A Formalisation of Memory

Author
Alexander Albrecht

Supervisors
Ronald de Haan
Michiel van Lambalgen

UvA

And now also the axe is laid unto the root of the trees:
therefore every tree which bringeth not forth good fruit is hewn down,
and cast into the fire.

MATTHEW 3:10

If you are the big tree
Then we are the small axe
Sharpened to cut you down
Ready to cut you down.

BOB MARLEY

Abstract

In this thesis a formalization of Kant's account of the architectonic construction of a (or *the*) system of concepts, called the Unity of Reason, is proposed. This is intended to extend the formalization and computational implementation of the Unity of the Understanding called the Apperception Engine put forward in Evans [2020]. Formalizing this construction of a system of concepts is important in two respects: the first being an extension of the Apperception Engine by the addition of memory, allowing it for the first time to remember concepts and reuse these in new situations, giving it test of the cross-applicability of a theory a new criterion for the truth of its theories, as well as granting a large speedup for cases it has already seen. Secondly this allows us to make better sense of the concepts of the Architectonic and the Unity of Reason in Kant, which have for long puzzled commentators and have for that reason received little attention until recent years, even though for Kant they represented the capstone of his project. Though publications like Anderson [2015], Ferrarin [2015] and Ypi [2022] have given proper attention and elucidation to the place and import these topics deserve in Kant's system, they have not yet given a properly technical account of the structure of the Architectonic or the Unity of Reason, which this thesis will seek to provide.

Acknowledgments

First I would like to deeply thank Ronald, without whose consistent support and insight this thesis would surely not have materialized. I am also very thankful to Michiel, for all the ideas I could ever want and for sticking with even if the project was far too ambitious. I would also like to thank Richard and Hein for the helpful discussions and comments. Next I would like to thank Chris for the ever insightful discussions on Kant we have had over the years, and the MLBB for keeping me philosophically stimulated throughout it all. I am grateful to my father for always asking how the thesis is progressing, and equally grateful to my mother for never asking how the thesis is progressing. I am beyond thankful to Scipio, who saved me in a clinch when all seemed lost. And of course I am forever grateful to Fieke, for being there at every step of the way, helping me get through it all.

Contents

System of Abbreviations and Short Titles

In the text we will often refer to different texts of Kant, so per convention and for the sake of readability we use abbreviations for his different works. Note that the *Critique of Pure Reason* is abbreviated both as CPR and as A/B: the A and B indicate the two versions of the CPR, which each have different page-numbers. Referring to a page from CPR will thus look like (A647/B675), to indicate the correct page in both versions. The other citations from Kant will follow the Akademie Ausgabe, and in for example the case of CJ will be written like (CJ 20:201). This citationstyle is also used in the Cambridge Edition of Kant's Collected Works, which is the translation used for each book below.

Abbreviations	**Works by Kant**
CPR or A/B	*Critique of Pure Reason*
CPrR	*Critique of Practical Reason*
CJ	*Critique of the Power of judgement*
MFNS	*Methaphysical Foundations of Natural Science*
OP	*Opus Postumum*
JL	*Jaesche Logic*

Preface

Memory has from the very beginnings of philosophy been a central yet elusive topic, ever since Plato used it in the Meno to solve arguably the first explicit problem of epistemology after Meno asks Socrates the following: " How will you look for it, Socrates, when you do not know at all what it is? How will you aim to search for something you do not know at all? If you should meet with it, how will you know that this is the thing that you did not know?"[1] Without any prior knowledge about the topic she seeks to know an enquirer will know *where* to look, and once it stumbles upon it, will not know that she has *found* it, as she can not know that what she stumbles over is *relevant* to her questions. But if every piece of knowledge requires prior knowledge, then how could we have ever started to know anything at all, if we didn't *know* that we did not know? It is thus impossible to learn anything new. Despite the elliptical qualities of this question Plato takes this problem tremendously seriously, as the attempt to answer this problem is exactly what leads Plato to formulate his theory of the Ideas and the method of *anamnesis* [un-forgetting] to reach these Ideas. For once we dwelled in the sun of the Ideas, but fallen into the darkness of the subterranean caves we have forgotten all we once knew, seeing only shadows now, cast by the light shining through the shards of our once complete system of knowledge. To acquire knowledge thus does not mean learning anything new, as Plato has shown this to be impossible, but to remember, to un-forget, what we once knew. With this Plato short-circuits epistemology right at its birth, as paradoxically to discover something *new* we must remember something *old*, to go forwards in *phenomonal* time, thus learning something we feel we did not know *before*, is to go backwards in *ideal* time, back to the preternatural state indwelling in the Idea. He shows this process by having Socrates guide Meno's unschooled slave into proving a rudimentary geometrical theorem. Socrates lets the slave make a guess to the solution, which turns out to be incorrect, prompting the slave to recognize for the first time that he does *not* know the answer, that the shadows he took for knowledge are not the *real* thing. Then he takes the slave through an elaborate set of examples, which allow the slave to determine the correct solution by himself, once he recognizes the logical steps underlying the sequence of examples. Of course it was Socrates who gave the direction to the Idea through the examples, but Plato argues that this only served to lead the slave to recognize the Idea governing the geometrical problem by himself, even if phenomenally he had never studied any geometry, as the he only had to remember his aquaintance with the Idea. This *anamnesis*, or the unforgetting of the Idea, is thus not spontaneous for Plato, but has to be gradually awakened through a systematic sequence, starting with the recognition of un-knowing, and progressing through relevant examples, leading eventually to the remembrance of the Idea. Now to our modern ears these mythic revelries and initiatory gestures might seem all too quaint, yet throughout this thesis we will see the themes touched on in this preface return again and again: 2500 years later the paradoxical core of epistemology is still inhabited by Memory.[2]

[1] Socrates rephrases it then as follows: "A man cannot search either for what he knows or for what he does not know? He cannot search for what he knows- since he knows it, there is no need to nor for what he does not know, for he does not know what to look for." [Plato, 1949, p. 80]

[2] To see Kant bring the faculty of Reason into Platonic territories as a veiled guide issuing edicts, while also criticizing sophistry donning Platonist garb, read his wonderful short text *On a Recently Prominent Tone of Superiority in Philosophy*[Kant, 1781]

1 Introduction

To properly learn a concept one must be able to remember and then reapply it it in different situations. This is a trivial fact, but performing this task, or allowing a computational program to perform this task, is a necessary component of any learning agent, and a non-trivial exercise. The construction of memory involves a plethora of inter-linked components and methods, starting with a system to efficiently store and reproduce the concepts, which for its proper functioning needs a well thought out structure, criteria for the inclusion of these concepts consisting of a dynamic feedback loop to assess new situations and reassess old situations in light of new knowledge and thus a way to store these old situations in an uninterpreted form. In this thesis we will propose a multi-layered structure for such a memory, heavily inspired by Kant's considerations regarding Reason and the system of knowledge it seeks to produce. To concretize and test Kant's proposal we will implement it in the apperception engine,[3] which is a Kantian machine learner created by Richard Evans.[4] The implementation of Kant's theory of cognition into a computational environment was inspired by Michiel van Lambalgen's formalisation of Kant's logic of cognition,[5] or the transcendental logic, which for the first time showed that Kant's claims regarding the logical foundations of cognition were more than mere embellishments for his theory of mind. [6] Yet this view was merely a consequence of the misapplication of classical logic to Kant's transcendental logic, and once a properly powerful system of logic, namely geometric logic,[7] was used to formalize Kant's transcendental logic by van Lambalgen, the logical ground of his theory of cognition, and in a sense of cognition in general, was laid bare.

This prepared the way for a testing of Kant's thesis in the form of a machine learner based on Kantian ideas, and thus the apperception engine was born. In the upcoming sections we will properly describe its functioning, but for now it is enough to know that it works by being presented sensory data, ideally obtained by observing objects and their behaviour, which it then seeks to explain by postulating concepts, objects and rules governing the behaviour of these objects, thus giving reasons for *why* its 'sensors' were activated in this specific manner, why the sensory data is as it is. In simplified Kantian terms this sensory data is a *sensory manifold* observed by the faculty of Sensation, which passes it on to the faculty of the Understanding, which seeks to *unite* this manifold by connecting the different datapoints by means of judgements consisting of causal rules, thus explaining the sensory data in terms of judgements. For example, when I receive *red* and *round* as information in my visual senses, and *sweet-smelling* in my olfactory senses I make the *judgement*, that the object in front of me is an *apple*, simultaneously connecting these divergent sensory data into one unified interpretation that Kant calls the Unity of the Understanding, which is here simply a single object. This same task, called the *apperception task*, the apperception engine can also perform, even without observing any sensory data previously, which sets it apart from the whole Neural paradigm in Machine Learning, which requires heaps of data before it can make any prediction on a single data-point. Yet in its current implementation the apperception engine is unable to retain any of the concepts it has generated, which means that it needs to redo the (costly) *apperception task* for sensory sequence it receives as input, even for the same sequence. Thus at the moment it is not truly engaged in machine *learning*, as it can not remember the concepts it has produced!

[3] Henceforth simply referred to as apperception engine or AE.

[4] [Evans, 2020]

[5] [Achourioti and Van Lambalgen, 2011, Achourioti et al., 2017]

[6] As claimed by the majority of literature on Kant, for a paradigmatic example take [Strawson, 1959], who castigates Kant for not being Frege.

[7] A fragment of first-order logic consisting of all and only those formulas that can be written as conjunctions of geometric implications, i.e. of formulas of the form $\forall \bar{x}(\theta(\bar{x}) \rightarrow \psi(\bar{x}))$, where θ is a conjunction of atomic formulas, and ψ is constructed from atomic formulas using only $\vee, \wedge, \exists, \bot$. It is called geometric logic because Euclidean geometry can be axiomatised using only geometrical implications, an interesting result given the function of geometry in Plato's Meno discussed in the Preface.

Now the addition of memory to the apperception engine might be considered as only secondary to the already complete and self-contained primary process of cognition it performs. One might argue that because the apperception engine can already unify the manifold of sensation by means of the understanding, that it is already full-fledged cognition, and that this while the addition of memory is a helpful one, it changes nothing with regards to the core process of cognition.[8] Yet with Kant I would point to the third major faculty involved in cognition, namely Reason,[9] which in its practical aspect is involved with setting ends for the agent, but in its theoretical aspect is exactly concerned with the construction of memory, or a system of concepts, by the systematic investigation of the world, thus by guiding the understanding through the infinite multiplicity of sensory manifolds in an ordered way.[10] It is absolutely necessary for the understanding to be fed the right inputs, in the right order, for it to learn any relevant concepts, or even to learn any concepts at all. Simply said, if the apperception engine where incarnated into a robot, and this robot did not have any systematic way of exploring the world, but went in arbitrary directions, receiving arbitrary sensory data, it could simply end up like a roomba in a corner, staring at a white wall for all eternity, continually generating concepts to interpret the various creases in the paint in ever more elaborate ways, ultimately weaving "a tale, told by an idiot, full of sound and fury, signifying nothing."

1.1 Overview of proposed contributions

Before we move to the proper exposition of the Apperception Engine we will give a brief overview of the limitations we observe in the AE and the solutions we propose to the overcome these. The full statement of these will come once we have explained the functioning of the apperception engine, but here we will already present them to give some handles through the text. We have identified three main problems to be attacked in this and future work, listed below:

1. The **Memory Problem** is the main problem to be attacked, as at the moment the apperception engine has no way to store and reproduce the concepts is has learned. We will propose to expand the apperception engine by adding a system of concepts in which it can store the concepts and theories it produces in a structured way, allowing it to then reproduce these in new situations.
2. The **Theory-Choice Problem** consists of the fact that in the current implementation of the apperception engine it can only decide between the different consistent interpretations it produces for a sensory sequence by calculating which of these has the lowest cost, thus needs the least rules to explain the sequence. This is a principle akin to Occam's razor, which works well as a heuristic for the best interpretations, but can not be the final decision on which interpretation is the best, as some theories might for example be overly optimized for the sensory sequence under consideration, while being unable to explain a different but in essence similar sequence. Testing the generality and truth of theories through a method of systematic investigation, thus by testing the theories against other similar situations, would provide a better criterion. For this it is necessary that the apperception engine has a form of memory, as otherwise it can not test the theory it learns in one situation against another situation. This is why it is necessary to first attack the **Memory Problem** if we are to solve the **Theory-Choice Problem.**

[8] In essence this is argued by Evans in [Evans, 2022], where he states that the apperception engine is already able to generate fully-determinate cognitions, by virtue of the fact that the judgements of the understanding are related to the sensory data of sensation.

[9] Besides the imagination of course, but for that I refer to [Soeteman, 2022].

[10] "To every faculty of the mind one can attribute an interest, that is, a principle that contains the condition under which alone its exercise is promoted. Reason, as the faculty of principles, determines the interest of all the powers of the mind but itself determines its own. The interest of its speculative use consists in the cognition of the object up to the highest a priori principles; that of its practical use consists in the determination of the will with respect to the final and complete end." (CPrR 5:119-20)

3. The **Search Problem** is the last major problem under consideration, which we propose because the time-complexity of the *apperception task* is quite high, namely in Σ_2^P, which combined with the fact that the apperception engine now uses a costly diagonal search through an infinite search-space to find the correct amount and structure of concepts used to generate an interpretation of a sensory sequence causes it to take hours to days for to solve complex sensory sequences. Here the functionality of the implementation of memory comes to the fore again, as *checking* an interpretation for a sensory sequence is merely in P, which means that if we can reproduce the theory we learned previously in the initial *apperception task* we can speed this up a lot. Furthermore, even if we are not yet in possession of the right theory, the system of concepts could provide a more optimized search through the possibility-space of the concepts for the *apperception task* if we have any prior information stored in memory for a new sensory sequence, allowing us to cut down on the costly diagonalization. To attack this problem it is again necessary to first implement memory in the apperception engine, which is why the **Memory Problem** will be our primary concern going forward, enabling us to solve the other problems when a form of memory is implemented.

Furthermore we note that on the uptake for the philosophical side of this thesis will be the systematic exposition and testing of Kant's view of Reason and the system of concepts, for while the importance and function of these are extensively discussed in literature like [Anderson, 2015, Ferrarin, 2015, Ypi, 2022, Guyer et al., 2005, Longuenesse, 2020], I have yet to find any clear exposition of the exact structure of the system of concepts he proposes. Attempting to implement this in in a computational environment will force us to lay it out clearly and distinctly, which in itself will thus already be a new addition to the literature.

