct action.

When the task begins, only two slots of the workspace are occupied: V1 (visual buffer) and G1 (goal buffer). V1 contains the visual input, here the word “vanilla” represented with a red disc in figure 4(a) and G1 contains the goal associated with the paired-associate task, here “to solve the task” represented with a green disc. If the model selects the primitive operation V1→RT2, the item “vanilla” is copied to the second long-term memory slot RT2, this situation corresponds exactly to the figure 4(a). Then, the model has to retrieve another primitive operation until the task ends. If the sequence of operations it retrieved led to success, new operators are created by concatenating primitive operations (brown discs, Fig. 4(b)). Each time a sequence of production leads to success the association between the goal and each production is reinforced using formula (1), (2) and (3). The productions that are positively associated with the goal after learning can be visualized in figure 4.(b). Two new operators (V1toRT2+C1toRT1 and RT1equalC1+RT3toAC1) are associated with the goal, they are "the solution".

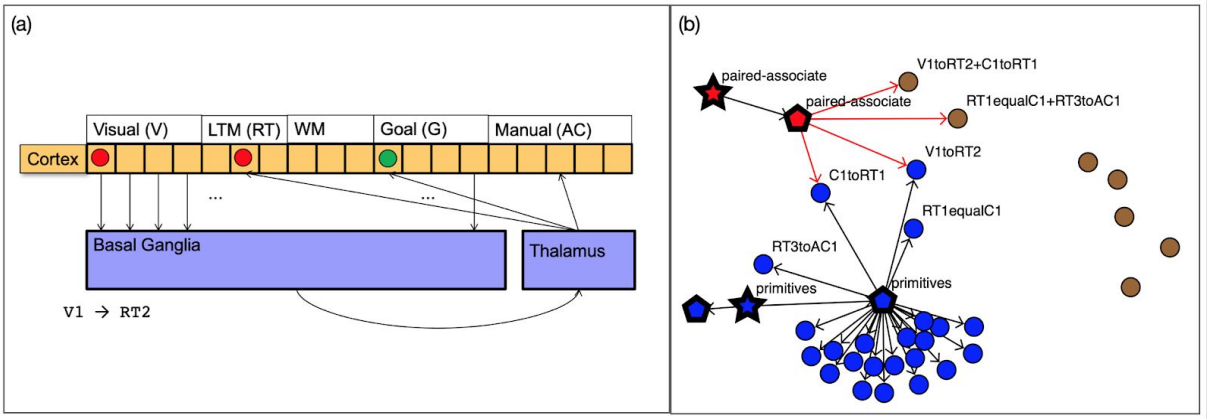

Figure 4. What happens during a simulation in PRIMs: (a) Rolled-out view of the workspace in which the primitive operation V1→RT2 has been executed. The piece of knowledge (red disc) that was initially only in the visual buffer (V1) has been copied to the long-term memory buffer (RT2). (b) Productions positively associated with the goal “solve paired-associate task”. Primitive operations are represented with blue circles, operators created during the learning process with brown circles, the positive associations between the goal and productions with red arrows.

Three types of simulations: control, random and progress driven

This section summarizes the experimental conditions I designed. Three types of simulations were run: **control** (no goal selection), **random** goal selection and **progress driven** goal selection. The table 1, below, summarizes the simulations made.

Condition	Goal Selection	Reset	Number of trials	N
Control 1	Ø	True	10 000*7	100
Control 2	Ø	True	1430*7	100
Control 3	Ø	False	(1330 + 100)*7	100
Random	random	False	10 000	100
Progress driven 1	progress driven, $d = 30$	False	10 000	100
Progress driven 2	progress driven, $d = 50$	False	10 000	100
Progress driven 3	progress driven, $d = 80$	False	10 000	100
Progress driven 4	progress driven, $d = 130$	False	10 000	100

Table 1. Summary of the simulations run. The first column indicates the name of the simulation. The second column indicates the goal selection function and the distance parameter for progress driven simulations. The third column indicates if the model was reset between goals. The fourth column indicates the number of trials within a simulation. For control simulations the number of trials is the product of the number of trials per goal (10 000 or 1430) and the number of tasks (7). The fifth column contains the number N of simulations done.

In the **control** simulations, the model executed a sequence of eight goals previously set by the modeler. Each goal was repeated 10 000 or 1430 times before the model had to switch to the next one. 1430 corresponds to the average number of trials in which a goal was selected in the random condition (10 000 trials in total, eighth tasks $\Rightarrow$ each task was selected $\sim \frac{10\,000}{7} = 1430$ times). In Control 1 and Control 2 the model was reset at each goal change, that is the history of the model was erased before a new learning occurred. In Control 3, as for the remainder of the simulations, there was no reset of the model within a simulation, meaning that the learning had to be optimized for the set of tasks instead of a single task. In addition, in Control 3, after the seven goals had been repeated 1330 times without reset, goals were picked again 100 times in a random order (to have a better idea of the goal dynamics over trials in this condition, see figure 7 top right). This was done to test the learning performance of each goal not just after the repeating of that specific goal but after all goals had been trained.

In the **random** simulations, a new goal was picked at random after each trial and added to the goal buffer. On average, each goal was selected in one seventh of the trials, that is $\frac{10\,000}{7} = 1430$ times.

In the **progress driven** simulations, a competition between goals took place after each trial. The goal g for which the model had the highest learning progress r_g was selected. If r_g was negative or close to zero ($r_g \leq \frac{0.1}{d}$), meaning that the learning performance of the model was declining or stagnating r_g was set to zero. If the model succeeded successively at the same goal for at least $d + 20$ trials r_g was set to -1. This was done to switch off learning once the goal was mastered. When r_g was the

same for all goals, a goal was picked at random. To evaluate r_g I used the derivative of the accuracy over the past d trials. The accuracy of the model for a given goal at trial t corresponded to its success rate over the past n trials in which the goal was selected ($n = 10$ for all progress driven simulations):

$$Accuracy_{g,t} = \frac{\sum_{i=1}^{n} \delta_{g,t-i}}{n} \qquad (4)$$

with $\delta_{g,t-i} = 1$ if the model achieved the goal and $\delta_{g,t-i} = 0$ if the model failed at trial $t - i$.

The first order derivative of the accuracy function was estimated as follows:

$$r_g = \frac{\Delta Accuracy_g}{\Delta trials_g} = \frac{Accuracy_{g,t} - Accuracy_{g,t-d}}{d} \qquad (5)$$

with d the number of trials between the accuracy in the close past ($Accuracy_{g,t}$) and the accuracy in the distant past ($Accuracy_{g,t-d}$).

The threshold chosen to consider that r_g should be set to zero ($r_g \leq \frac{0.1}{d}$) corresponds to a difference of accuracies of 0.1, that is one trial of difference between $Accuracy_{g,t}$ and $Accuracy_{g,t-d}$.