The uptake for the AI side will first of all be the extension and improvement of the apperception engine by adding memory and showing where it can still make improvements to be both more faithful to Kant and more capable as artificial intelligence. Beyond that, and here I will speculate, we build further on the unique advantages the apperception engine framework proposes over the current neural paradigm, which is that the apperception engine can produce explicit symbolic theories consisting of logical judgements for both symbolic and sub-symbolic sensory sequences. The advantage of these explicit theories over the predictions made using neural archictecure is that these are able to exhaustively explain a sequence, meaning that if it produces the correct theory its predictions remain correct over the entirety of the sequence, as it generates and then follows explicit logical rules that it judges to govern the behaviour of the sequence. This makes it human readable, mathematically verifiable and immune to degeneration, as if the theory is correct it is correct for 100% of the sequence, while all neural architectures, how sophisticated they may be, still ultimately rely on statistical prediction. This means that if it has an accuracy of 99% over a sequence, that in the limit it will start to degenerate to random chance, as it starts to take its own mistakes into the prediction for the next step, for it can not deduce through from any rule that it has made a mistake. Adding to this any neural network needs a high amount of previous data to make a prediction, while the apperception engine needs only the sequence itself, at least in simple cases where no prior knowledge is needed. Adding memory to this framework would provide interesting addition to this discussion, as then the apperception engine will be able to *build* upon its previous knowledge, decreasing its computation time and allowing for the interpretation of progressively more complex sensory situations. This differentiates it from the neural architectures, as these could not learn progressively in such an efficient fashion, for any change to the patterns governing their predictions needs a lot of examples of this new datapoint, where these examples might not be available. The apperception engine extended with memory would need merely one datapoint that is informative enough to learn the rules governing the new situation, which would allow it to quickly make steps in interpreting new situations. This could be especially interesting when we consider sensory situations which are also unknown for us, as when

the apperception engine has a larger system of concepts for reference it could allow us to use this as context to ourselves make sense of the interpretation it has produced in this for us unknown situation. If the apperception engine did not have a larger frame of reference, but could still solve the to us unknown situation it would be nigh impossible to figure out what it has exactly learned, as we would have no references for the new concepts it had produced. Yet when these new concepts are presented in relation to a system of concepts that we ourselves are already acquainted to, we would be able to make sense of it, allowing the apperception engine with memory to make truly new discoveries and share these with us. Over against the implicit knowledge base of neural networks, an explicit knowledge base like the one we will propose allows better exploration of unknown and data-sparse environments, as well as providing a link back from the artificial agent to our own system of knowledge, which would make the discoveries it makes actually useful instead of seeming to be senseless by being outside our frame of reference.

1.2 Basic exposition of the Apperception Engine

Before we can proceed to explaining the structure of the construction of a system of concepts for cognition, what Kant calls the Architectonic or "the art of systems""[11], we will need to describe the functioning of the Apperception Engine[12].
It is easiest to explain using the Seek-Whence problems developed by Douglas Hofstadter[13] as an example. We state an example problem: *extend the following sequence 'abbcccdddd'*, to which a possible answer would be '5 times *e*, then 6 times *f*, etc.'. The rule-set that explains the sequence and our proposed extension could be stated as 'for each letter {x} in the alphabet, print {x} {n} times where {n} correspond to the index of {x} in the alphabet, in the order of lowest to highest index starting with *a* and 1'. With Wittgenstein's rule-paradox[14] in mind we know that there is no *uniquely true* extension to this sequence, as it is always possible to devise a consistent set of rules that extends the sequence in a different manner[15]. The task is thus to construct *any* rule-set that explains the pattern presented, as only then are we able to make a *justified* prediction on its extension, thus one for which we can proffer a rule-set as an argument for the prediction, making it a *justified* prediction instead of a mere guess.[16] It is impossible to make a stronger claim for the rule-set explaining the sequence than it just being one justified prediction among many, as a singular sequence by itself can not determine the *correct* rule-set of itself, the gist of this thesis is that the truth of such a rule-set can only be ascertained throught the systematic comparison to other sequences.[17] The Apperception Engine formalizes this notion of a rule-set, and is capable of finding rule-sets that are able to extend such a sequence. As we discussed above this sets it apart from most modern Machine Learning, which relies on the implicit recognition and extension of patterns, instead of the explicit positing of rule-sets to explain and extend such patterns.
We now present this in formal terms:

[11] "By an architectonic I understand the art of systems." (A832/B860)

[12] Henceforth also called AE

[13] [Hofstadter et al., 1982]

[14] "This was our paradox: no course of action could be determined by a rule, because every course of action can be brought into accord with the rule."[Wittgenstein, 2010, §198], [Kripke, 1982] The explanation for this paradox can be observed perfectly in the case of the apperception engine, as for every given sensory sequence we could devise an infinite amount of rule-sets explaining the sequence in a different yet consistent fashion. An example is given in the text.

[15] Take for example the rule-set that states that after printing 4 *d*'s the sequence starts again at 1 *a*.

[16] A guess can of course be completely *random* or *informed*. The random case is self-explanatory, and the informed case we can model as a guess informed by previous experience, but not justified by any explicit argument. We contend that most neural next-character predictors take *informed* guesses, but can not make any *justified* predictions.

[17] This underdetermination of the rule-set by the sequence will be a central problem to tackle later, as a criterion beyond what the sequence can deliver is needed to determine which rule-set is 'correct' for the sequence.

1.3 Formal exposition of the Apperception Engine

1.3.1 Definition of terms

First Evans defines the *sensory sequence*:[18]

Definition 1. *A **sensory sequence** is a sequence of sets of ground atoms. Given a sequence $S = (S_1, S_2, \ldots)$, every state S_t in S is a set of ground atoms, representing a partial description of the world at a discrete time step t. An atom $p(a) \in S_t$ represents that sensor a has property p at time t. An atom $r(a,b) \in S_f$ represents that sensor a is related via relation r to value b at time t. If $\mathscr{G}$ is the set of all ground atoms, then $S \in \left(2^{\mathfrak{G}}\right)^*$.*

Definition 2. *A theory is a four-tuple $\theta = (\phi, I, R, C)$ where:*

- *$\phi = (T, O, P, V)$ is a **type signature**. T is a set of types for objects, predicates and variables, and O, P, V are sets of objects, predicates and variables which are typed according to T.*
- *The **initial conditions** I of a theory (ϕ, I, R, C) is a set of ground atoms representing a partial description of the facts true at the initial time step.*
- *R is a set of **rules**: There are two types of rule in Datalog. A static rule is a definite clause of the form $\alpha_1 \wedge \ldots \wedge \alpha_n \rightarrow \alpha_0$, where $n \geq 0$ and each α_i is an unground atom[19] consisting of a predicate and a list of variables. Informally, a static rule is interpreted as: if conditions $\alpha_1, \ldots, \alpha_n$ hold at the current time step, then α_0 also holds at that time step. A causal rule is a clause of the form $\alpha_1 \wedge \ldots \wedge \alpha_n \ni \alpha_0$, where $n \geq 0$ and each α_i is an unground atom. A causal rule expresses how facts change over time. Rule $\alpha_1 \wedge \ldots \wedge \alpha_n \ni \alpha_0$ states that if conditions $\alpha_1, \ldots, \alpha_n$ hold at the current time step, then α_0 holds at the next time step.*
- *C is a set of constraints grouped in three kinds. A **unary constraint** is an expression of the form $\forall X, p_1(X) \oplus \ldots \oplus p_n(X)$, where $n > 1$, meaning that for all X, exactly one of $p_1(X), \ldots, p_n(X)$ holds. A **binary constraint** is an expression of the form $\forall X, \forall Y, r_1(X,Y) \oplus \ldots \oplus r_n(X,Y)$ where $n > 1$, meaning that for all objects X and Y, exactly one of the binary relations hold. A **uniqueness constraint** is an expression of the form $\forall X, \exists! Y : t_1, r(X,Y)$, which means that for all objects X of type t_1 there exists a unique object Y such that r(X, Y).*

A theory θ generates an infinite sequence of sets of ground atoms called its **trace** $\tau(\theta) = (A_1, A_2, \ldots)$. Where each set of atoms A_t is the smallest set satisfying first $I \subseteq A_1$, then satisfying all rules in R, with one additional condition in the form of the frame axiom:

Definition 3. *Frame axiom: if α is in A_{t-1} and there is no atom in A_t that is incompossible with α w.r.t constraints C, then $\alpha \in A_t$. Two ground atoms are **incompossible** if there is some constraint c in C and some substitution σ such that the ground constraint $c\sigma$ precludes both atoms being true.*

Explaining is then defined as follows:

Definition 4. *A theory θ with trace $\tau(\theta) = (A_1, \ldots)$ explains a (finite) sensory sequence $S = (S_1 \ldots S_n)$ if $S \subseteq \tau(\theta)$, meaning that $S_t \subseteq A_t$ for $1 \leq t \leq n$.*

Now a theory is said to make sense if the following holds:

Definition 5. *A theory θ **makes sense** of a sensory sequence S if θ explains S, and θ satisfies the following unity conditions, it is then called a **unified interpretation**:*

[18] I will gloss over the explanation of the ASP terms such as ground atoms for now, as it is not really relevant to my discussion, but I will of course add this to the final thesis for completeness' sake.

[19] Thus a variable that can be instantiated (grounded) by any object defined in the type signature if it falls under the predicate defined in the rule.

1. ***Object connectedness****: objects are united by being connected via chains of binary relations*
2. ***Conceptual unity****: predicates feature in at least one constraint*
3. ***Static unity****: atoms are united in a state by jointly satisfying constraints and static rules for every state A_t*
4. ***Temporal unity****: states are united in a sequence by causal rules*

Lastly we define the **cost** of a theory θ as $\text{cost}(\theta)$:

Definition 6. *Given a theory $\theta = (\phi, I, R, C)$, the **cost** of θ is*

$$|I| + \sum \{n+1 \mid \alpha_1 \wedge \ldots \wedge \alpha_n \circ \alpha_0 \in R, \circ \in \{\rightarrow, \ni\}\}$$

Here, $\text{cost}(\theta)$ is just the total number of ground atoms in I plus the total number of unground atoms in the rules of R.

The AE then performs an *apperception task* by taking as input a sensory sequence S, a *template* $\chi = (\phi, N_\rightarrow, N_\ni, N_B)$ containing a type signature ϕ and upper limits for the number of static rules ($N_\rightarrow$), causal rules $N_\ni$ and atoms allowed in the body of R (N_B), and usually some input constraints C. With this it produces a lowest cost theory $\theta = (\phi', I, R, C')$ that *makes sense* of S, where $C \subseteq C'$, $\phi \subseteq \phi'$. Now the template θ and the constraints C are usually provided externally to decrease computation time, but for the templates it is also possible (and of course conceptually preferable) to let the AE find these itself. This is done through a two-tiered process of iteration, where first we iterate through sets T of types (which will act as the concepts which contain both predicates P as intension and objects O as extension) and then given a set of types T we enumerate tuples $(O, P, V, N_\rightarrow, N_\ni, N_B)$ for a particular T. As it is impossible to enumerate all tuples a constant bound (n) for the number of tuples is given and enumerated as well, where a (T, n) pair with will emit n tuples $(O, P, V, N_\rightarrow, N_\ni, N_B)$ using the types in T. As a theory using more types but smaller tuples and thus potentially a lower cost is possible we enumerate through the (T, n) pairs in a diagonal fashion[20] and we don't stop immediately at the first found solution, but continue until a predefined endpoint to the runningtime, keeping all found theories in memory. The diagonal enumeration is clarified in table 1.

	100	200	300	400	...
1	1	2	4	7	...
2	3	5	8	...	
3	6	9	...		
4	10	...			
...	...				

Table 1: Exemplar of the diagonal enumeration of (T, n) pairs. Row t means that there are t types in T, while column n means there are n tuples of the form $(O, P, V, N_\rightarrow, N_\ni, N_B)$ to enumerate. n is incremented by 100. The entries in the table represent the order in which the (T, n) pairs are visited.

This enumeration process is costly, as each *apperception task* is already costly in itself, let alone when it is done for a myriad of possible templates. Which is why in a lot of cases the template is given up front, as Evans notes: "To tame the search space, we provide type signatures for many of the examples described above. Although the Apperception Engine is capable in principle of working without any provided type signature, by enumerating signatures of increasing complexity (see Section 3.7.1), in practice for many of the harder examples, we provide a type signature that has been designed to be sufficiently expressive

[20] Keep in mind that the enumeration goes from smaller tuples to larger tuples, thus a higher n will result in larger tuples.

for the task at hand."[21] Improving on this costly process of enumeration will be one major uptake of the implementation of a conceptual system in the AE attempted in this thesis and we will call this the **Search Problem**. For now we will pass over this problem to further explain the apperception task by way of an example. In this example we will then give the AE a template up front, like usually done in practice, on which it will expand by enumerating further templates from this base template.

1.3.2 Seek Whence Example

As example we use a *Seek Whence* problem as discussed by Evans and provide the initial template χ_1:

$$\phi = \left\{ \begin{array}{l} T = \{sensor, cell, letter\} \\ O = \{s : sensor, c_1 : cell, l_a : letter, l_b : letter, l_c : letter, ...\} \\ P = \{value(sensor, letter), h(sensor, p(cell, letter), q1(cell), r(cell, cell)\} \\ V = \{X : sensor, Y : cell, Y_2 : cell, L : letter, L_2 : letter\} \end{array} \right\}$$

$$N_{\rightarrow} = 1$$
$$N_{\ni} = 2$$
$$N_B = 3$$

As we iterate through the templates $(\chi_1, \chi_2, \chi_3, ...)$, we iterate over tuples $(O, P, V, N_{\rightarrow}, N_{\ni}, N_B)$ and thus increase the number of objects, the number of fluent and permanent predicates, the number of static rules and causal rules, and the number of atoms allowed in the body of a rule. The types are kept static as we have determined up front that no more are necessary.

Evans further notes that domain-specific knowledge is given in the form of the successor relation between letters $l_a, l_b, l_c, ...$ This is provided as the *succ* relation by stating $succ(l_a), succ(l_b), ...)$ as binary predicates. Importantly Evans states the following: "Please note that this knowledge does not have to be given to the system. We verified it is possible for the system to learn the successor relation on a simpler task and then reuse this information in subsequent tasks. We plan to do more continual learning from curricula in future work."[22] This is in fact a central point our implementation will seek to amend, as while it is important that the AE can learn the successor relation, there is still no way for the AE to carry over the concepts it has learned in previous instances to a new situation. Not only would it speed up the apperception task by importing previously learned concepts, these concepts are also necessary for the true understanding of the task at hand, as without the knowledge that the letters $l_a, l_b, l_c, ...$ stand in a successor relation there is no possibility of solving the *Seek Whence* problem, as from a sequence such as **abbcccdddd** it is impossible to predict that **e** is the next character without prior knowledge consisting of $succ(l_d, l_e)$. This is the central point to be solved by the implementation of a conceptual system in the AE and we will call this the **Memory Problem**. For now we pass over this problem and note the addition of the *succ* relation to the template.