To include the goal selection functions within the PRIMs architecture, I created a clone version of the model I could modify (PRIMs is developed using the swift programming language)[9].

The seven goals and their paired tasks

At each trial, the model had to select a goal among seven (except in the control conditions, see above). Each goal was paired with a single task so that goals consisted of succeeding at their paired task. Table 2 provides a brief description of the tasks used and summarizes their main characteristics.

Task	Minimum number of primitive operations	Optimal sequence of primitive operation that guarantees success	Description of the task
recall*	2	V1 → WM1 WM1 → AC1	The model is shown a stimulus, has to store it in working memory and then has to report it.
press-key*	3	V1 → WM1 next-goal → G2 WM1 → AC1	The model has to perceive what finger needs to be pressed, then switch to the "press-key" goal, and then press the key.
paired-associate*	3	V1 → RT2 C1 → RT1 ($p = \frac{1}{3}$) (RT1 == C1) RT3 → AC1	The model is shown a word. It has to retrieve the fact indicating the corresponding finger (fingers and words were previously linked) and press it.
complete	3	V2 → RT2 C1 → RT1 ($p = \frac{1}{2}$) (RT1 == C1) RT3 → AC1	Two adjacent numbers are displayed. The model has to pursue the enumeration, i.e. it has to retrieve the fact containing the next number from the second number displayed.

[9] This modified version of PRIMs is available at: https://github.com/asedauphin/PRIMs_cmd.

category*	5	V1 → RT2 C1 → RT1 ($p = \frac{1}{2}$) RT3 → RT2 C1alt → RT1 ($p = \frac{1}{2}$) (RT1 == C1alt) RT3 → AC1	In this task a living thing is displayed. The model has to decide whether it is a plant or an animal, and has to press left for an animal, and right for a plant.
attribute	3	V1 → RT3 C1 → RT1 ($p = \frac{3}{4}$) (RT1 == C1) RT2 → AC1	In this task an attribute is displayed. The model has to retrieve in memory an item with the corresponding characteristic and say it.
what-comes-before	3	V1 → RT3 C1 → RT1 ($p = 1$) (RT1 == C1) RT2 → AC1	The model is shown a number between 1 and 10 and has to answer "what number comes before this one?"

Table 2. Summary of the different tasks used. The first column contains the name of the task. The second column contains the minimal number of primitive operations necessary to successfully solve the task. The third column contains the shortest sequence of primitive operations that guarantee success. The primitive operations RT1 == C1 and RT1 == C1alt are in parentheses because they are not necessary to solve the task but are required for the compilation to succeed and therefore the model has to learn to use them before compilation. The probability p (next to the operations C1 → RT1 and C1alt → RT1) represents the probability of a successful retrieval if all the operations above have been done. The fourth column contains a verbal description of the task.

Tasks marked with asterisk in Table 1 were designed by Taatgen upstream from this research. I extended this set of tasks to offer additional tasks of middle difficulty (using the number of primitive operations necessary to solve a task as a first indicator of its difficulty) and tasks where the transfer of new compiled operators could be done, i.e. tasks in which the optimal sequence of primitive operations that solve them overlap entirely (e.g attribute and what-comes-before) or partially (e.g paired-associate and complete).

Analyses

First, to quantify the influence of goal selection processes on action selection learning, two questions were asked: (1) In which condition(s) does the model converge mostly? (2) In which condition(s) does the model converge the most rapidly? In addition, for each of these two questions, the following one was also considered: How significant is the contrast relatively to other conditions?

To assess whether or not the model converged towards the optimal solution for a given goal, I looked at its accuracy for the last 50 trials in which it performed that task. An accuracy superior or equal to 0.9 during these 50 trials was sufficient to consider that the model converged.

Then, exploratory analyses were performed to understand the differences found in the group analyses. The method used here consisted of sampling 10 simulations and looking at the ordering of goals, the accuracy over trial per goal, and the evolution of the probability of transition between primitive operation per goal. I designed and implemented new tools to visualize the goal dynamics and the probability of transition between primitive operations using python. The corresponding code is available at: https://github.com/asedauphin/Goal-Selection-PRIMs.

3. Results

Three levels of difficulty among goals

Control 1 was used to measure and compare the level of difficulty of the goals. Given the performance of the model (Fig. 5), the goals paired-associate, press-key and what-comes-before (Fig. 5, number of convergent simulations $= 100$) can be considered easy goals, complete and recall (Fig. 5, number of convergent simulations ≥ 80) goals of medium difficulty, and attribute and category (Fig. 5, number of convergent simulations ≥ 50) hard goals.

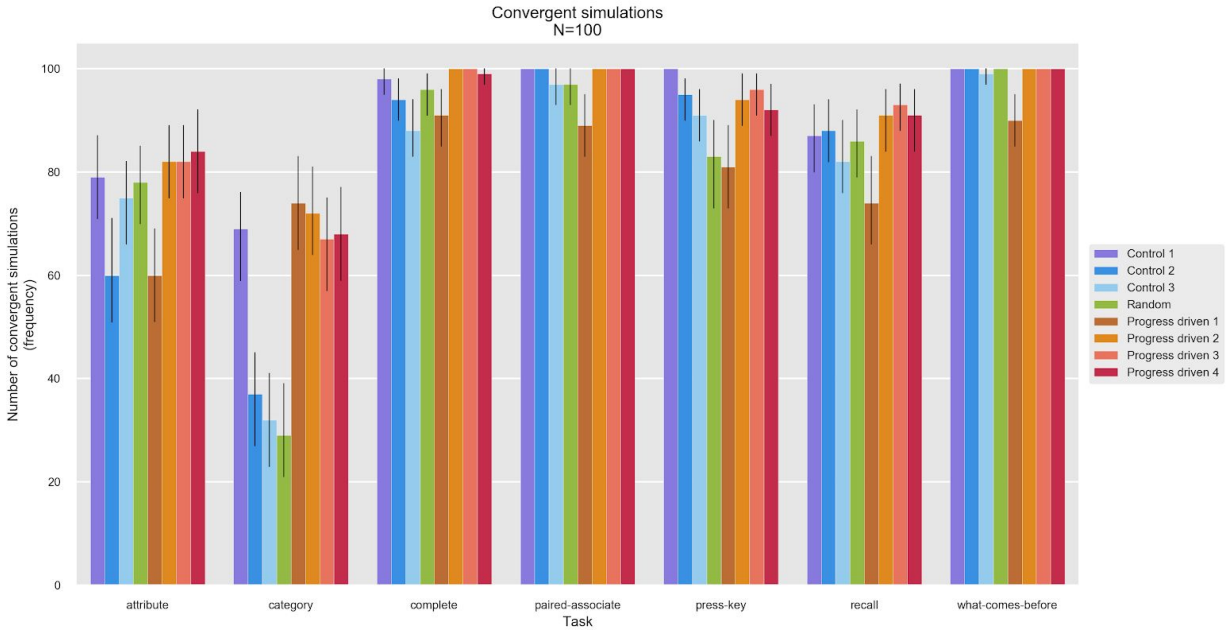

Figure 5. Number of convergent simulations per task for the different conditions (control 1, 2, 3, 4: blue shades, random 1,2: green shades, progress driven 1, 2, 3, 4: orange shades). Within each condition, $N = 100$ simulations were done. The error bars (black) are the 95% confidence intervals for each value (number of bootstraps equals 150).