Now as sensory sequence to solve we take the "theme song" of the project: **b, a, b, b, b, b, b, c, b, b, d, b, b, e, b, b, ...**. In terms of the AE this is represented as a sensory sequence $S_{1:16}$ where:

[21] [Evans, 2020, p. 218]
[22] [Evans, 2020, p. 83]

$$
\begin{array}{lll}
S_1 = \{value(s,l_b)\} & S_2 = \{value(s,l_a)\} & S_3 = \{value(s,l_b)\} \\
S_4 = \{value(s,l_b)\} & S_5 = \{value(s,l_b)\} & S_6 = \{value(s,l_b)\} \\
S_7 = \{value(s,l_b)\} & S_8 = \{value(s,l_c)\} & S_9 = \{value(s,l_b)\} \\
S_{10} = \{value(s,l_b)\} & S_{11} = \{value(s,l_d)\} & S_{12} = \{value(s,l_b)\} \\
S_{13} = \{value(s,l_b)\} & S_{14} = \{value(s,l_e)\} & S_{15} = \{value(s,l_b)\} \\
S_{16} = \{value(s,l_b)\} &&
\end{array}
$$

When the apperception task is run, the first few templates aren't able to find a solution, which shows the import of the search through the templates. The first template that is expressive enough to admit a solution is one where there are three latent object c_1, c_2, c_3. This interpretation (ϕ, I, R, C) is shown below:

$$
I = \left\{ \begin{array}{llll}
p(c_1,l_b) & p(c_2,l_b) & p(c_3,l_a) & q_1(c_3) \\
q_2(c_1) & q_2(c_2) & r(c_1,c_3) & r(c_3,c_2) \\
r(c_2,c_1) & h(s,c_1) &&
\end{array} \right\}
$$

$$
R = \left\{ \begin{array}{l}
h(X,Y) \wedge p(Y,L) \rightarrow value(X,L) \\
r(Y,Y_2) \wedge h(X,Y) \ni h(X,Y_2) \\
h(X,Y) \wedge q_1(Y) \wedge succ(L,L_2) \wedge p(Y,L) \ni p(Y,L_2)
\end{array} \right\}
$$

$$
C = \left\{ \begin{array}{ll}
\forall X : sensor, & \exists! L : value(X,L) \\
\forall Y : cell, & \exists! L : p(Y,L) \\
\forall Y : cell, & q_1(Y) \oplus q_2(Y) \\
\forall Y : cell, & \exists! Y_2 : r(Y,Y_2)
\end{array} \right\}
$$

In this interpretation there are three latent objects or cells $c_1, c_2, c_3, ...$ over which the sensor moves in the order $c_1, c_3, c_2, c_1, c_3, c_2$. The concept of cell has been subdivided into two different concepts defined by the unary predicates q_1 and q_2, where q_1 can be interpreted as *cell that changes its property "letter" by increasing it* and q_2 can be seen as *cell that keeps it property "letter" static*. The static rule $h(X,Y) \wedge p(Y,L) \rightarrow value(X,L)$ states that the sensor takes the value of the cell it's on, while the causal rule $r(Y,Y_2) \wedge h(X,Y) \ni h(X,Y_2)$ states that the sensor moves from left to right along the objects each timestep. These rules determine the behaviour of the concept of *sensor* in this situation. The next causal rule $h(X,Y) \wedge q_1(Y) \wedge succ(L,L_2) \wedge p(Y,L) \ni p(Y,L_2)$ determines the behaviour of the concept defined by q_1, which is that these cells increase their letter property or p-value the timestep after a sensor has moved on the cell. In this way we can see that the q_2 cells remain the same, while the q_1 cells change property over time in a sequence of **a,b,c,...**.

The constraints usually make sure that the behaviour of the concepts is well regulated, and in that sense define the limits and extensions of the concepts. The first constraint $\forall X : sensor, \exists! L : value(X,L)$ for example determines the fact that a sensor can only have one value at a specific time-step, similarly for the second and fourth constraints. These constraints ,which we call *intensional constraints*, filter out redundancy in the theory, as in principle the sensors *could* have multiple values, but as the sensory sequence only has one value per sensor per timestep, ascribing multiple values would not be explanatory to the sequence. These type of constraints thus serve to explicitly constrict the generation of redundancy, while at the same time giving more determinacy to the concept of sensor, as now the theory explicitly states that a sensor can only have one value.[23] A different type of constraint, which we call *extensional*

[23] As Evans states: "To see the importance of this, observe that if there are no constraints, then there are no exhaustiveness or exclusiveness relations between atoms. An xor constraint e.g. ∀X:t, on(X) ⊕ off (X) both rules out the possibility that an

constraints, is the third constraint $\forall Y : cell, q_1(Y) \oplus q_2(Y)$, as this serves to determine the extension of the cell concept, stating that a cell is either a q_1 cell or a q_2 cell, which both are different species of the genus cell, behaving according to different rules. The initial conditions finally serve to assign predicates to objects and through that generate the trace.

1.3.3 The Problem of the Criterion

One last important thing to note is that the theory θ explaining the sensory sequence S has ultimately been selected by us. This is because for every sensory sequence the AE generates multiple possible theories, presenting as output several of the lowest cost theories, of which one will usually be the theory we desired, which we then decide to take as the correct one. As usually the lowest cost theory is the best theory, why can we not just always choose the lowest cost theory?
First of all there simply many cases where there are multiple theories with the same lowest cost, which means that there is *some* decision to be made, for which the AE is not equipped as it has no criteria (like prior knowledge or systematic concerns) besides the Occam's Razor heuristic to act upon. Secondly there are cases where the input is in some sense not general enough, thus causing the apperception engine to learn rules overly optimized for that input, which can then by being insufficiently general not be reused in a different input, even if the different input should be governed by the same rules as the first input. Taking the shortest theory can thus lead to over optimization. Lastly we must consider the simple fact that the lowest cost theory might *not* be the true theory. This is hard to illustrate with the example I have given above, but imagine the very simple situation represented in table 2 where the AE has two sensors next to each other, where first both sensors are deactivated, at the next time-step the left one is activated, then the following time-step the left one is deactivated and the right one is activated, and at the last time-step both sensors are deactivated. The lowest cost theory to explain this sensory sequence is to postulate an object moving from left to right, activating both sensors in order, but what if *in actuality* the activations of the sensors were produced by two objects moving in lockstep from below the sensors to above the sensors, where the right object was moving a bit behind the left object. This theory is of higher cost as it postulates more objects, but it is the *correct* theory, so how can we ever determine this, as it impossible to decide from the sensory sequence itself? This brings us back to the Wittgenstein's rule-paradox problem we discussed earlier, which boils down to the point that any sequence is underdetermining for the rule-set explaining the sequence. To decide on a rule-set we would thus need to inspect other sequences involving the same object(s), to decide which of the interpretations was the correct one, which compels us to introduce other concerns, primarily of a pragmatic or a systematic nature. Our focus in this thesis lies on the systematic side, which will involve questions of memory, systematicity and systematic investigation; we will call this nexus of problems the

	Sensor 1	Sensor 2
1	off	off
2	on	off
3	off	on
4	off	off

Table 2: Representation of the possibly ambiguous sequence

object is simultaneously on and off (exclusiveness) and also rules out the possibility that an object of type t is neither on nor off (exhaustiveness). It is exhaustiveness which generates states that are determinate, in which it is guaranteed every object of type t is e.g. either on or off . It is exclusiveness which generates incompossibility between atoms, e.g. that on(a) and off (a) are incompossible. Incompossibility, in turn, is needed to constrain the scope of the frame axiom (see Definition 9 above). Without incompossibility, all atoms from the previous time-step would be transferred to the next time-step, and the set of true atoms in the sequence (S1 , S2 , ...) would grow monotonically over time: Si $\subseteq$ S j if i $\leq$ j, which is clearly unacceptable. The purpose of the constraint of conceptual unity is to collect predicates into groups (See [A103-11]. See also: "What the form of disjunctive judgement may do is contribute to the acts of forming categorical and hypothetical judgements the perspective of their possible systematic unity", [Lon98, p.105]) , to provide determinacy in each state, and to ground the incompossibility relation that constrains the way information is propagated between states." [Evans, 2020, pg. 39]

Theory-choice Problem. In the next section we will expand on the three problems we highlighted above and with that conclude our problem statement.

2 Problem statement

2.1 The Memory Problem

The central problem this thesis aims to tackle is the **Memory Problem**, which flows from the simple fact that the current implementation of the Apperception Engine is not able to "remember" the concepts it constructs to make sense of a given sensory sequence and thus not able to reuse these concepts for similar sensory sequences. Because it has no memory it in principle does not "learn" any concepts, making it unable to use previous knowledge to decide on which concepts would be more apt to apply in a new situation or to shorten search time for the explanation of new sensory sequences. Implementing memory in the AE would thus allow us to make headway in both the **Theory-choice Problem** and the **Search Problem**, making it the principal problem to solve.
The problem itself contains multiple layers which will interact in specific ways with the other two problems, where solving the other problems will in turn shed light on the problem of Memory. To make this concrete we will give a first sketch of what it would mean to implement memory in the AE, to be continued in the next chapter.
Our proposed schema for memory is to divide it into four layers:

1. *The Raw Sensory Sequences Layer*: this layer consists of all the previous sensory sequences, 3in a compressed version and possibly cut down to only the "relevant"[24] sequences. We retain these old sensory sequences to allow for the possibility of re-explaining specific sequences in light of changes to our conceptual system, thus when we either add, change or remove a concept that was or could be applied to an already explained sensory sequence. Implementing this layer raises mostly practical and computational concerns, thus our current theoretical discussion and proof-of-concept will gloss over the specifics of its implementation and relegate it to future work.

2. *The Theory Layer*: this layer consists of the explanations given for the previous sensory sequences, ideally composed as a continuous "history" that has lead up to the present sensory sequence *if* they are able to be temporally connected.[25] This layer should also be able to contain multiple hypothetical theories per sensory sequence if there is a moderate to high degree of uncertainty on which theories are most apt (or which concepts are most relevant[26]) for the situation. The idea guiding this postulation of multiple hypotheses is that future experiences will allow us to decide on, or completely reframe, the explanations for previous experiences. As already hinted at above a certain feedback-loop between devising new concepts and deciding on which old hypotheses are most salient will be central to solving the **Theory-choice Problem**. These exact interactions can be discussed in greater detail once we engage with the problem of systematicity in the next layer.

3. *The Conceptual System*: this layer consists of an ordered system of concepts learned and memorized from the theories making sense of earlier sensory sequences, to be applied to new and in some respects similar sensory sequences. This, combined with the *Theory Layer*, is what Kant would call the *Unity of Reason*. Explaining its construction,[27] value and mechanisms, and then implementing

[24] Although defining relevancy will prove to be a recurring hard problem, linked to systematicity.

[25] In principle all experiences should be able to be temporally connected, but the Apperception Engine is now handed disjointed sequences like the Seek-Whence problems, which neither can nor should be connected.

[26] Here relevancy comes up again, and we will engage with it more explicitly at a later moment.

[27] Which is called the *Architectonic of Reason* by Kant.

this into the apperception engine will be the central aim of this thesis. It is composed of a tree with potentially infinite subtrees defined by the intension and extension of concepts and united at the top in the concept of *object in general*, with the tree of sensory objects being united in the concept of *matter*.[28] The structure and composition of the tree will be dictated on the one side by the *time-complexity* of the application of concepts in new situations and the *space-complexity* of the total tree, and on the other side by the use to which the application of concepts is put: the *purpose* of the explanation of a sensory sequence.[29] The construction of the conceptual system is thus on the one hand guided by questions of theory-choice and search-time, but is also determinant of the heuristics used to decrease search-time and most importantly of the decision for which concepts to apply to a specific situation, as we can decide if a rule is apt to use in a situation by how it figures in the system of concepts, and thus how well it allows us to explain other situations: "the systematic unity of the understanding's cognitions. . . is the touchstone of truth of rules" (A647/B675). We will return to this in more detail in the sections corresponding to these problems.

4. *Implicit Memory*: as the tree-like conceptual system can become exceptionally large[30] there needs to be some efficient way to traverse the tree in a heuristic fashion. The simplest and in principle exhaustive algorithm would be to traverse the tree starting from the top node *object in general* and recursively attempting to apply the concepts lying in its extension to construct a working theory. The tree traverse algorithm can be improved in several ways of course, but ideally the AE would not be constantly traversing the tree from top to bottom, but would be able to recognize the location of the concept in the tree by several features through implicit memory instantiated in something like a Neural Network. This would allow the AE also to train it's memory to 'intuitively' recognize certain object as falling under certain concepts, analogous to how human memory operates. Implicit connections should dictate that certain concepts are connected to other concepts in certain situations, with weighted denotations of relevance per concept. Though in principle this is a practically and conceptually important addition, for the current proof-of-concept the tree will remain small enough that a simple traversal algorithm will suffice, and any more sophisticated neural architectures will be are relegated to future work.

These four layers can act as memory of the Apperception Engine and allow it thus to construct a system of the world, or *Weltanschauung* for Kant, which should for the first time allow it to decide *by itself* on which theory is best as well as cut down on search time. We now move to the first descriptions of the **Theory-choice Problem** and the **Search Problem**.

2.2 Theory-choice Problem

The second problem to discuss is the **Theory-choice Problem**, which sprouts from the fact that criteria are needed beyond the sensory sequence itself to decide on a rule-set to explain said sequence, as sensory data is underdetermining for the theories explaining it. The current implementation of the Apperception Engine tries to deal with this impasse in a sensible way, by applying a function akin to Occam's Razor to the generated theories, thus choosing theories with the lowest cost, where cost is defined by the number of initial conditions and rules contained in the theory θ. But, as we touched on above, this is still not enough to act as a proper criterion for theory-choice, presenting us with several problems: (1) There may

[28] As explained in Anderson [2015] and worked out by Kant in the MFNS and OP.

[29] The *ends of science* as Kant says, which allows us to formulate separate trees (or branches of the larger tree) when looking for example at the domain of plants either for their nutritional value, where they are divided into edible or non-edible, or for their medicinal purpose, where they are divided into medicinal or non-medicinal. There are different possible structures of relations for each domain of objects.

[30] It is in principle infinite after all.

be multiple smallest theories, leaving us still with a decision without a criterion, or worse (2) the smallest theory may simply be false, and because the AE is only given an isolated sensory sequence it has no way to *verify* the truth or falsity of its smallest theory, lastly (3) a theory (be it small or large) may simply not be relevant to the purpose of our investigation. Let us expand on these points in order.

2.2.1 Criterion Problem

The first two problems can actually be reduced to a single problem which we call the **Criterion Problem**, for when we solve problem (2) by devising a criterion to decide on the truth of a theory besides the Occam's Razor principle we can solve problem (1) with the same stroke. Our guiding thread in finding a proper criterion for judgement is the following statement made by Kant: "The hypothetical use of reason is therefore directed at the systematic unity of the understanding's cognitions, which, however, is the touchstone of truth for its rules." (A647/B675).[31] Kant is speaking of the 'hypothetical use of reason' here, thus exactly the use of reason we are concerned with, which posits multiple hypotheses for the explanation of a sensory sequence (the synthesis of a manifold) and must be provided with some means to judge the veracity of these hypotheses.[32] Evans provides us with the principle of the lowest-cost theory to decide on the better theory, but problem (1) and (2) show that this is not enough. Kant recognizes this as well and proposes the systematic unity of our cognitions as the ultimate touchstone of the truth of a cognition or theory. This can not be taken in the strict sense, as empirical truth is for Kant always hypothetical, always an approximation that seeks to improve itself further. Yet a notion of systematicity as touchstone of truth is even important here, as it is what gives direction in this ever-progressing approximation.[33]
Now the question is of course how systematic unity is able to give us a criterion for the judgement of hypotheses. Our proposal is that this is done by first of all applying the idea of Occams Razor not simply to the singular theory explaining a sensory sequence, but to the whole *Conceptual System*[34] employed to produce such a hypothesis. This means that we will seek to minimize the amount of concepts needed to explain multiple sensory sequences, thus preferring hypotheses that incorporate concepts which are able to be reused in previous or future situations. Uniting concepts under common genuses will decrease the size of the system, as specific intensionsional information does not have to be saved twice.[35] This is why the implementation of memory is necessary to solve the **Theory-Choice Problem**, as then we can remember and compare multiple sensory sequences, and possibly revise the explanation of previous sequences by discovering that a different concept has better reproducibility. We will make this more concrete once we begin the exposition the implementation of the system of concepts.
Systematic unity is not exhausted by considerations over space and time-complexity in the *System*

[31] "the law of reason to seek unity is necessary, since without it we would have no reason, and without that, no coherent use of the understanding, and, lacking that, no sufficient mark of empirical truth. . . " (A651/B679)

[32] "[If] the universal is assumed only problematically, and it is a mere idea, the particular being certain while the universality of the rule for this consequent is still a problem; then *several particular cases, which are all certain, are tested by the rule, to see if they flow from it, and in the case in which it seems that all the particular cases cited follow from it, then the universality of the rule is inferred,* including all subsequent cases, even those that are not given in themselves. This I will call the "hypothetical" use of reason." (A646-7/B674-5, emphasis mine)

[33] "The hypothetical use of reason, on the basis of ideas as problematic concepts, is not properly constitutive, that is, not such that if one judges in all strictness the truth of the universal rule assumed as a hypothesis thereby follows; for how is one to know all possible consequences, which would prove the universality of the assumed principle if they followed from it? Rather, this use of reason is only regulative, bringing unity into particular cognitions as far as possible and thereby approximating the rule to universality."(A647/B675)