Control 2 converged as often and as quickly as Control 1 in easy and medium difficulty goals

Control 1 almost systematically converged before 1430 trials in complete, paired-associate, press-key and recall goals (Fig. 6, Appx. B Fig. 10). Therefore, it was coherent that both the number of convergent simulations (Fig. 5) and the distribution of trials from which the model converged (Fig. 6) did not differ for these goals between Control 1 and Control 2 conditions. For the goal what-comes-before, the distribution of trials from which the model converged spread above 3000 trials in the Control 1 condition (Fig. 6) although the model converged systematically in both Control 1 and Control 2 conditions. This result suggests that the model made occasional mistakes long after learning occurred. This hypothesis was corroborated by the inspection of the accuracy over trials within 10 simulations picked at random, figure 11 (Appx. B) shows one of them. It is also coherent with the numerous outliers seen in figure 6 for the Control 2 condition. Consequently, for the goals complete, paired-associate, press-key, recall and what-comes-before, we can reasonably conclude that

when the model did not converge after 1430 trials (Control 2 condition) it did not need additional trials to do so. On the contrary, for goals attribute and category, additional trials improved the model performance (this hypothesis was further corroborated by the analysis of the progress driven conditions).

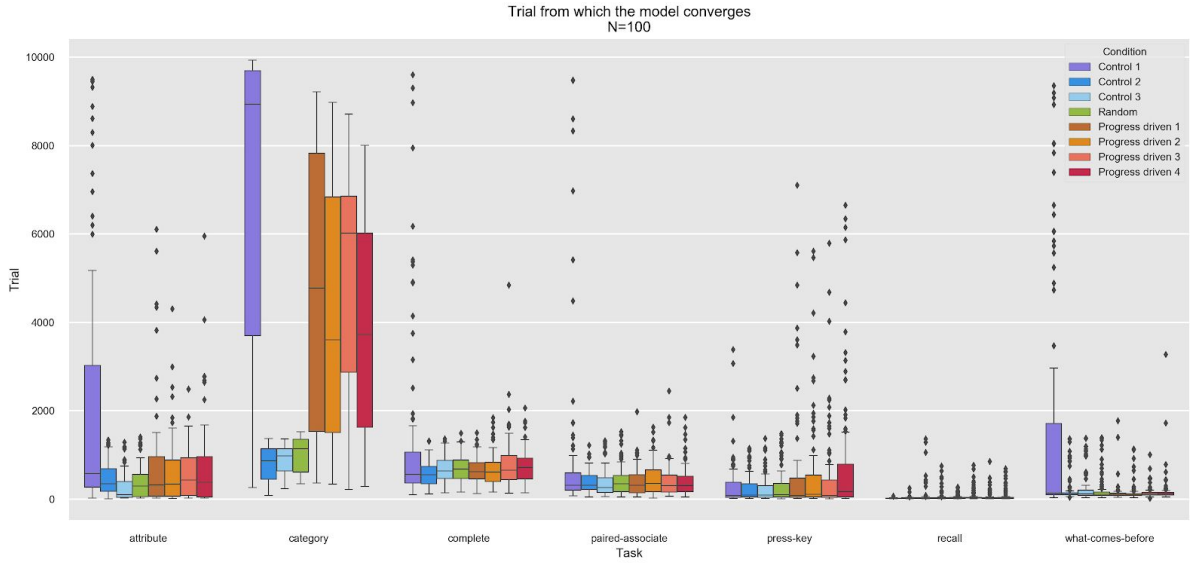

Figure 6. Rank of the first trial from which the model converged per task for the different conditions (control 1, 2, 3: blue shades, random: green, progress driven 1, 2, 3, 4: orange shades). Within each condition, $N = 100$ simulations were done. The numbers of simulations that converged and from which ranks were calculated are: (78, 68, 97, 100, 100, 86, 100) for Control 1, (60, 36, 93, 100, 94, 87, 100) for Control 2, (74, 31, 87, 96, 90, 81, 98) for Control 3, (77, 28, 95, 96, 82, 85, 100) for Random, (60, 73, 90, 88, 81, 73, 89) for Progress driven 1, (81, 71, 100, 100, 93, 90, 100) for Progress driven 2, (81, 66, 100, 100, 95, 92, 100) for Progress driven 3, and (83, 67, 98, 100, 91, 92, 100) for Progress driven 4, per tasks (ranked by alphabetical order). A zoomed version of this figure can be found in Appendix B.

Limited positive interference between the goals what-comes-before and attribute in the Control 3 condition

Control 3 was created to compare random and progress driven conditions to a control condition where the goals could interfere (positively or negatively). Recall that in Control 1 and Control 2 goals were independently learnt (reset of the model after each goal, see figure 7 top left). I expected Control 3 to converge more often and more quickly than Control 2 in goals where transfer learning was possible and desirable. In particular, attribute was designed so that the optimal sequence of primitive operations to solve it was the same as what-comes-before (see Table 2), a goal in which the model succeeded almost systematically. Control 2 showed a little advantage over Control 3 for goal attribute. The limited effect was coherent with the random ordering of goals in the Control 3 condition[10] but was called into question by the similar performance of Control 3 and Progress driven 3 conditions as explained below.

[10] What-comes-before was learnt before attribute with probability 0.5 and the model converged in about 60 simulations (IC95=[51.1, 68.5]) without any transfert learning (Control 2), therefore in Control 3 condition transfert learning may have occurred in (100-60)*0.5=20 simulations.

Little contrast among progress driven conditions

Between progress driven conditions, I expected to see differences that would help to set the d parameter in the equation (2) I used to compute the learning progress r_g. We see that Progress driven 1 shows quasi-systematically the worst performance. Between Progress driven 2, 3, and 4 there are slight differences but no clear advantage for one of these conditions. Progress driven 3 is the condition which differs the most from Progress driven 1. Consequently, in the following analyses, I chose to compare only Progress driven 3 to the control and random conditions.