[34] Thus the third layer discussed in the section on the **Memory Problem** above

[35] The intuition is that a conceptual system which orders the concepts *dog* and *cat* under the genus *animal* and only then under *object in general*, instead of immediately under *object in general*, will cut down on space-complexity immensely, as it does not have to save the intensional marks contained under *animal* that *dog* and *cat* share twice. Kant thus prefers deep trees over broad trees.

of Concepts, but includes furthermore the involvement of the *Theory Layer* and the construction of systematic unity through systematic investigation. In the *Theory Layer* of Memory, a continuous history of those sensory sequences is constructed that stand in a relation of succession in time, allowing us to track objects over multiple instances and thus to falsify or verify the behaviour we ascribed to it. This process of verification we can term systematic which amounts to the express isolation or manipulation of an object to decide which of the possible hypotheses are best applied to the previous instances, or if wholly new concepts need to be fashioned to explain the object. Though eminently important to the construction of the Unity of Reason, and to deciding on the correct hypothesis, systematic investigation is outside the scope of this thesis, as it it represents a whole new level of implementation for the Apperception Engine, namely the capacity for autonomous end-oriented activity.[36]

2.2.2 Relevance Problem

The **Relevance Problem** consists of the fact that an explanation, be it short or long, might simply not be relevant to our investigation, while still in principle synthesizing the manifold. Currently this is occluded by the fact that the sensory sequences supplied, and thus the problems posed, to the Apperception Engine are in some sense *one-dimensional*, contrary to the singular infinity contained in the *actual* sensible manifolds presented to a human subject. They are one-dimensional because ultimately *one* explanation is desired in the form of the correct theory for the sensory sequence. In the problem of the one object passing past the sensor from left to right we simply desire a rule-set that postulates some object moving from left to right past our sensors, as this allows the trace generated to cover the sensory sequence. But with a real sensory manifold this is far from enough to properly explain all the intricate details making up our experience of the moving object. The postulation of a moving object can be seen as the *start* of investigation,[37] but only to allow us to then divulge on what kind of object it is; how and why it is moving; what the molecular, atomic and sub-atomic properties are that allow for this specific movement or appearance of the object; what the biological, chemical and physical makeup is of our own sensory organs that allows us to perceive it in such and such a way; why we ourselves are in the situation to perceive it; and ultimately once we go beyond the causal explanation, what matter is and why anything is at all! This is why according to Kant the actual smallest theory explaining a single manifold can be nothing less than a theory of everything: true determinacy comes only from the complete system,[38] which means the end-goal of scientific investigation can be nothing less than the totality of knowledge. Now we are not in possession of this total system, but as Kant notes our theory-construction depends upon the *projection* of this ultimate system, this Unity of Reason, which we take upon ourselves as a project:

[36] And thus autonomous end-oriented activity, or *agency* in short, is also central to systematic unity, which will prove to be a serious problem to the implementation of the Unity of Reason in the Apperception Engine, even for the notion of artificial intelligence in general, which I will argue in an upcoming work.

[37] Which in my view is why Kant goes to the trouble to exhaustively describe the metaphysical properties of and conditions for moving objects in general in the *Metaphysical Foundations*, thus acting as the assumed basis from which proper determination through specification can start.

[38] Kant means this quite literally: to truly determine any single thing we need to compare it with every possible predicate; a demand by reason that the understanding can only fulffill in approximation. Thus Kant states the following: "The proposition Everything existing is thoroughly determined signifies not only that of every given pair of opposed predicates, but also of every pair of possible predicates, one of them must always apply to it; through this proposition predicates are not merely compared logically with one another, but the thing itself is compared transcendentally with the sum total of all possible predicates. What it means is that in order to cognize a thing completely one has to cognize everything possible and determine the thing through it, whether affirmatively or negatively. Thoroughgoing determination is consequently a concept that we can never exhibit in concreto in its totality, and thus it is grounded on an idea which has its seat solely in reason, which prescribes to the understanding the rule of its complete use." (A573/B601)

> "systematic unity (as mere idea) is only a projected unity, which one must regard not as given in itself, but only as a problem; this unity, however, helps to find *a principle for the manifold and particular uses of the understanding*, thereby guiding it even in those cases that are not given and making it coherently connected." (A647/B675, emphasis mine)

The synthesis of a manifold by the understanding, or the Unity of the Understanding, is thus the first step in the projected Unity of Reason, wherein a stable world is produced that acts as a basis from which to further determine this manifold. We state that the objects postulated in this theory are thus not fully determined and can only be taken up in a process of continual approximation towards determinacy.[39] We can thus concur with Arie Soeteman[40] and Beatrice Longuenesse that to synthesize the manifold the Apperception Engine does not need to specify all causal rules determining the behaviour of the postulated objects of experience,[41] but simply needs to presuppose that it is *possible* to determine the objects through causal rules - the objects thus need simply be amenable to causal explanations. Though with Longuenesse I would like to stress the fact that these objects are posited precisely in service of the striving to find the causal rules determining the objects, thus the hypothetical judgement presupposes the investigation towards its confirmation. As Longuenesse notes:

> "The statement that 'everything that happens presupposes something else upon which it follows according to a rule' does not mean that we cognize this rule, but that we are so constituted as to search for it, for its presupposition alone allows us to recognize a permanent to which we attribute changing properties"[42]

Longuenesse makes sense of the earlier quote by Kant by focusing on the "feedback loop" inherent in the synthesis of the manifold, in that only by presupposing unknown rules - that cohere in a larger system - governing the objects we posit *can* we posit these objects as permanent objects, which we can then use as the basis for our investigation into these exact rules. The rules presupposed by the Unity of the Understanding do not stand to each other in arbitrary relations, but cohere inside a larger system, the Unity of Reason, without the presupposition of which there would be no way to actually conduct the investigative striving to find the rule governing the objects. The Unity of Reason thus not only depends upon the cognitions produced by the Unity of the Understanding, but this exact process of cognition presupposes the Unity of Reason as a problem.[43]
As Kant states we must project a systematic unity in our cognitions, as the projection of this Unity of

[39] Contra Evans [2022], where Evans claims that he defends Kant against a Brandomian/Hegelian attack that the Unity of the Understanding does not yet provide determinacy, by taking the supposedly determinate content of the AE as counterexample. I contend that with Hegel, Kant would not consider this determinate yet, but would see this Unity of the Understanding only as the first step towards determinacy, with true determinacy only residing in the Unity of Reason, the complete system. I will expand on this in future work.

[40] [Soeteman, 2022, p. 27]

[41] Which would simply be impossible if the problems of the AE weren't one-dimensional, as for Kant intuition is singular infinity, thus has infinite depth.

[42] [Longuenesse, 2020, pg. 366], furthermore on the same page she continues: "I think Kant's argument is that our perception itself [...] is, in a sense, already a striving to find the rule, and that this striving is also what generates our awareness of the unity of objective time determinations." Note also the end of this last quote, which I take to mean that for Kant the discrepancy between the real and the Ideal is exactly what constitutes the awareness of objective time for the subject: the striving to obtain the Ideal, the Unity of Reason, is what allows us to order our time-perceptions in terms of objective time as we discover a specific causal rule that determines whether an object precedes or follows another object. Our striving allows us to see *that* there is objective time and not a merely subjective stream of perceptions, otherwise we would not strive, but only in the Ideal system that we strive to can objective time be determinately known in its contents.

[43] "The reflective power of judgement thus works with given appearances to bring them under empirical concepts of determinate natural things not schematically, but technically; not merely mechanically, like a tool controlled by the understanding and the senses, but artistically, according to the universal but nonetheless indeterminate principle of a purposive and systematic ordering of nature. Our power of judgement is favoured, as it were, by the conformity of the particular laws of nature (about

Reason is what allows for *systematic investigation*, which is what allows us to build a system in the first place: the Idea is a self-generating totality that determines itself through the positing of its own possibility, which is why Alfredo Ferrarin stresses the fact that for Kant Reason is ultimately directed towards itself, towards the realization of its own ends.[44] This sounds overly abstract, but we can make this point very concrete: as we pointed out above there are a myriad of possible avenues of explanation for any single sensible manifold - biological, chemical, physical, historical, religious, etc. - and without a notion of *systematic* investigation an agent would get lost in all the possible lines of investigation.[45] Without *a principle* to guide it in its investigation the subject would descend into an endless rhapsodic enquiry for all the different rules governing its objects, and thus we need to be able to determine which lines of investigation are relevant for our purposes.[46] Still, before we can break our heads concerning the feasibility and implementation of purposes and values in a computational environment, we should waylay the question of the relevance of concepts to focus on the construction of *any* conceptual tree at all. The relevance of concepts is where Kant draws the dividing line between the *scholastic* system of concepts, concerned merely with systematic unity for the sake of unity - thus ultimately for arbitrary purposes, and the *architectonic* system of concepts, which is constructed with the theoretical and practical ends of reason in mind, thus concerns itself with the purpose and relevance of its concepts.[47]

2.3 Search problem

Finally we discuss the **Search Problem**, which consists of two parts, the first being the current diagonal search over the templates, and the second being the time-complexity of the *apperception task* itself given a template. Finding a template and running the *apperception* task is a very costly affair for the base apperception engine, and using the system of concepts we will seek to improve on this.

2.3.1 Template search

First we consider the diagonal search for a template as discussed in section 1.3.1. This diagonal search allows the apperception to in principle construct every possible template, thus allowing it to find a template for every possible input, but in practice this search space can get so exceptionally large, that combined with the time-complexity of the *apperception task* this becomes unfeasible for complex inputs. This is why, as Evans notes, in most cases we provide a template upfront, either fully complete or as a base for iteration:

> "To tame the search space, we provide type signatures for many of the examples described above. Although the Apperception Engine is capable in principle of working without any

which the understanding is silent) to the possibility of experience as a system, *which is a presupposition without which we have no hope of finding our way in the labyrinth of the multiplicity of possible particular laws.*"(CJ 20:213-214, emphasis mine)

[44] "[T]hat reason is entirely occupied with its ends means that it is directed only to itself. Its unity is a self-enclosed organized unity of interests and ends. Reason cares about and can apply itself only to laws and principles. It alone can establish them. In its autarchy reason will not let any restriction stand in its way. It shows its 'abhorrence' of limits and of all 'principles that are not its own work' (Ak 18: 272–75)." [Ferrarin, 2015, p. 31-32] This I bring up against Evans' spontaneity sandwich in his recent Evans [2022], as thought for Kant is directed at its self-posited goal of the realization of the Good which is the unity of freedom and nature in the kingdom of ends, not just instrumentally at the synthesis of intuition. I will also expand upon this in future work.

[45] This is akin to the Frame Problem.

[46] Compare footnote 43

[47] As explained in [Ypi, 2022] But as Kant notes, the construction of the *architectonic* system of concepts presupposes the ability to construct a *scholastic* system of concepts[48], which we identify with the *Conceptual System* layer discussed in section 2.1: this constitutes the main theoretical and technical goal of this thesis. In the next section we will then outline a possible construction of the *scholastic* system of concepts *in abstracto*, which will subsequently be implemented in the Apperception Engine *in concreto*.

provided type signature, by enumerating signatures of increasing complexity (see Section 3.7.1)[All references in this quotation refer to sections in [Evans, 2020]], in practice for many of the harder examples, we provide a type signature that has been designed to be sufficiently expressive for the task at hand. The dominant reason for our system's scaling difficulties is that it uses a maximising SAT solver to search through the space of logic programs. Finding an optimal solution to an ASP program with weak constraints is in Σ_2^P ; but this complexity is a function of the number of ground atoms, and the number of ground atoms of our ASP program is exponential in the length of the Datalog$^{\ni}$ programs we are synthesising (see Section 3.7.4)." [Evans, 2020, p. 218]

Each step of the diagonalization thus initiates an *apperception task* in Σ_2^P, where this goes over the number of ground atoms, which are also exponential to the length of the template.[49] Being able to decrease the complexity of the *apperception task* be the most beneficial to the time-complexity, but once the problems become more complex it will also be important to reduce on the time taken in diagonalization, given that each longer template found in the iteration takes exponentially more time to compute. Recall table 1 where the diagonalization process was shown, and note that if our input requires 4 types in the template we have to perform the *apperception task* 1600 times before we get to an actually useful template, while when it requires 5 types we would have to perform 3400 *apperception tasks*, which each become exponentially costlier the more types are included. This is why we normally provide a more complex template up front, to skip the plethora of trivial *apperception tasks*. A way to cut down on this diagonalization without providing the template ourselves would thus be hugely beneficial, which is where the proposed memory comes in, to provide better adapted templates up front for the apperception engine, on which it could then diagonalize further. The amount of templates it produces is dependent upon the amount of hypotheses we allow it to produce, which is depending on the either a constant number, or dependent upon size of the memory tree. This is because for each concept in the input there could be multiple nodes in the tree that includes that concept, depending on how we set up the tree. We will return to the specifics later, but note here that the upper-bound for the amount of templates produced by the memory is $|\Theta_\tau| \leq \|C\| \cdot \|\tau\|$, where $\|\tau\|$ is equal to the amount of nodes in the tree. This means that every concept could found in every node in the tree, which should be nigh impossible, as it indicates a very malformed tree where every node contains the same concept. What is more important than the current upper-bound is that the memory for the first time allows us to optimize the iteration algorithm, as now we can use previously learned information to construct a template, while without the memory we would be permanently stuck with the same diagonalization algorithm. Like with the other problems, the addition of memory does not simply solve the problem, but does for the first time allow for the articulation of possible solutions, thus creating a dynamic and evolving system. One thing the memory does solve quite optimally though is the speed-up for the *apperception task* when the input or a similar one has already been seen, which is what we discuss next.

[49]The calculation from the complexity of the template of the amount of ground atoms in the Datalog$^{\ni}$ program for the three most expensive clauses is approximately

$$5 \cdot \left|\Sigma_\phi\right| \cdot (N_\rightarrow + N_\ni) \cdot \left|U_\phi\right| \cdot t$$

where $\left|\Sigma_\phi\right|$ is the number of ground atoms for the variables V for objects O, thus $\left|\Sigma_\phi\right| \leq |O|^{|V|}$, $\left|U_\phi\right|$ is the number of ground atoms for the variables V for predicates P, thus $\left|U_\phi\right| \leq |P| \cdot |V|^2$, $(N_\rightarrow + N_\ni)$ are the maximal number of arrow and cause rules, t is the amount of time-steps in the input and the constant 5 is added for the different ground clauses these ground atoms figure in.

2.3.2 Theory reproduction

As noted above the *apperception task* is in Σ_2^P,[50], which makes it a very costly endeavour, combined with the exponential growth of the ground atoms depending on template and input size. Yet an incredible speed-up simply lies in the storing of previously found theories,[51] as checking such a theory for a given template/input combination is in P. The ability to reproduce a theory, or at least a partial theory, for a given input using previously learned rules is thus obviously the largest time-save we can effect for the apperception engine. This does not presuppose that the new situation is identical to the previous situation, as we expect the apperception engine to learn properly general concepts and rules, reusable in various other situations. Though this generality can not be assumed, as an improperly general input can allow the apperception engine to learn overly specified rules for the input, which we will see in the results of section 5.2.2. This just to state that especially for this part of the search problem *systematic investigation* is imperative for the proper functioning of the system of concepts. Now without further ado we will embark upon the theoretical exposition of this system of concepts, to explain how we can reproduce previously learned rules in new situations, leading to an immense speedup.