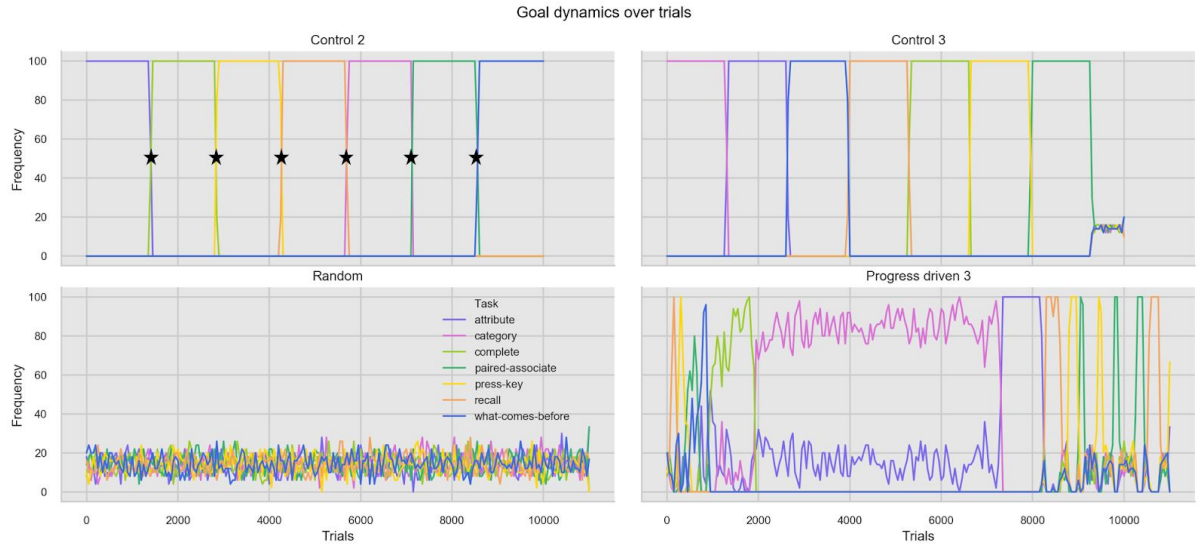

Figure 7. Example of the goal dynamics over trials within a simulation for the conditions Control 2 (top left), Control 3 (top right), Random (bottom left) and Progress driven 3 (bottom right). The goals are represented using color coding: attribute purple, category pink, complete light green, paired-associate green, press-key yellow, recall orange and what-comes-before blue. The black stars indicate the reset of the model. Figures 7, 8 and 9 were made from the same simulation.

Progress driven 3 converged more often than Random for the press-key goal

For the easy goals, I expected no difference between Random and Progress driven 3 conditions. Surprisingly, the model showed a statistically significantly better performance in the Progress driven 3 condition compared to the Random condition for the press-key goal (Fig. 5). An inspection of the accuracy over trials within 10 simulations picked at random (same procedure as above) revealed that the model made occasional mistakes long after learning occurred as it did for the what-comes-before goal. The mistakes seemed to occur less frequently for the press-key goal (not quantified). Again, this is coherent with the outliers found in the trials distributions (Fig. 6). In Progress driven 3 simulations, it happened that all goals have been learnt before 10 000 trials (e.g. Fig. 7 bottom right, the frequencies of the goals reached zero one after the other until the goal attribute did so around 8200 trials). In that case, a random goal selection took place until the model made a mistake in one goal and progress driven goal selection took place again. In the simulation shown in figure 7 it was the case for the goals paired-associate, press-key and recall. Depending on the seriousness of the mistake accuracy fell under 0.9 or not and the effect was detected or not (see Methods). For example, in figure 8 D2,

D3, D4 we see that the model made minor mistakes while trying to solve the task paired-associate. As the accuracy remained superior or equal to 0.9 these mistakes were not detected during the group analyses.

Progress driven 3 converged as often as Random for the attribute goal

For the goals attribute, and what-comes-before, that have the same optimal sequence of primitive operations but different probability of success for declarative memory retrievals (Table 2), I expected what-comes-before to have better performance than attribute in Random 1 condition but I did not expect any difference between the two goals in Progress driven 3 condition. Indeed, I expected the model to first learn to solve easy goals before transferring his knowledge to harder goals and as a result to succeed equally at both. However, the performance of the model for the goal attribute was equivalent in both conditions and the figure 8 A4 suggests that the model converged at the task attribute because it spent about 2000 trials trying to solve this task instead of 1430. This is coherent with the limited positive interference between the goals what-comes-before and attribute in the Control 3 condition.

Progress driven 3 converged more often than Random for the category goal

For the goal category, I expected a better performance in the Progress driven 3 condition where the model would stop selecting easy goals once they were mastered ($r_g = -1$) and therefore spend more trials learning about harder ones. The results support this hypothesis as the model showed statistically significantly better performances in the Progress driven 3 condition compared to the Random condition for the goal press-key and category (Fig. 5) and the distribution of trials jumped drastically (Fig. 6, median = 6000 trials).

The spontaneous goal ordering in the Progress driven 3 condition made better use of trials

As expected, the spontaneous goal ordering in the Progress driven 3 condition (which was the initial motivation to use the learning progress to drive goal selection) went from easy to harder goals. An example of this is shown in figure 7, bottom right (similar patterns were observed in all simulations sampled). Figure 8 was drawn from the same simulation as figure 7 and we can clearly see that when the tasks complete and paired-associate were learnt the model stopped dedicating additional trials to these goals (number of trials dedicated to each of these goals $\sim$1000). Our results confirm that an economical goal ordering can simply emerge from a local goal selection function (computed after each trial).

To converge the model needs to minimize the size of the space of productions

To visualize the effect of action selection learning on the space of productions the graph shown in figure 9 was drawn from the same simulation as figures 8 and 7. We see the probabilities of transition between primitive operations for the task category at different trials (Fig. 8, B4, purple dotted line). From a space including 15 primitive operations (A) the model converged to a space made of 8 ones

(C). Interestingly, we see that the solution found is robust (it cannot lead to mistakes) but not optimal (less transitions could be used for the same result).

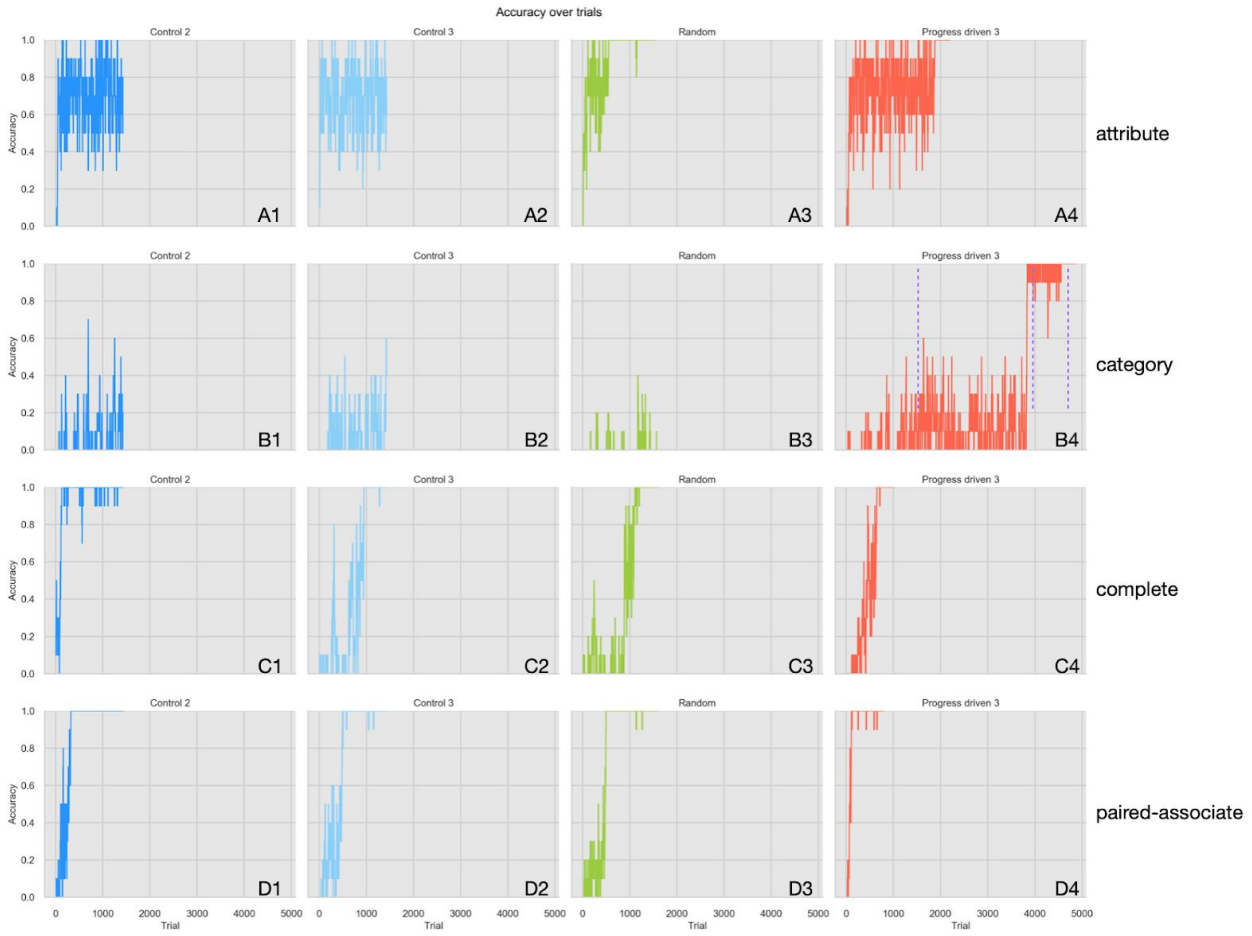

Figure 8. Example of the dynamics of the accuracy over trials within a simulation for the different conditions: Control 2 (first column), Control 3 (second column), Random (third column) and Progress driven 3 (fourth column). For each condition, only four tasks were shown: attribute (first row), category (second row), complete (third row) and paired-associate (fourth row). The three purple dotted lines in graph B4 indicate the trials from which figure 9 was drawn as figures 7, 8 and 9 were made from the same simulation.