3 Theoretical Exposition of the System of Concepts

3.1 Introduction

As outlined above the aim of this thesis will be to implement the *Conceptual System* layer of the proposed memory for the Apperception Engine. This will allow the AE to properly retain the concepts it has generated to explain a single sensory sequence and to subsequently use these concepts to explain new sensory sequences, be they different or similar. Besides enabling a massive speedup for similar situations and providing the ability to build on previously learned information,[52] the formation of a conceptual system will also allow us to venture the first steps into attacking the **Theory-Choice Problem** by providing the touchstone of systematic unity as an extension of the current Occam's Razor heuristic. Following Kant, our model for the system of concepts will be a simple, yet ultimately infinitely expandable, genus-species tree, coupled with a concept-application algorithm that iteratively constructs hypotheses to explain the sensory sequence at hand. These multiple hypotheses are needed, as even with a simple tree there will be multiple viable and mutually exclusive theories that can explain the sensory sequence, and here again the choice for a specific hypothesis will have to be guided by systematic and time- and space-complexity related concerns. The **Theory-Choice Problem** thus remains in some sense, yet the import of this system of concepts is that now the AE should be able to make a decision on grounds beyond Occams's Razor, namely on the fitness of the system of concepts that will again result from the application of concepts to the specific sensory sequence. The two stages of *system construction*, where we integrate the concepts generated for a specific sensory sequence into the conceptual tree, and *concept application*, where we apply the concepts inhabiting the conceptual tree to a new sensory sequence, are thus intimately related in their functioning, even though we will treat them separately in the next sections. A third stage in this process will also be summarily discussed, which is the *maintenance* stage, responsible for pruning, compressing and reordering the tree in the absence of any sensory sequence. Though important, this stage is not crucial to the proof-of-concept we present here in this thesis, as it's main upshot is the optimization of the conceptual system, while unnecessary for construction of the system per se.
To illustrate the two central stages of the construction and application of the system of concepts we will

[50] [Evans, 2020, p.64], [Brewka et al., 2003]

[51] If P is not NP at least.

[52] Like in the simple example of learning the successor relation between letters to be used in making sense of Seek Whence problems.

first present a suitably complex example, *Symbolic Sokoban*, as working through this example will allow us to make the different steps in the process clear and distinct. After that we will show how this system of concepts is practically implemented in the Apperception Engine.

3.2 Symbolic Sokoban

As the leading example and 'playground' to exemplify the mechanisms of a system of concepts for the AE we introduce the *Symbolic Sokoban* problem; a simplified version of the Sokoban problem used by Evans to test the application of the AE to raw sensory data. As we are not focusing (yet) on the explanation of *raw* sensory data we will lift the Sokoban problem to the symbolic domain, primarily for simplicity's sake and to decrease computation time, as we use the problem not for its raw sensory data environment but for its aptness for the intuitive understanding of concept-application and the potential to add new objects allowing us to test several features of the conceptual system. First we will present the *Symbolic Sokoban* problem, coupled with the results of the AE given for the analogous Sokoban problem from Evans' study. Later we will then continually rework the problem to show the function of the system of concepts for the Apperception Engine.

Sokoban is a puzzle game where the player controls a figure that can move around a two-dimensional grid world to push blocks onto designated target squares. We generate a sequence of human play such as seen in figure 1 and set the system to make sense of this sequence. As the AE can not yet explain motivations/dispositions,[53] we treat the human actions as *exogenous*, meaning that while they can be used to explain the sensory sequence, they themselves do not need to be explained. In [Evans, 2020] the Sokoban problem is constructed as a next-step problem, where the AE has to predict the next step given the sensory sequences and the player action for the last step. As the correct rule-set should be able to explain every Sokoban step for every game with every board size it is a better metric to inspect the validity of the rule-set ourselves than to evaluate the correctness of the next-step prediction, but we include it here for exposition's sake.

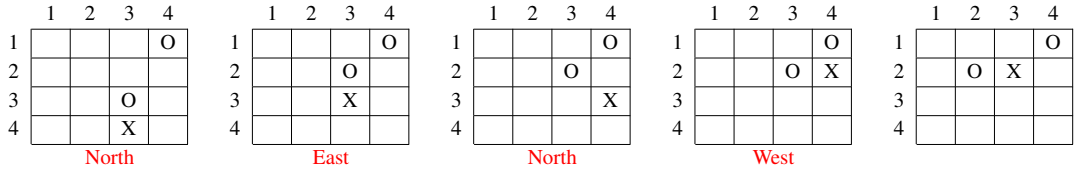
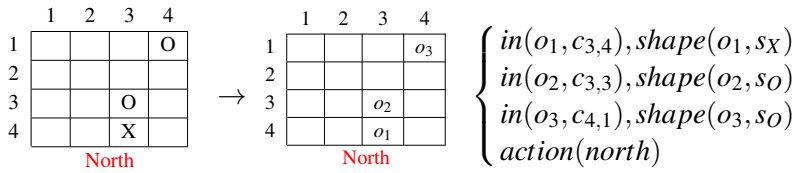


Figure 1: An example sequence of a game of Sokoban with the player actions printed below the grid for each timestep. The designated end-square was here (1,2), but this is not given to the AE as we don't require it to explain the player actions, only the behaviour of the different objects depending on the player actions.

For the *apperception task* the input type signature ϕ and initial constraints C are:

[53] Or will never be able to, a question we will return to in a later discussion.



$$\begin{cases} in(o_1, c_{3,4}), shape(o_1, s_X) \\ in(o_2, c_{3,3}), shape(o_2, s_O) \\ in(o_3, c_{4,1}), shape(o_3, s_O) \\ action(north) \end{cases}$$

Figure 2: An example of the symbolic state translated from the Sokoban board state, which taken as input for the apperception task.

$$\phi = \begin{cases} \begin{cases} T = \{cell, object, symbol, d\} \\ O = \begin{cases} c_{x,y} : cell \mid (x,y) \in \{1,2,3,4\} \times \{1,2,3,4\} \\ o_1 : object \ldots o_m : object \\ north : d, east : d, south : d, west : d \\ symbol : s_X, symbol : s_O \end{cases} \\ P = \begin{cases} shape(object, symbol) \\ in(object, cell) \\ action(d) \\ right(cell, cell) \\ below(cell, cell) \end{cases} \\ V = C : cell, A : d, S : shape, X : object \end{cases} \\ C = \begin{cases} \forall X, \exists! C : cell, in(X, C) \\ \forall X, \exists! S : symbol, shape(X, S) \\ \exists! A : d, action(A) \end{cases} \end{cases}$$

In the frame ϕ given above one can see that we have modified the the implementation given in [Evans, 2020] by removing the types $v_1, \ldots, v_n$ and replacing them with the *object* and *symbol* types, coupled with the binary *shape* predicate. This is because we want to give the objects and their symbols as input, in distinction to the original implementation, which featured raw sensory input where a Neural Net identified the shape of the different objects. As a consequence of this we have also replaced the in_i predicates with the *in* predicate. In the current example this changes little for the apperception task and the resulting theory, but because we have not yet assigned a specific type to an object this will allow us in future examples to implement ambiguity in the relation between the *accidental* (shape) and *essential marks* (*moving* player or *movable* block) of the objects; where for example multiple types could have the same shape or one type could have multiple shapes. In the current implementation we see in the rules section of the theory θ that the feature whereby the apperception engine distinguishes the player and the block is their shape, respectively X and O, this we will call the *sufficient mark*[54] of a concept. A set of *sufficient marks* is supposed to be *sufficient* to distinguish a concept from all other concepts in the system, and in this case the shapes fulfill this role splendidly. Allowing ambiguity in the shape of the objects will force a change in *sufficient marks* for the concepts in question, which will allow us to demonstrate the functioning of the conceptual system. We will return to this later with an example. Now let's run through the type signature ϕ for the sake of understanding.

The *object* type keeps track of the individual objects without deciding on their type, which allows us to present a relatively flexible input sequence when coupled with the *symbol* type and *shape* predicate. The AE can of course iterate over the amount of types and objects until it has found a working theory, but because the minimal background info necessary to give the input is already sufficient for the production of a theory it will not need to produce more types.
Furthermore the cells are explicitly defined using their coordinates as $c_{x,y}$, the binary *in* predicate is defined and the player actions are covered by the *action* predicate which can take as value one of the four cardinal directions. Note also that for the relative spatial positioning between the objects only *right* and *below* are given, as these can also represent *above* and *left* by switching the places of the variables.

[54] "A mark is *sufficient* insofar as it suffices always to distinguish the thing from all others; otherwise it is insufficient, as the mark of barking is, for example, for dogs. The sufficiency of marks, as well as their importance, is to be determined only in a relative sense, in relation to ends that are intended through a cognition. *Necessary marks*, finally, are those that must always be there to be found in the thing represented. Marks of this sort are *also called essential* and are opposed to extra-essential and *accidental marks*, which can be separated from the concept of the thing." (JL 60, emphasis mine)

Several constraints are given to make sure an object can not have multiple shapes or be in multiple places at once, and so there is only one player action per time step.
Lastly as background knowledge the spatial arrangement of the grid cells is provided: $right(c_{1,1}, c_{2,1}), below(c_{1,1}, c_{1,2})$, etc.[55]

Now we can move on to the best theory learned by the Apperception Engine for the above Sokoban example as shown below:[56]

$$I = \left\{ \begin{array}{l} in(o_1, c_{3,4}), shape(o_1, s_X) \\ in(o_2, c_{3,3}), shape(o_2, s_O) \\ in(o_3, c_{4,1}), shape(o_3, s_O) \end{array} \right\}$$

$$R_1 = \left\{ \begin{array}{l} \text{action(north)} \wedge shape(X, s_X) \wedge in(X, C_1) \wedge \text{below}(C_2, C_1) \ni in(X, C_2) \\ \text{action(east)} \wedge shape(X, s_X) \wedge in(X, C_1) \wedge \text{right}(C_1, C_2) \ni in(X, C_2) \\ \text{action(south)} \wedge shape(X, s_X) \wedge in(X, C_1) \wedge \text{below}(C_1, C_2) \ni in(X, C_2) \\ \text{action(west)} \wedge shape(X, s_X) \wedge in(X, C_1) \wedge \text{right}(C_2, C_1) \ni in(X, C_2) \end{array} \right\}$$

$$R_2 = \left\{ \begin{array}{l} shape(X, s_X) \wedge shape(Y, s_O) \wedge in(X, C_1) \wedge in(Y, C_2) \wedge \text{below}(C_2, C_1) \wedge \text{action(north)} \rightarrow p_1(Y) \\ shape(X, s_X) \wedge shape(Y, s_O) \wedge in(X, C_1) \wedge in(Y, C_2) \wedge \text{right}(C_1, C_2) \wedge \text{action(east)} \rightarrow p_2(Y) \\ shape(X, s_X) \wedge shape(Y, s_O) \wedge in(X, C_1) \wedge in(Y, C_2) \wedge \text{below}(C_1, C_2) \wedge \text{action(south)} \rightarrow p_3(Y) \\ shape(X, s_X) \wedge shape(Y, s_O) \wedge in(X, C_1) \wedge in(Y, C_2) \wedge \text{right}(C_2, C_1) \wedge \text{action(west)} \rightarrow p_4(Y) \end{array} \right\}$$

$$R_3 = \left\{ \begin{array}{l} p_1(Y) \wedge in(Y, C_1) \wedge \text{below}(C_2, C_1) \ni in(Y, C_2) \\ p_2(Y) \wedge in(Y, C_1) \wedge \text{right}(C_1, C_2) \ni in(Y, C_2) \\ p_3(Y) \wedge in(Y, C_1) \wedge \text{below}(C_1, C_2) \ni in(Y, C_2) \\ p_4(Y) \wedge in(Y, C_1) \wedge \text{right}(C_2, C_1) \ni in(Y, C_2) \end{array} \right\}$$

$$C = \left\{ \begin{array}{l} \forall X : t_1, \exists! C : \text{cell}, in(X, C) \end{array} \right\}$$

The initial conditions state in which cells the different objects are positioned, coupled with their respective shape.
The first cluster of rules, thus those of the form $\text{action(east)} \wedge shape(X, s_X) \wedge in(X, C_1) \wedge \text{right}(C_1, C_2) \ni in(X, C_2)$ govern the movement of the player X, identified as *player* by X featuring in $shape(X, s_X)$, upon a specific player action: if for example we see the player action *north* and the player X is in a cell C_1 below a cell C_2 then the next step the player X will be in C_2.
The second cluster of rules, thus those of the form $shape(X, s_X) \wedge shape(Y, s_O) \wedge in(X, C_1) \wedge in(Y, C_2) \wedge \text{below}(C_1, C_2) \wedge \text{action(south)} \rightarrow p_3(Y)$ define a new latent predicate p using a static rule to indicate the direction a block, identified as *block* by Y featuring in $shape(Y, s_O)$, will move to if pushed by the player.
Lastly the third cluster of rules, thus those of the form $p_1(Y) \wedge in_2(Y, C_1) \wedge \text{below}(C_2, C_1) \ni in_2(Y, C_2)$, move the 'pushed' block to the intended direction defined by predicate p if there is a cell in that direction.

Now as Evans also notes this theory does not include a check to see if there is a second block in the space that the player is trying to push a first block, as in the Sokoban rules this is a forbidden action, but the AE can not have created this rule this as it has not *seen* this interaction yet. Taking this as a lead our first question will be to see if it is possible to robustly store the rules and concepts the AE has learned for

[55] It is possible to let the AE construct this spatial information as is shown in chapter 5.5.3 of Evans [2020] and expanded upon in Soeteman [2022], but this is unnecesary for our purposes.

[56] For the sake of legibility when building the concept tree later on I have divided the rules section R in three clusters of rules R_1, R_2, R_3. This is purely for visual clarity and is not reflected in the actual structure below.

Sokoban, and add such a check later when it encounters it in a new sensory sequence. This is essentially the **Memory Problem**, let us then consider the formal construction of a conceptual system to tackle this problem.

3.3 Concept formation

To tackle the **Memory Problem** and thus gain the capacity to add concepts to the theory of the Apperception Engine we will have to construct a system of concepts, *to which* these concepts can be added in the first place. Specifically we will discuss the construction of the *Conceptual System* discussed in chapter 2.1. As in our view a concept can only truly be called a concept when it is part of a system of concepts, we will refer to the whole process as the process of *Concept Formation*. This process consists of two main interlinked components called *System Construction* and *Concept Application*, which we will present in this order. Keep in mind though that in the end it is a single process, a feedback-loop as you will, thus applying a concept will be shown to be an integral part of constructing a system and vice versa. Let's then begin with constructing a system out of the partly-determinate concepts constructed by the AE for the Sokoban problem.

3.3.1 Philosophical considerations of System Construction

In this chapter we will seek to specify philosophically Kant's notion of the system of concepts, to prepare the field for the more concrete construction. Let us thus begin with a quote from Kant describing the form of the system of concepts as presupposed for the functioning of the understanding:

> "Reason thus prepares the field for the understanding:[57] 1. by a principle of sameness of kind in the manifold under higher genera, 2. by a principle of the variety of what is same in kind under lower species; and in order to complete the systematic unity it adds 3. still another law of the affinity of all concepts, which offers a continuous transition from every species to every other through a graduated increase of varieties. We can call these *the principles of the homogeneity, specification and continuity of forms.* The last arises by uniting the first two, according as one has completed the systematic connection in the idea by ascending to higher genera, as well as descending to lower species; for then all manifolds are akin one to another, because they are all collectively descended/ through every degree of extended determination, from a single highest genus." (A657/B685) [Emphasis mine]

As we can see Kant has a deceptively simple notion of the system of concepts, namely one of a genus-species tree with *object in general* as its root, the intension to every subconcept, subdividing into an infinitely leaved array of specifications. There is no lowest species (*species infima*), thus every subconcept can in principle be specified through *subdivision* into further subconcepts ad infinitum.[58] More concretely, we can describe Kant's three *principles of systematicity* in terms outlined previously. The first *principle of homogeneity*, which seeks to unity seemingly dissimilar concepts under a common genus ultimately directs us to view all concepts under the abstract notion of the *object in general*, can be equated to the Occam's Razor principle. The execution of this principle seeks to constrain the amount of concepts

[57] Note here the unequivocal statement that the systematic concerns of reason precede the understanding by *'preparing the field'* for it. Reason is thus directive with respect to the understanding.