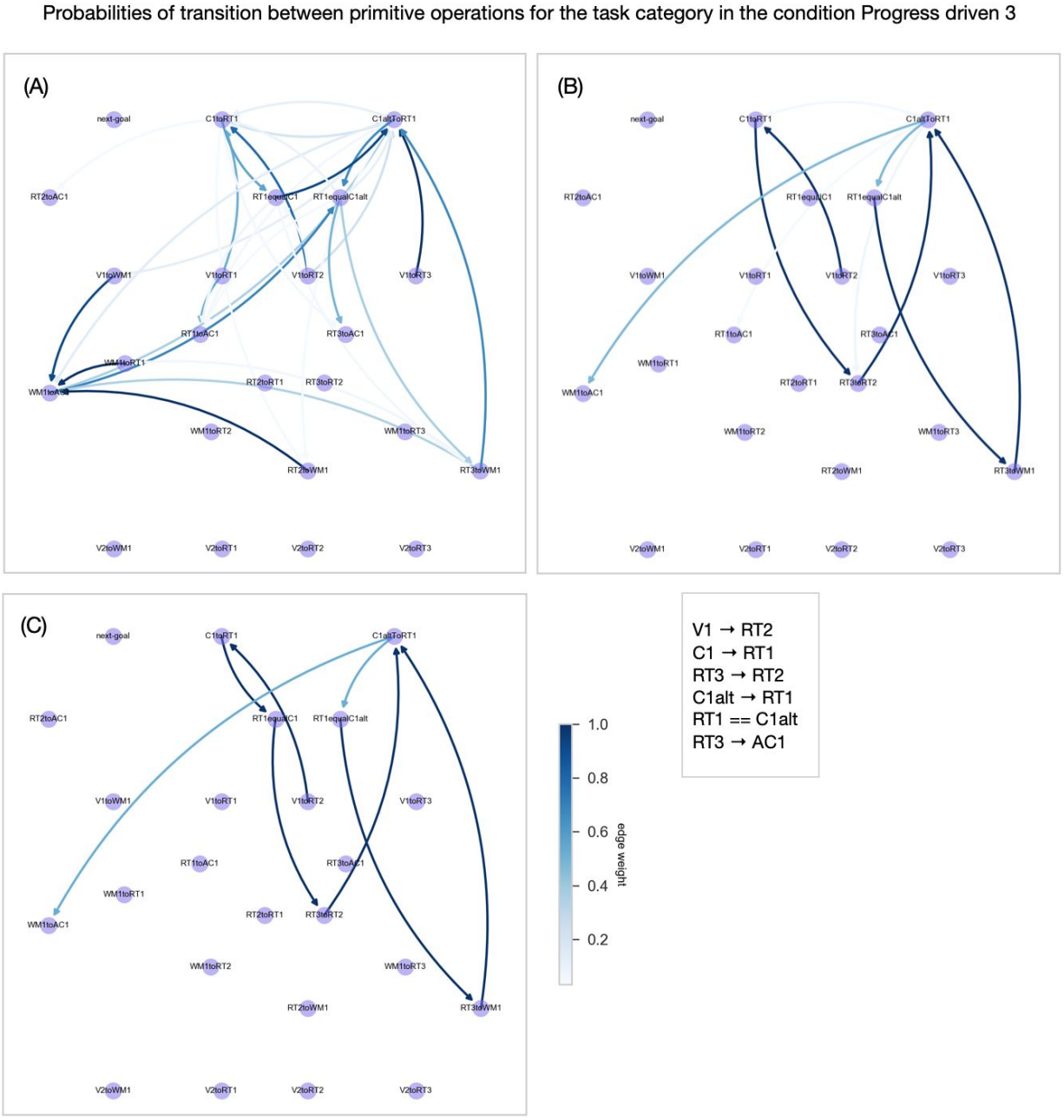

Figure 9. Probabilities of transition between primitive operations for the task category in the condition Progress driven 3. Each node represents a primitive operation. Each arrow represents a transition between two primitive operations. The color indicates the probability that the model carried out that transition. Probabilities were computed over 100 trials. (A), (B), and (C) correspond to three different periods within the same simulation from which the graphs were drawn, respectively from 1700 to 1800 trials (A), from 3900 to 4000 trials (B), and from 4700 to 4800 (C). (A), (B), and (C) are indicated by purple dotted lines in figure 8, B4 as figures 7, 8 and 9 were made from the same simulation.

4. Discussion

Brief summary of the results

To investigate possible goal selection functions and study their influence on action selection learning three types of simulations were compared using seven goals. The frequency of convergence of the model did not differ between the three types of simulations for easy and medium goals. For the hardest goal the progress driven condition showed a clear advantage over the random and the control (no goal selection) conditions. This advantage was caused by the emergent ordering of the goals: easy goals were selected first and then, once the model learnt how to solve them, harder goals were selected. As a result, a larger number of trials were dedicated to the learning of hard goals. However, contrary to my expectations, the benefit of sharing newly created operators was not clearly established. None of the conditions showed a clear positive influence of the mastery of one goal on the speed of convergence of another goal requiring the same sequence of operations to be solved.

The goal dynamics question the taxonomy of needs as defined in the MicroPsi theory

Coherent with the guidance mechanism proposed in Oudeyer et al., 2007, the progress driven goal selection function I implemented improved the learning performance of the PRIMs model thanks to a better distribution of the temporal resources (trials). In addition, the model spontaneously tackled easy goals before switching to harder ones. By doing so, it maximized the number of goals it mastered in a finite amount of time without planning and without knowing when it would need to stop. This result questions the necessity to distinguish between the cognitive needs of competence within a task (1.a) and general competence (1.c) as suggested in Bach, 2015. Indeed, in the present work, when the model selected a goal driven by the past learning progress of the agent, it resulted in satisfying the two above needs.

Neither positive nor negative interaction between the created operators was found

The cognitive architecture literature has limitedly explored the issues of interference between tasks during the learning process as most of the research in this domain has focused on modeling single tasks. On the contrary, in neural networks interference between tasks during the learning process, better known as catastrophic forgetting, is an established problem the community tries to solve. As the present work tested, for the first time, the learning mechanism of the PRIMs architecture on several tasks within the same simulation, it was the occasion to study the transfer of action selection knowledge within a production system.

Interestingly, we did not find negative interference between goals during the learning process although the retrieval mechanism of knowledge is unidimensional (competition between activations). There are two potential reasons for that: the update of base level activations of productions (1) was switched off in all conditions (Appx. A), and spreading activation between productions used successfully to solve a given goal (2) was also switched off in all conditions. Originally, (1) was

added to keep track of the history of a piece of knowledge (declarative, procedural or episodic as ACT-R does not distinguish between types of knowledge). (2) was introduced in PRIMs to facilitate the solving of tasks requiring several productions to succeed. By switching off (1) and (2), the activations of productions were modulated only by the buffers, including the goal buffer after learning. As a result, no interference (or transfer of knowledge) between goals could occur.

If negative interferences were removed, this came at the cost of possible positive interferences which were also removed. One possible solution to create positive interactions between goals could be to lend weight to the environmental cues.

Transfer learning is better seen as a two step process

The absence of positive interactions between goals helped me to clarify the transfer learning process by emphasizing the necessity to split it into: (a) the creation of new productions, and (b) the learning of when to use them. Indeed, knowledge, whatever its form, needs to be twinned with efficient means to access in a useful manner. Therefore, to learn means to encode new information twinned with the means to access it. In the present work, the creation of new operators (a) was efficient. However, these newly created operators had to compete with a large number of already existing productions (24 primitive operations plus all the operators created during learning). Because the model had no means to compare goals and recognize that a sequence of operations it used for one goal could be reused for another, (b) was long and inefficient. Consequently, to improve transfer learning one needs to address two different questions: How to create new productions? (e.g. in PRIMs newly created operators are built from the concatenation of primitive operations) and How to store and retrieve them efficiently?

The question of how to organise memory in order to retrieve items in the most efficient way remains an open problem in computer science[11]. In the reinforcement learning literature, several attempts have been made to organize action selection knowledge in a hierarchical manner[12]. As PRIMs was designed to offer several levels of granularity among productions it offers an unique environment to study this question (Taatgen, 2013). The model offers the possibility to deal with pieces of knowledge at different time scales using independent mechanisms. One can imagine that it will be then possible to navigate seamlessly among these levels of abstraction to solve the retrieval problem.

On the biological plausibility of a continuum from action selection and goal selection processes

From the just before mentioned point of view, to select a goal can be seen as selecting the next action to perform at a larger time scale. Such a continuum between action selection and goal selection is supported by the connectivity of the basal ganglia (R. C. O'Reilly et al., 2018/2020). O'Reilly identified five parallel loops linking basal ganglia with the prefrontal cortex (PFC) by way of specific

[11] For a developed explanation of the problem, see Sloman, 1978.

[12] For a clear presentation of the problem in the reinforcement learning context, see Bruno C. da Silva, 2016.

nuclei of the thalamus. Each of these circuits control action selection/initiation in the corresponding frontal area. The most extensively documented circuit is the motor loop which starts from the supplementary motor area (SMA), joins the putamen (which receives in addition inputs from the premotor cortex (PM), the primary motor area (M1) and the primary somatosensory area (S1)) which in turn reaches the ventral lateral nucleus of the thalamus (VL) that finally connect back to the SMA. Goal management could require the interaction of three loops in which a dynamic gating similar to the one observed in the motor loop could modulate active maintenance in the PFC: the prefrontal loop (dorsolateral prefrontal cortex (DLPFC), also controlled by posterior parietal cortex, and premotor cortex), the orbitofrontal loop (orbitofrontal cortex (OFC), also controlled by inferotemporal cortex (IT), and anterior cingulate cortex (ACC)) and the cingulate loop (anterior cingulate cortex (ACC), also modulated by hippocampus (HIP), entorhinal cortex (EC), and inferior temporal gyrus (IT)) (R. C. O'Reilly & Frank, 2006). In this view, the prefrontal loop selects plans by integrating both the affective information about expected outcomes from the orbitofrontal loop and the affective information about the actions necessary to reach these outcomes (cost of the plan) from the cingulate loop.

Perspectives offered by the present work

Interestingly, at the beginning of this research the question of whether or not the goal selection and the action selection functions should be separated arose. Indeed, in the PRIMs architecture all modules are equivalent, that is the productions that can be used to copy or compare items in the buffers do not differ between modules, including the goal module. An alternative to the present work would have been to manage goals within the goal buffer using productions. As a different activation dynamic for goals would have been necessary for action selection learning to occur without goals being constantly changed, I decided to first try to identify a signal that could drive goal activations. Future research can take advantage of the signal search done here to look at how goal management operates within the workspace. By doing so, the temporal requirements for a goal selection function identified by Hawes (Hawes, 2011) could be met.

Conclusion

The present work was done to explore the relationship between goal selection and action selection learning processes. The goal selection function I adapted from Oudeyer's work improved the learning performance of the architecture thanks to a better distribution of the temporal resources (trials). However, I did not find evidence supporting the view that the creation of new operators for a given goal helped the learning of other goals. This result suggests that a main difficulty in transfer learning lies in the retrieval process of the created operators: how can the model identify that two goals can be solved using the same procedure? I argued that the PRIMs architecture offers an interesting

environment to investigate this question as it was built to offer several levels of granularity among productions contrary to other research paradigms such as reinforcement learning.

5. Acknowledgments

I especially thank Professor Niels Taatgen for his supportive presence during my remote internship. I also want to thank Mark Yang Ji for his help with PRIMs. Finally, I want to thank the Cognitive Modeling group at Groningen University for the enriching meetings and discussions we had.

6. Bibliography

Anderson, J. R. (2007). How Can the Human Mind Occur in the Physical Universe? In *How Can the Human Mind Occur in the Physical Universe?* Oxford University Press. https://www.oxfordscholarship.com/view/10.1093/acprof:oso/9780195324259.001.0001/acprof-9780195324259

Anderson, J. R., & Schooler, L. J. (1991). Reflections of the environment in memory. *Psychological Science*, *2*(6), 396–408. https://doi.org/10.1111/j.1467-9280.1991.tb00174.x

Bach, J. (2009). *Principles of Synthetic Intelligence PSI: An Architecture of Motivated Cognition*. Oxford University Press, USA.

Bach, J. (2015). Modeling Motivation in MicroPsi 2. In J. Bieger, B. Goertzel, & A. Potapov (Eds.), *Artificial General Intelligence* (Vol. 9205, pp. 3–13). Springer International Publishing. http://link.springer.com/10.1007/978-3-319-21365-1_1

Baldassarre, G., Mirolli, M., Gurney, K., Redgrave, P., Schmidhuber, J., Doya, K., Tani, J., & Oudeyer, P.-Y. (2010, October). What are the Key Open Challenges for Understanding Autonomous Cumulative Learning of Skills? *CDS Newsletter*.

Beaudoin, L. P. (1994). *Goal processing in autonomous agents* [University of Birmingham]. https://www.researchgate.net/publication/2334804_Goal_Processing_in_Autonomous_Agents

Boden, M. A. (2006). *Mind as Machine: A History of Cognitive Science*. Clarendon Press.

Boden, M. A. (2007). Creativity in a nutshell. *Think*, *5*(15), 83–96. https://doi.org/10.1017/S147717560000230X

da Silva, Bruno C. (2016). *Machine Learning Advancing Artificial Intelligence*. https://www.youtube.com/watch?v=YTyg8R3PL7s

da Silva, Bruno Castro. (2014). *ICRA 2014 talk*. https://www.youtube.com/watch?v=-kULO9yAhSs

Dean, T., & Boddy, M. (1988). *An Analysis of Time-Dependent Planning*. 49–54.

Goel, A., & Joyner, D. (2019). *Knowledge-Based AI: Cognitive Systems—Georgia Tech Online Course*. https://www.omscs.gatech.edu/cs-7637-knowledge-based-artificial-intelligence-cognitive-systems

Granato, G., & Baldassarre, G. (2019). *Human Flexible Goal-Directed Behavior and the Internal Manipulation of Perceptual Representations: A Computational Model*. 33.

Hawes, N. (2011). A survey of motivation frameworks for intelligent systems. *Artificial Intelligence*, *175*(5), 1020–1036. https://doi.org/10.1016/j.artint.2011.02.002

Ji, M. Y., van Rij, J., & Taatgen, N. A. (2019). *Discoveries of the Algebraic Mind: A PRIMs Model*. 7.

Minsky, M. (2007). *The Emotion Machine: Commonsense Thinking, Artificial Intelligence, and the Future of the Human Mind* (Reprint). Simon & Schuster.

Minsky, M. (2011). *The Society of Mind—MIT OpenCourseWare*. https://www.youtube.com/watch?v=-pb3z2w9gDg&t=5809s

O'Reilly, R. (2018). *Programming in the Brain*.

https://web.stanford.edu/class/cs379c/archive/2018/calendar_invited_talks/lectures/04/12/videos/Randall_OReilly_CS379C_04-12-18.mp4

O'Reilly, R. C., & Frank, M. J. (2006). Making Working Memory Work: A Computational Model of Learning in the Prefrontal Cortex and Basal Ganglia. *Neural Computation*, *18*(2), 283–328. https://doi.org/10.1162/089976606775093909

O'Reilly, R. C., Munaka, Y., Frank, M. J., & Hazy, T. E. (2020). *Computational Cognitive Neuroscience-Wiki Book* (4th ed.). Computational Cognitive Neuroscience. https://github.com/CompCogNeuro/ed4 (Original work published 2018)

Oudeyer, P.-Y., Kaplan, F., & Hafner, V. V. (2007). Intrinsic Motivation Systems for Autonomous Mental Development. *IEEE Transactions on Evolutionary Computation*, *11*(2), 265–286. https://doi.org/10.1109/TEVC.2006.890271

Schmidhuber, J. (2010). Formal Theory of Creativity, Fun, and Intrinsic Motivation (1990–2010). *IEEE Transactions on Autonomous Mental Development*, *2*(3), 230–247. https://doi.org/10.1109/TAMD.2010.2056368

Schooler, L. J., & Anderson, J. R. (1997). The role of process in the rational analysis of memory. *Cognitive Psychology*, *32*(3), 219–250. https://doi.org/10.1006/cogp.1997.0652

Sloman, A. (1978). *The Computer Revolution in Philosophy: Philosophy, Science and Models of Mind*. http://www.cs.bham.ac.uk/research/projects/cogaff/crp/

Sloman, A., Chrisley, R., & Scheutz, M. (2005). The Architectural Basis of Affective States and Processes. In J.-M. Fellous & M. A. Arbib (Eds.), *Who Needs Emotions?* (pp. 203–244). Oxford University Press. https://doi.org/10.1093/acprof:oso/9780195166194.003.0008

Stocco, A., Lebiere, C., & Anderson, J. R. (2010). Conditional Routing of Information to the Cortex: A Model of the Basal Ganglia's Role in Cognitive Coordination. *Psychological Review*, *117*(2), 541–574. https://doi.org/10.1037/a0019077

Sutton, R. S., & Barto, A. G. (2014). *Reinforcement Learning: An Introduction* (Second). The MIT Press.