[58] "every genus requires different species, and these subspecies, and since none of the latter once again is ever without a sphere, (a domain as a conceptus communis),' reason demands in its entire extension that no species be regarded as in itself the lowest; for since each species is always a concept that contains within itself only what is common to different things, this concept cannot be thoroughly determined, hence it cannot be related to an individual, consequently, it must at every time contain other con- cepts, i.e., subspecies, under itself. This law of specification could be ex- pressed thus: *entium varietates non temere esse minuendas.* ["The varieties of entities are not to be diminished rashly."]" (A655/B683)

populating the system to the furthest possible extent, ultimately with the aim to capture all concepts as modifications of the single root concept *object in general*. Simply said, when encountering two slightly different rocks in the sensory manifold, this principle urges us to unite them under the genus of *rock*, instead of devising two separate concepts for the two separate rocks. Without this principle of homogeneity the sensory manifold would only be interpreted as so many distinct and unrelated entities, as ultimately no two objects are ever fully the same. But as no two objects are ever the same we need to make a distinction between the *essential* and *accidental marks* inherent in these objects, thus two be able to unite these two objects under the same concepts through their *essential marks* like hardness and chemical composition, while distinguishing them through *accidental marks* like size, shape and location. The *principle of homogeneity* drives us to connect the objects in the manifold, and the concepts gleaned from these objects, under similar genuses, thus acting as a constraint on the space-complexity of the concept tree.

Opposed to this *principle of homogeneity* is the *principle of specification*, which counteracts the tendency to reduce each object in the manifold to the abstract *object in general* by attempting to articulate each of the minute differences inherent in each object through the application of concepts to this object. Kant argues that because a concept contains in itself only what is common to different things[59] it can never point *directly* to an individual object, but can only seek to specify its concepts further and further to approximate a proper determination of any object. This is why Kant states that there are *no species infima*, which confronts us with the fact that in principle the concept tree has an *infinite* depth. We can deal with this problematic introduction of infinity into our computational environment by assigning to it a *potential infinity*, the thoroughgoing determination of each object thus becomes *specification as a task*.[60] Every concept is given content by specifying its intent and extent, and ultimately the highest concept of *object in general* is only given content by specifying what these objects can be. This is given as an infinite task constituting the drive for systematic investigation. Where the *principle of homogeneity* thus gave us criterion for the truth of concepts through the application of Occam's Razor to the whole conceptual system arising from the application of these concepts, thus a criterion through systematic *unity*, the *principle of specification* provides us a criterion for the truth of concepts by the continual specification - and thus confirmation or falsification - of these concepts through the complementary notion of systematic *investigation*. Yet to prevent a descent into infinite regress by pursuing this infinite task in arbitrary fashion we need to keep in mind the *relevance* of certain concepts and investigations. What if the first cavemen merely tried to zealously execute the *principle of specification* by investigating all the properties of a single rock, forgetting the potentially more important investigation into the conceptual properties of foodstuff? A proper notion of purposivity and relevance is thus intimately linked to the construction of a conceptual system, even in the case of a purely theoretical scholastic system. This is occluded by the fact that *we* as researchers are selecting the sensory sequences as input to the AE, thus deciding which sort of concepts are relevant to include in the conceptual system the AE is building. If we would let the AE build its conceptual system 'by itself', it could (and without any agency would) just stare at a blank wall or a random input stream for all eternity. This is exactly why for Kant the *architectonic* system incorporating ends of reason is not a secondary addition to the purely theoretical *scholastic* system, for in reality the latter is a mere deficient representation of the former, even if as a limited representation it can be useful

[59] See footnote 58

[60] "Now the understanding cognizes everything only through concepts; consequently, however far it goes in its divisions, it never cognizes through mere intuition but always yet again through lower concepts. The cognition of appearances in their thoroughgoing determinacy (which is possible only through understanding) demands a ceaselessly continuing specification of its concepts, and a progress to the varieties that always still remain, from which abstraction is made in the concept of the species and even more in that of the genus." (A656/B684) This makes concrete the remark by Longuenesse in footnote 42, according to which cognition is itself already a striving to find a further rule.

for its own reasons.
Another aspect of Kant's dictum of *no species infima* is his notion of *codivision*, which indicates the fact that every concept can in principle be applied to every other concept.[61] This notion of codivision is where the third principle of systematicity comes in, called the law of the affinity of concepts, or the continuity of forms, which Kant arrives at by combining the first two laws of higher genera and infinite specification, and indicates that every concept can be applied to every other concept because they share the same genus and the same infinite specification by every other concept through the principle of thoroughgoing determination.[62] Both the principle of no lowest species (infinite depth) and of codivision (infinite breadth) bring a problem of infinity to the table, but while the infinite depth of specification is handled by being treated as a *task*, the infinite breadth of codivision can only be handled through constraint and a notion of the relevance of the operation of codivision. The real question is which of all the potentially infinite concepts are actually *relevant* to apply to the concept at hand. Both of these problems thus involve defining the *ends* of investigation, which Kant captures in his *architectonic concept* of a system. Expanding the AAE to gain agential features is outside of the scope of this thesis, thus we will choose to forego these problems by focusing on the *scholastic concept* of a system, thus a system unified merely for the sake of unity. This means an investigation and system simply geared towards the classification and mechanical explanation of objects,[63] where we as researchers will decide which situations are relevant to pursue in systematic fashion instead of the AE devising which sensory situations it would need to investigate to meaningfully expand its system of concepts.[64]
The task at hand is then to devise a scholastic system capable of efficiently classifying and explaining a presented sensory sequence.

3.3.2 Building the tree

To start to build the (or a) system of concepts we will take the *template* and the *theory* we just discussed for the Sokoban example, and show how we can construct a genus/species tree τ given the cognitions produced by the understanding. [65] The aim is to include every line in the template and the theory as a mark of at least one of the concepts in the tree τ, so that the whole template as well as the theory could be

[61] Or at least checked if it applies or not.

[62] See the quote at the start of this section.

[63] Thus the earlier discussed distinction between a system that can classify plants using the least amount of rules versus a system that classifies plants according to their medicinal value, which could have much higher complexity.

[64] We thus take over one role of Reason: systematic investigation. This will prove to be an crucial limitation of the AAE, and a probably fruitful avenue for future work. We will note along the way were we take over this role.

[65] "Reason never relates directly to an object, but solely to the understanding and by means of it to reason's own empirical use, hence it does not create any concepts (of objects) b but only orders them and gives them that unity which they can have in their greatest possible extension, i.e., in relation to the totality of series; the understanding does not look to this totality at all, but only to the connection through which series of conditions always come about according to concepts. Thus reason really has as object only the understanding and its purposive application, and just as the understanding unites the manifold into an object through concepts, so reason on its side unites the manifold of concepts through ideas by positing a certain collective unity as the goal of the understanding's actions, which are otherwise concerned only with distributive unity."(A643/B671) A complaint both technical and philosophical can be raised here, as it is exactly the point of my thesis to argue that a system of concept (or at least the drive to systematicity) is needed to direct the use of the understanding, and that without this the understanding can not properly function, while here it seems we only start building the system of concepts *given* the autonomous actions of the understanding, raising the question if the system of concepts is *really* needed for the operation of the understanding. This is a deep and intricate question that will get a full response in the later philosophical section, which we will thus bracket for the time being, while raising the preparatory points that in fact the understanding has already been guided by *our* reason on the following points: 1) in the providing of the categories of the understanding, which contain the seeds of the systematicity of reason, 2) in the providing of a salient sensory sequence for cognition instead of mere random noise or an unmoving white wall, etc. (thus we enact the role of *systematic investigation*) 3) in the providing of the template for the apperception task, which might have been generated on its own but which would be computationally infeasible (thus we solve the *Search Problem* for the AE by acting as

recalled from the tree τ when the sensory situation calls for it, thus if some or all of the concepts can be successfully applied to a new sensory sequence. This will allow us then to sequentially and systematically construct the template used for producing a theory for a sensory sequence (depending on if the inclusion of a concept can allow us to produce a theory that *makes sense*), instead of simply iterating over templates in a diagonal fashion, like is done now. But before we can discuss the business of using the system of concepts to apply concepts to new sensory sequences, we need to construct this system of concepts.

First we will do this in the naive way, without including *a priori* concepts in the tree, except the root node *object in general*[66], under which the different types T in the template will be placed as species. We then take each of these types to be concepts in their own right, and determine the marks and constraints that fall under each type in the following way:

1. *Essential marks:* E is the set of all predicates, variables, objects, rules and constraints in which the concept figures. These can be seen as the necessary consequences of the application of the concept, and are added to the template and theory if the necessary conditions of their application are also met.
2. *Sufficient marks:* Σ is the minimal set of essential marks that allows one to differentiate the concept from all other concepts. These are the sufficient conditions of the application of the concept, which means that if all of these are present the concept has to be included in the template.
3. *Accidental marks:* A is the set of objects, predicates and initial conditions which are not present in *all* objects falling under the concept over *all* timesteps of the sensory sequence. (For example the spatial position of a Player object, or the genus Object being either Player or Block. It is essential that it is either one of these, but accidental *which* one.
4. *Extensional constraints:* C_e is the set of constraints that act as *disjunctive judgements* and determine the sufficient marks of the concepts lying in the extension of the concept under consideration. (Thus we see that the constraint that determines an Object has one of the two possible shapes determines with that the sufficient marks of the concepts of Player and Block.)
5. *Intensional constraints:* C_i is the set of constraints that determine the accidental marks of the concept. (Thus for example the constraint specifying that an Object can only be in one cell at a time.)

Once the types are instantiated as concepts the *extensional constraints* C_e are taken up and each of the marks under the disjunctive judgement is made a *sufficient mark* for the concept in the extension of the specific type, and for this concept the same operation is performed again, to determine which rules figure in this concept. In this way we recursively build up a concept tree τ, as shown in figure 3.

3.3.3 Philosophical justification of the *a priori* part of the system of concepts.

Now we have seen the first construction of a tree τ from the cognitions produced by the understanding, but as of yet it is only constructed in *a posteriori* fashion, uniting the the products of the understanding in a manner determined in full by the understanding. In part this is of course correct, as the contents and discoveries of the understanding are supposed to determine the system of concepts to some extent, but in

its memory, and lastly 4) by deciding on the correct theory out of all the generated theories (thus solving the *Theory-Choice Problem* for the AE by providing a criterion for decision and deciding on it).

[66] "The highest concept with which one is accustomed to begin a transcendental philosophy is usually the division between the possible and the impossible. But since every division presupposes a concept that is to be divided, a still higher one must be given, and this is *the concept of an object in general* (taken problematically, *leaving undecided whether it is something or nothing*). (A290/B346, emphasis mine) Note the fact that it is left undecided whether the object is something or nothing, a notion that will return in the following sections.

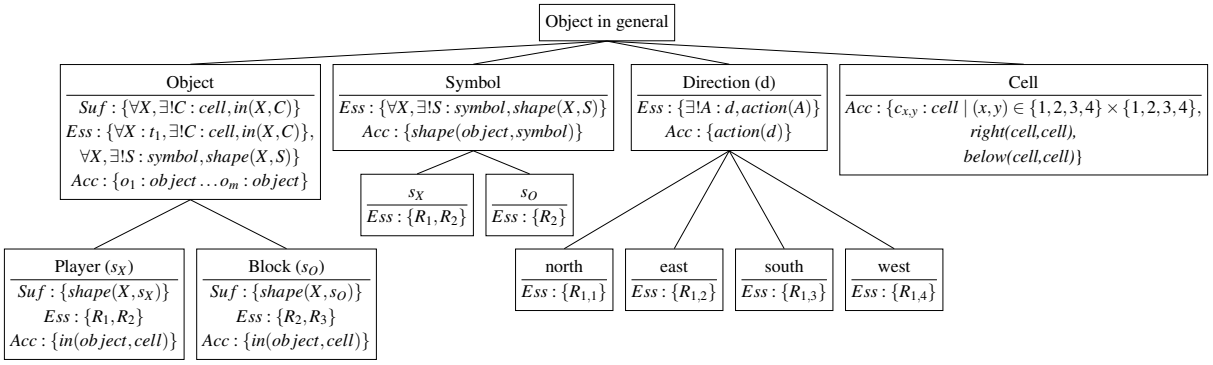


Figure 3: Visual representation of a naive concept tree τ.

the Kantian view the system of concepts is supposed to be articulated *a priori*, with the empirical part determining it only from downstream. Let's investigate the following quote:

> "Under the government of reason our cognitions cannot at all constitute a rhapsody but must constitute a system, in which alone they can support and advance its essential ends. I understand by a system, however, the unity of the manifold cognitions under one idea. This is the rational concept of the form of a whole, insofar as through this the domain of the manifold as well as the position of the parts with respect to each other is determined a priori. The scientific rational concept thus contains the end and the form of the whole that is congruent with it. The unity of the end [der Einheit des Zwecks], to which all parts are related and in the idea of which they are also related to each other, allows the absence of any part to be noticed in our knowledge of the rest, and there can be no contingent addition or undetermined magnitude of perfection that does not have its boundaries determined a priori. The whole is therefore articulated (articulatio) and not heaped together (coacervatio);a it can, to be sure, grow internally (per intus susceptionem)[from an internal cause] but not externally (per appositionem)[by juxtaposition], like an animal body, whose growth does not add a limb but rather makes each limb stronger and fitter for its end without any alteration of proportion." (A832-3/B860-1)

This is the purpose of an Idea, acting as the *end* or *purpose* of investigation, as taking an as yet indeterminate Idea of the whole as the purpose of inquiry allows us to judge that and how the system is still inadequate to the Idea of the whole, and subsequently how our inquiry should seek to fill the gaps, which could only be seen *as gaps* by seeing the present system of concepts as a limitation of the Whole, which is unconditioned and unlimited. Because the Idea of the whole transcends our capacities of comprehension, we can properly say that the grasp of the Idea of the whole on which we orient ourselves is the *schema* of the Idea, which is an ever changing approximation depending on the progress we make in our scientific inquiry.

This *a priori* part is thus on the one hand articulated as the *Idea* of the whole, which is a projected totality acting as the seed, purpose and limit of inquiry, only able to articulated afterwards, but guiding it from the beginning in an unconscious fashion, while on the other hand, and more concretely explicit, we have the *schema* of the whole, which is the actual plan according to which the inquirer investigates nature, which provides both a practical plan for conduct as well as an *a priori* structure of concepts under which the empirical concepts can be brought. For the exposition of the schema we quote Kant again:

> "For its execution the idea needs a schema, i.e., an essential manifoldness and order of the parts determined a priori from the principled of the end. A schema that is not outlined in accordance with an idea, i.e., from the chief end of reason, but empirically, in accordance

> with aims occurring contingently (whose number" one cannot know in advance), yields technical unity, but that which arises only in consequence of an idea (where reason provides the ends a priori and does not await them empirically) grounds architectonic unity. What we call science, whose schema contains the outline (monogramma) and the division of the whole into members in conformity with the idea, i.e., a priori, can not arise technically, from the similarity of the manifold or the contingent use of cognition in concreto for all sorts of arbitrary external ends, but arises architectonically, for the sake of its affinity and its derivation from a single supreme and inner end, which first makes possible the whole; such a science must be distinguished from all others with certainty and in accordance with principles." (A833-4/B861-2)

The schema of the idea has two sides, the one being practical for guiding the conduct of investigation, the other being theoretical, for providing the *a priori* structure of the system of concepts. The practical operation of the schema of the Idea, captured in the *Method* and *Architectonic* of reason, is a continual attempt at specifying the method and aims of investigation, according to the *essential ends* of reason (both practical and theoretical), which seeks at constructing a purposive system of the *Whole* of nature and freedom. In practice this boils down to determining which concepts and objects are relevant and worthwile to investigate, in this manner guiding the conduct of the inquirer towards the proper articulation of the Whole. This does not entail that all inquiry is supposed to be predetermined in its purposes, as it is not possible to truly and correctly articulate the purposes of inquiry without being in possession of the actual products of this inquiry. We must thus see it more as a recursive loop between free and relatively undirected inquiry according to a certain intuition or partly articulated ends, and the subsequent proper (but of course ultimately partial) articulation of the purposes of this inquiry in relation to the whole of the system and its Final End, which then allows us to start this free inquiry again from a new vantagepoint. [67]
The theoretical side of the articulation of the schema of the Idea involves Kant's conception of role of transcendental philosophy and the new role of metaphysics, which is to articulate *a priori* the foundational structure of the system of reason and the *general* objects and *material* objects it envelops. Transcendental philosophy seeks to determine the structure of any *concept* in general, while the Metaphysics of Morals articulates the *a priori* domain of morality,[68] with the Metaphysics of Nature articulating the *a priori* structure of the *material* object,[69], and finally the *Transition* from metaphysics to physics articulating the way the metaphysical structure of the material object results in the *a posteriori* but systematic structure of the laws of nature through the structure of the ether.[70]
This *a priori* theoretical part of the system of concepts is what we will focus on now, attempting to include it in part in the tree τ. I will not yet attempt to include the full extent of this *a priori* part, on the one hand because it is not fully articulated by Kant, on the other hand because it is in a higher order of complexity regarding the properties of material objects than is meaningful or apt in relation to the relatively simple sensory sequences presented to the Apperception Engine. For starters we will include a

[67] "Nobody attempts to establish a science without grounding it on an idea. But in its elaboration the schema, indeed even the definition of the science which is given right at the outset, seldom corresponds to the idea; for this lies in reason like a seed, all of whose parts still lie very involuted [eingewickelt] and are hardly recognizable even under microscopic observation. For this reason sciences, since they have all been thought out from the viewpoint of a certain general interest, must not be explained and determined in accordance with the description given by their founder, but rather in accordance with the idea, grounded in reason itself, of the natural unity of the parts that have been brought together. For the founder and even his most recent successors often fumble around with an idea that they have not even made distinct to themselves and that therefore cannot determine the special content, the articulation (systematic unity) and boundaries of the science." (A834/B862)

[68] Which is the whole domain of morality, as any empirical claims regarding morality are for Kant only technical or instrumental concerns, which do not properly belong in the domain of moral-practical reason.

[69] As set out in the *MFNS*.

[70] As partially articulated in the *OP*, but not fully worked out yet by Kant.

part from the Transcendental Philosophy which is useful to our present purposes, namely the division of the *object in general* into Something and Nothing, allowing us to place the Cell concept as an articulation of the Cartesian structure of space under the concept of space.[71] This allows us to non-arbitrarily, thus in *a priori* fashion, posit the fact that the specific instantiation *in* predicate is accidental to any other concept, as for Kant the Cartesian spatial position is always accidental.[72] The import and usefulness of this we will see later, when we look at examples of concept formation that include completely stationary objects, thus objects that keep the same *in* predicate over all timesteps.

3.3.4 Building the non-naive tree

To properly understand the breadth of the concept of 'the object in general' - and thus its position as the root concept - we must appreciate that Kant takes it to mean 'object of thinking' in the loosest terms, thus merely as an object that could figure in a judgement of general logic, as a 'One'. It thus stands above the possible and impossible, even above being something or nothing, because to define anything as being contradictory or empty of content we must first grasp it as an object able to be negated.[73] Since the categories are the only concepts that relate to objects in general, the distinction of whether an object is something or nothing must proceed in accordance with the order and guidance of the categories. Kant's description of the structure of Nothing according to the categories is given here:

> 1) To the concepts of all, many, and one there is opposed the concept of that which cancels everything out, i.e., none, and thus the object of a concept to which no intuition that can be given corresponds is == nothing, i e., a concept without an object, like the noumena, which cannot be counted among the possibilities although they must not on that ground be asserted to be impossible (ens rationis), or like something such as certain new fundamental forces, which one thinks, without contradiction, to be sure, but also without any example from experience even being thought, and which must therefore not be counted among the possibilities.
> 2) Reality is something, negation is nothing, namely, a concept of the absence of an object, such as a shadow or cold (nihil privativum).
> *3) The mere form of intuition, without substance, is in itself not an object, but the merely formal condition of one (as appearance), like pure space and pure time, which are to be sure something, as the forms for intuiting, but are not in themselves objects that are intuited (ens imaginarium).*
> 4) The object of a concept that contradicts itself is nothing because the concept is nothing, the impossible, like a rectilinear figure with two sides (nihil negativum)" (A290-2/B346-9, emphasis mine)

We have taken this description and give an example of a non-naive tree including *a priori* concepts and the *Forms of Intuition*in figure 4. Yet we will not dwell on this much longer, as we merely have sought to justify and show the strange position of the concepts related to the forms of intuition, but as we are not engaged with the construction of intuition directly in this thesis, it will suffice to have noted its place in the system of concepts. We now proceed to the process of concept application.

[71] Which is of course not properly a concept, but a pure intuition, but this is still *conceptually* articulated.
[72] Include quote from Amphiboly.
[73] Compare footnote 66

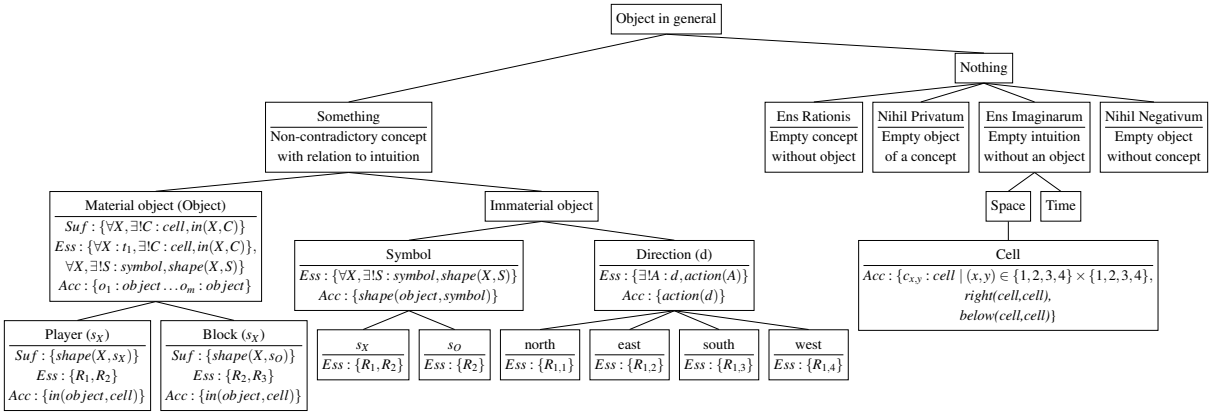


Figure 4: Visual representation of a non-naive concept tree τ containing some a priori elements.

3.4 Concept application

For the purpose of concept application once we already have a system of concepts, rough as it may be, I propose a new method for *making sense* of a sensory sequence. Instead of performing a diagonal search over the possible concepts (in the form of types, rules, constraints, etc.) incorporated in a template to find a template that generates the least-cost theory, we will use recursive calls to the tree τ to step-wise construct one or more templates, including parts or the whole of the theory implied by such a template. Being able to use the concepts learned in previous instances, thus having *memory*, will in my hypothesis prove to be crucial to attacking the *Search Problem*, as not only do we avoid costly diagonal search for templates, we also improve upon the time-complexity of generating a *theory* once an appropriate template has been found, as the rules and constraints corresponding to specific concepts in the template can simply be recalled from memory.

Let's start with the simplest way to do such a step-wise construction: for each object presented in the sensory sequence[74] we consult first the root node of the tree τ and in depth-first fashion choose one of the concepts in its extension as defined by the disjunctive judgement represented by the extensional constraints. The determination of choice in the extension relies on the sufficient marks present in the object in the sensory sequence, thus in the Sokoban example we can see that if object in question has the property *in(object, cell)* we can determine it to be a *material* object as opposed to an *immaterial* object.[75] Once we have chosen such a concept we repeat these same steps one level lower, each time adding the rules, predicates, constraints, etc. found in the intension of the concept in question to the template and theory under construction, until we have reached the bottom of the tree τ for the marks we have now.[76] Once we have repeated this for every object in the sensory sequence we have to check if the generated theory actually *makes sense* of this sensory sequence, and if it does not this will give us the prompt to revise our system of concepts, which we will describe in more detail in the following sections. Let us first inspect the simplest case of the making sense of a sensory sequence using a tree τ constructed for a functionally similar sensory sequence.

[74] Note that this is already not so simple when we are presented with a raw sensory sequence, where the AE has to define the limits of the object itself. We will return to this, but for now add that to solve this we will also have to incorporate the *schema's* embodied in the neural network of the AE in the system of concepts.

[75] Later we will see how determining the extension by rules instead of by predicates will provide a considerably larger challenge, but also provide ground for interesting innovations to the AE.

[76] In principle the tree can have no bottom of course, but this is because each time we reach a lowest node we get the imperative to further specify this lowest concept, not because the tree is *already* infinitely deep. Later we will improve upon this end-state by adding the possibility of *making partial sense* of a sensory sequence.

3.4.1 Similar sensory sequence

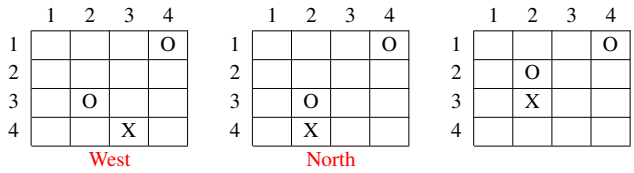


Figure 5: A sequence functionally similar to the earlier Sokoban example shown in figure 1, which is made sense of by the same theory with different initial conditions.

In figure 5 we are presented with a sensory sequence that should not require a different template or theory, except for the initial conditions, than the Sokoban example given in figure 1, yet without a proper form of memory the AE is not able to utilize this similarity. With the idea of the system of concepts caught in the tree τ it can reconstruct this template and theory in the following fashion.

For simplicity's sake we will use the naive concept tree τ as presented in figure 3. In addition we include *a priori* the Cell type to the template, as the sensory sequence is always presented spatially as well as temporally, where in principle the spatial extent and form does not need to be given *a priori*, but can be left open to be determined *a posteriori*.[77] The Object type, as the representation of the concept of a material object can also be added to the template *a priori*, as every example for now (as well as in Evans [2020] involves spatial objects.[78] Now let us take a look at the symbolic state input in figure 6, here we can simply inspect each object figuring in this sensory sequence. Let us do this step-wise, where each step includes the selection and traversal of a branch (concept) for an object, as well as the consequences following from this selection:

Step 1:[79] We start by taking up object o_1 and try to progress along the concept tree τ by starting at the root node *object in general* and seeing if any of the predicates assigned to the object allow us to go down a branch. As it happens the $in(o_1, c_{3,4})$ predicate is a sufficient mark for the Object concept,[80] which means that if the o_1 also instantiates all essential marks in the concept Object we can add all these essential marks to our template and theory. If it does not instantiats all essential marks, then either something is wrong with our concept of Object, or with our observation, an essential issue we will return to later. As it stands the o_1 object does fulfill all essential marks, thus as it is essential to the Object concept to figure in a binary *shape*, we can immediately add the Symbol class to our template and theory, but will refrain from including its extension until one of the species s_X, s_O has been presented in the sensory sequence.

[77] As in the *Figurative Apperception Engine* in Soeteman [2022]

[78] A *material* object in distinction from a *spatial* object obviously involves a lot more, but we need no more than a minimal description of a material object as spatial object for the purposes of the AE for now.

[79] Drawn in red in figure 7

[80] This is a slight invention for explanatory purposes, as o_1 is properly speaking already placed under the Object concept in the template, as otherwise it could not be presented as symbolic input.

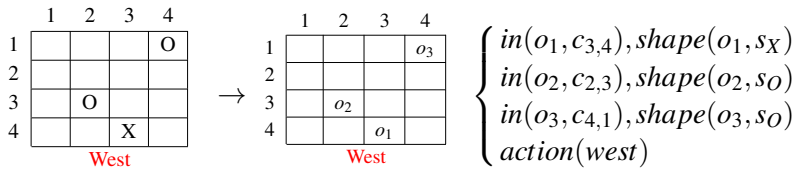


Figure 6: The visual example translated to the symbolic state given as input.

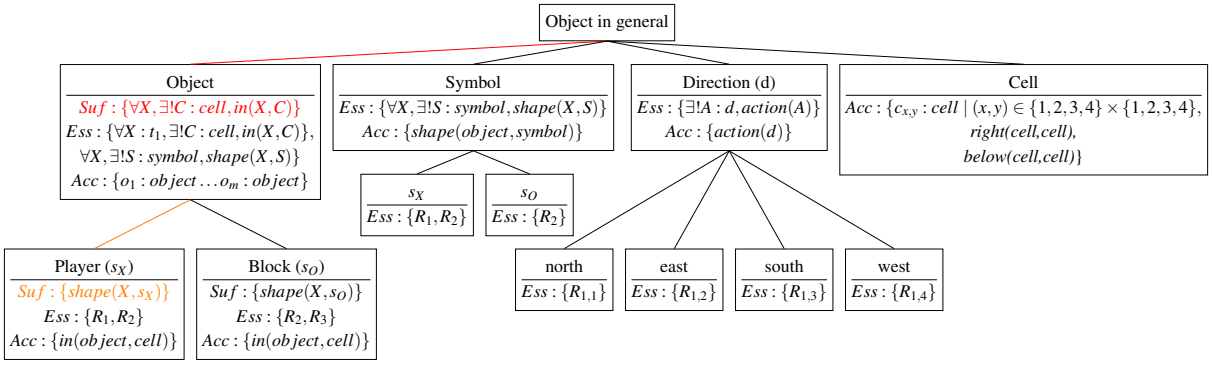


Figure 7: Visual representation of the stepwise progression of the tree, where two steps are taken down the branches of the naive concept tree τ to place the object o_1 under the correct concept. These branches for each step are respectively colored red and orange, as well as the sufficient marks determining which branch is taken.

Step 2:[81] Finished with the immediate consequences of applying the Object concept we try to get down another step and for that inspect the extensional constraint encoding the disjunctive judgement found in the Object concept, which postulates that the sufficient mark for the extension of Object is captured by the binary *shape* predicate. Thus we look for a *shape* predicate featuring the Object o_1, which we of course discover to be $shape(o_1, s_X)$. From this it follows that this object falls under the Player (or simply s_X) concept, which allows us to add the marks in this concept, as well the marks for the s_X concept, to our theory and template. In the rules figuring in this concept several other concepts are also contained, namely each of the directions, as well as $shape(Y, s_O)$. These can be looked up in concept tree τ to check if any of them figure as sufficient marks for a concept (which they do), and then to check in the sensory sequence if they are present (which they are), to be able then to add these concepts and their marks to the theory and template. And just so we were able to fully populate both the template and the theory by inspecting only the first timestep of the provided sensory sequence, which we designate as the initial conditions. Of course each timestep afterwards is checked to see if there is nothing missing from the theory (and in this case the North direction will still be added to the template),[82] as well as checking if the generated theory and template truly *make sense* of the sensory sequence, which they do.