Taatgen, N. (2017). Cognitive Architectures: Innate or Learned? *AAAI Fall Symposia*.

Taatgen, N. (2019). *PRIMs Tutorial*. https://github.com/ntaatgen/PRIMs-Tutorial/blob/master/PRIMs%20Tutorial.pdf

Taatgen, N. A. (2013). The nature and transfer of cognitive skills. *Psychological Review*, *120*(3), 439–471. https://doi.org/10.1037/a0033138

Appendix A. Details of Declarative Memory in PRIMs

The activation (A_i) of a chunk i is defined by:

$$A_i = B_i + \sum_{k}^{buffers} \sum_{j}^{slots} S_{ji} * W_k + noise \qquad (6)$$

The base-level activation (B_i)* is a measure of the likelihood for the information in chunk i to be useful. It depends on the time passed since the last retrieval of the chunk (for details, see Anderson, 2007, p.103 and Taatgen, 2019, p.21) The measure was designed following the work done by Anderson and Schooler (Anderson & Schooler, 1991; Schooler & Anderson, 1997) and was shown to predict several effects observed in human memory such as the shapes of both the retention function (memory performance as a function of delay) and the learning function (memory performance as a function of practice).

The spreading activation (double summation in (6)) reflects the strength of the association between the chunk i and the elements of the current context. The context is defined as the set of elements in the workspace. It reflects the state of the environment (perceptual buffers) and the internal state of the agent (goal, declarative memory and working memory buffers). The association between chunk i and elements in each of the k buffers that compose the workspace is weighted by the parameter W_k. For example, a way to model an agent with little determination could be to modulate the influence of any visual information on A_i so that it is always superior to the influence of any goal information : $W_{vision} > W_{goal}$. Each non-empty slot in a given buffer is a potential source of spreading activation assuming that it has a non-zero strength of association S_{ji}. The product $S_{ji} * W_k$ is a measure of the likelihood that element j would be part of the context given that i is needed (likelihood(j|i) in (7)).

In bayesian terms, (6) can be read as follow (Anderson, 2007, p.109):

$$log[posterior\ (i|C) = log[prior\ (i)] + \sum_{j \in C} log[likelihood\ (j|i)] \qquad (7)$$

With posterior(i|C) being the odds that memory i will be needed in context C and prior(i) the odds that memory i will be needed based on factors such as recency and frequency.

(*) Remark concerning base level activations in the experimentations: The base level activation of productions is not goal-dependent and can interfere with associative learning between goals and productions. Therefore, its increase over trials as a chuck is used successfully was switched off in all experiments. Independently, the update over trials of base level activation for declarative knowledge remained active.

Appendix B. Supplementary materials

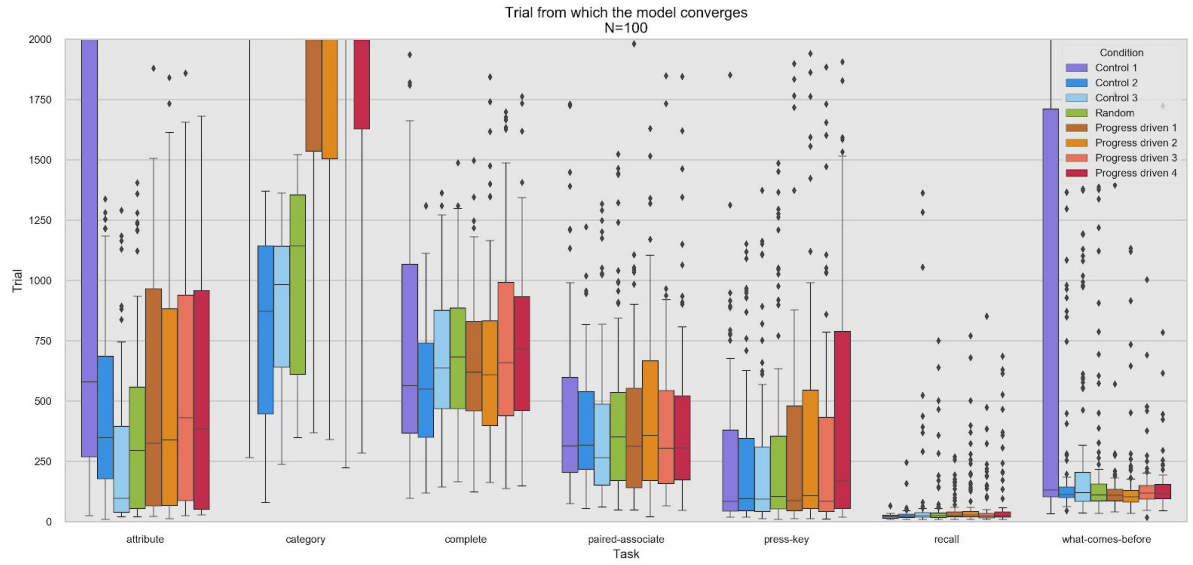

Figure 10. (Zoom in Figure 6) Rank of the first trial from which the model converged per task for the different conditions (control 1, 2, 3: blue shades, random: green, progress driven 1, 2, 3, 4: orange shades). Within each condition, $N = 100$ simulations were done. The numbers of simulations that converged and from which ranks were calculated are: (78, 68, 97, 100, 100, 86, 100) for Control 1, (60, 36, 93, 100, 94, 87, 100) for Control 2, (74, 31, 87, 96, 90, 81, 98) for Control 3, (77, 28, 95, 96, 82, 85, 100) for Random, (60, 73, 90, 88, 81, 73, 89) for Progress driven 1, (81, 71, 100, 100, 93, 90, 100) for Progress driven 2, (81, 66, 100, 100, 95, 92, 100) for Progress driven 3, and (83, 67, 98, 100, 91, 92, 100) for Progress driven 4, per tasks (ranked by alphabetical order).

Figure 11. Example of the dynamics of the accuracy over trials for the task what-comes-before within a simulation for the different conditions: Control 1 (first column), Control 2 (second column). The model made mistakes long after accuracy reached 1.0.