3.4.2 Extra piece

Adding a new piece will allow us to start unfolding the dynamics of the concept tree τ, and gives us the opportunity to introduce a new concept, namely the *dummy concept*, or indeterminate concept X_d. This concept is added to each disjunctive judgement that is not determined *a priori*, as each empirical concept could in principle gain a new species in its extension, while the *a priori* disjunctions, between for example Something and Nothing, can not gain an extra term through the process of empirical investigation.[83] It acts essentially as a flag for the Apperception Engine to start a proper *apperception task* for a newfound object, instead of simply recalling concepts generated in previous *apperception tasks*. This allows the AE

[81] Drawn in orange in 7

[82] A note on the directions is that because East and South have not been seen in the sensory sequence, they will properly *not* be added to the template and theory. This then also means that the rules involving the South and East directions will not be included, which is acceptable because the AE can *make sense* of the sensory sequence without these, and because they are stored in the system of concepts we can be certain that *if* they come up we can add these again to make sense of the new situation.

[83] "A division into two members is called dichotomy; but if it has more than two members, it is called polytomy. All polytomy is empirical; dichotomy is the only division from principles a priori, hence the only primitive division. For the members of a division are supposed to be opposed to one another, but for each A the opposite is nothing more than non A. Polytomy cannot be taught in logic, for it involves cognition of the object. Dichotomy requires only the principle of contradiction, however, without being acquainted, as to content, with the concept one wants to divide." (JL 147)

engine to still generate new concepts when encountering objects that have not been previously recorded, while learning a new attribute for already existing concepts will involve a different process which we discuss later. To make more sense of this new concept X_d we will work through an example involving the addition of a new piece, as can be seen in figure 8. The first steps of this example are identical to the steps described in the previous section 3.4.1, but once the AE encounters the predicates involving object o_4 we can observe new behaviour. We present it again stepwise:

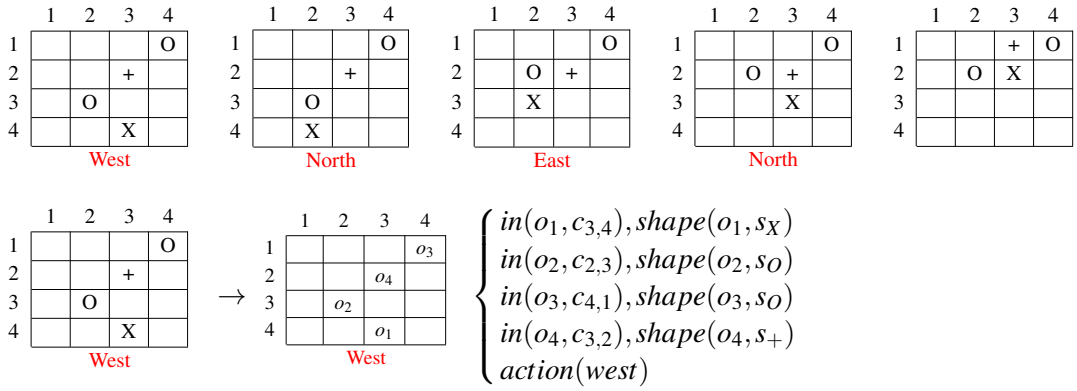


Figure 8: A game of Sokoban that includes a new piece, the + piece, that behaves the same way as the O piece. The second row shows the translation to a symbolic state given as input to the AE.

Step 1:[84] First it encounters the $in(o_4, c_{3,2})$ predicate,[85] which is a *sufficient mark* for the Object concept, thus it can bring the object o_4 under the Object concept[86], which does not have the consequence of populating the template and theory with the marks falling under the Object, as this has already been done in earlier steps, but *does* prompt the AE to try and determine to which species specified in the extensional constraint o_4 belongs.

Step 2:[87] Prompted to take a second step when trying to place o_4 under a concept the AE will check if it figures in any predicates that act as sufficient marks in the extension of Object, which it does not. Then it checks if there is another predicate it figures in, which in this case is $shape(o_4, s_+)$. This *shape* predicate features a new symbol s_+, which is not yet present in the concept tree τ, but *is* present in the input and the template.[88] This has several consequences, the first being in this case the inclusion of s_+ as a species under the Symbol concept, which subsequently allows $shape(X, s_+)$ to figure in the extensional constraint in the Object concept, thus we can take $shape(X, s_+)$ to be the sufficient mark for a new concept, which is now put in place of the dummy concept X_d, under Object. It does not yet contain any rules, as these still have to be constructed in the final step.

Step 3: As we have encountered a dummy concept the AE is prompted to run an *apperception task* after every predicate in the input sequence has been checked, with the template and theory constructed above. As we can run the *apperception task* with a template and theory already filled in for the Player and Block objects, it will be able to run with a much lower time-complexity than if it would have to be run from the

[84] Drawn in red in figure 7

[85] Properly speaking it does not matter which of the predicates figuring o_4 it encounters first, as once it encounters a new object the AE will first gather each predicate it figures in and subsequently use the predicate that acts as a *sufficient mark* for a branch from the *root node* to take the first step. If there are no predicates that act as a sufficient mark a node, be it the *root node* or a deeper one it will of course select a *dummy concept* at that level to start an *apperception task*.

[86] Which as noted in footnote 80 it already is, but let us pass over this for the exposition.

[87] Drawn in orange in figure 7

[88] As we are dealing with symbolic input now any new shapes given in the input already have to be inserted by us in the template, as otherwise the AE can not use it as input. This changes for the *raw* sensory sequences, where the dummy concept will have an even more important purpose.

ground up. Now this *apperception task* will of course be able to find a rule-set covering the behaviour of the new + piece, as it falls under the same rules as the Block concept. In figure 9 below we indicate these rules as R_2' and R_3'. A primed rule-set will indicate a copy of the rule-set with the sufficient marks switched.

Step 3.5: Another more efficient way, that is heuristically justified by the fact that species of the same genus oftentimes share behaviour, is to take the rules for every concept falling under the genus the new concept will be placed under, and swap in the sufficient mark for the new concept - $shape(X, s_+)$ - for the sufficient marks figuring in the rules for the old concepts, in this case either $shape(X, s_X)$ or $shape(X, s_O)$. This will allow us to check if the new concept is not simply a form of one of the old concepts, which in this case it is, which we can observe when the theory containing the rules for the Block concept swapped with the new $shape(X, s_+)$ predicate succeeds in *making sense* of the sensory sequence.

Step 4: There are two ways we could go about including the new concept for the + piece in the system of concepts now that we know it is identical in all but appearance to the Block concept. The first and simplest would be to simply add a concept besides the other concepts under the Object concept, like in the isolated tree fragment on the in figure 9. This solution leaves something to be desired, as with respect to its essence the + piece falls under the same rules as the Block, while it differs merely in appearance. A second way to solve this would be to take both the O piece and the + piece as species of the Block concept, where we will rewrite the rules R_2 and R_3 to be agnostic with respect to the *shape* of the Block piece, replacing both s_O and s_+ in the predicate with a variable, which will be filled in once either of the shapes has been observed in the input. We indicate this variable rule-set with a star like so: R_2^*, R_3^*. The Block concept now has two different sufficient marks relating to its shape and when one of these shapes is observed, it populates the template and theory with the essential marks with the specific shape predicate filled in. Because both pieces have the exact same rule-set, it is accidental which shape the Block object has, and we do not have to produce a new disjunction under the Block concept with s_+ and s_O as species, as this is only needed when they differ in their essence. This solution is shown in the tree fragment on the right in figure 9. Ultimately the system would prefer the tree on the right, as it is able to reduce the amount of nodes in the tree by the inference that s_+ is in its essential marks the same as the s_0 object, namely a Block piece, while only differing in their appearance, their accidental marks. Now that we have shown this addition of a piece to the system of concepts we will move on to the computational implementation.

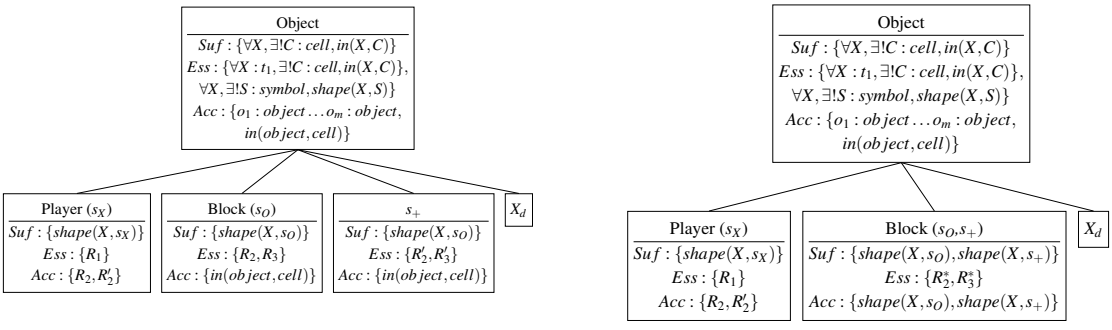


Figure 9: Two possible tree fragments encoding the addition of the + piece to the concept tree τ as specified in figure 7, with the addition of dummy concepts.

4 Computational Implementation of the System of Concepts

In this chapter we will first go over the adapted concept tree τ we use in the Architectonic Apperception Engine, which we simply call the *AAE tree* or *AAE concept tree*, using a simple sensory sequence as an example environment to explain the basic functioning of the AAE tree. After looking at this example we will discuss the justification for the changes made, like the disregard of constraints and the exclusion of the dynamic between the different marks. Some of the further details of the AAE tree will be left open in this chapter, as they will be taken up for presentation in the Results chapter. This might be an unorthodox approach, but in my opinion it is best to show the details of the sequential build-up of the AAE concept tree through this sequence of real examples. But before we dive into the Results this chapter ends with a quick overview of the code added to the Apperception Engine to expand it to our Architectonic Apperception Engine.

4.1 System of Reason

In the previous chapter we have discussed the construction of the system of reason or concept tree τ according to Kant's considerations as expounded at various point in his writings, though mainly in the CPR, JL and MFNS. While this is a justified possible sketch of the ideal situation, it is yet too expansive for the proof-of-concept we seek to build for the AAE. This means that we will mostly do away with talk of essential, sufficient or accidental marks, as well as the focus on the constraints, as we will simplify the nodes to merely denote types T, with the *sufficient mark* for such a type/node being the presence of a concept and object contained under the type. Previously we could describe a whole interplay between the essence and appearance of each object as caught by the different marks, which in turn then defined the position of the concept for that object in the concept tree τ, yet now, by merely focusing on the predicate[89] indicated in the sensory sequence, we can reduce the complexity of the problem by leaps and bounds. To be able to properly explain the construction process and structure of the AAE concept tree we will go through the following simple example where we take a single alternating object as input, indicating the specifications of the AAE concept tree along the way.

In the following example the AAE is presented with a simple sensory sequence, on which it then runs an *apperception task*, thus effecting apperception by synthesizing a Unity of the Understanding, whose concepts the *system construction* process will then use to start constructing a Unity of Reason in the form of a concept tree. We thus start with the empty *Root* node[90] and provide it an input to initiate the construction of the tree. In this simple case we do not need to present it with a template up front, as the AAE can generate a fresh template for the input.[91] If the concepts and objects in the input have not been seen yet the AAE will generate a new type, whose name will be based on the name of the concept for the sake of human readability. Yet for the sake of clarity in the explanation we will provide the AAE with a template up front, which assigns the concept and objects to the t_object type. But keep in mind that this is not necessary for this sequence, and we will provide an example of such a generated template in section 5.1.2. Note also that we do not consider the number of arrow rules, causal rules and body atoms, as these

[89] Which we will from now on in the parlance of the AAE call a concept, while what we previously called a concept, thus a node in the concept tree τ, will simply be called a type or node.

[90] Above we called this with Kant the *object in general*, but while this is an important concept in the sense that specification only makes sense when it is specifying this *object in general*, thus giving the empty concept content through its specification, yet in this computational proof-of-concept it will only lead to unnecessary confusion. This is why we choose to simply call it the Root node and leave it empty.

[91] Note that it can as of now only generate simple templates, the combination with a larger iteration process to generate more complex templates has not been optimally implemented yet, though it is possible to simply use the diagonalization without the tree to generate a template which will will be taken up into the tree after a successful *apperception task.*

are either iterated over or simply directly inferred from the reproduced template and theory, thus they are not reported. A consequence of this is that we will use the terms frame and template interchangeably in what follows. Now let us showcase the functioning of the AAE using the simple case of a single object alternating between two states, *on* and *off*, allowing us to go through the processes of system construction and concept application step by step.

4.2 Construction and application in the computational setting

4.2.1 Alternating object sequence

We take the following straightforward sensory sequence encoding object a alternating between *on* and *off* states as input.

$$S_1 = \{on(a)\} \quad S_2 = \{off(a)\} \quad S_3 = \{on(a)\}$$
$$S_4 = \{off(a)\} \quad S_5 = \{on(a)\} \quad S_6 = \{\}$$

Then we provide the following frame $\phi = (T, O, P, V)$, where:

$$\phi = \begin{Bmatrix} T = \{object\} \\ O = \{sensor_a : object\} \\ P = \{off(object), on(object)\} \\ V = \{X : object\} \end{Bmatrix}$$

Running the apperception task the AAE then finds the theory $\theta = (\phi, I, R, C)$, where:

$$\begin{aligned} I &= \left\{ \; on(sensor_a) \; \right\} \\ R &= \left\{ \begin{array}{l} on(X) \ni off(X) \\ off(X) \ni on(X) \end{array} \right\} \\ C &= \left\{ \; \forall X : object, \; on(X) \oplus off(X) \; \right\} \end{aligned}$$

First we can inspect the initial condition I, which in this instance will not be taken into consideration by the system constructor, as the fact that $sensor_a$ starts in the *on* condition is of course contingent to the *particular* situation, and thus not taken up into the by definition *general* concepts folded into the tree.[92] Yet in other cases the constructor does consider I, as the so called Permanent concepts - like the successor relation between the letters a and b - are also contained in this set, and these are indeed general. This we will discuss in a later example.
In the case of the rules R we can see that the Apperception Engine learns the two simple rules needed to explain the sequence: $on(X) \ni off(X), off(X) \ni on(X)$ - when the state of the object is on, switch it to off and vice versa.
Lastly we are presented with the constraints, but as we will discuss in the next section these are given in the input and not actually learned, so while these are reported for the sake of completeness they are not as of yet taken up into the AAE tree.

4.2.2 Construction phase

Now the system construction stage kicks in and first the types T, objects O and concepts P figuring in the frame ϕ are taken up into the concept tree τ, which starts as the empty *Root* node. Importantly the

[92] This initial condition and the temporal relations between the different states would of course be important for the *TheoryLayer* or history of Memory we discussed in section 2.1, but this goes beyond the scope of the thesis.

AAE simplifies the concept tree by constructing the nodes of the tree τ simply on the basis of the *types* in the frame ϕ. Each type seen in the frame ϕ gets assigned to a node, and each node is thus just a type, containing objects, concepts and rules *in* it, and other types *under* it. In this case we have only one type, *t_object*,[93] which is thus placed as a node directly under the Root node as can be seen on the left side of figure 10.

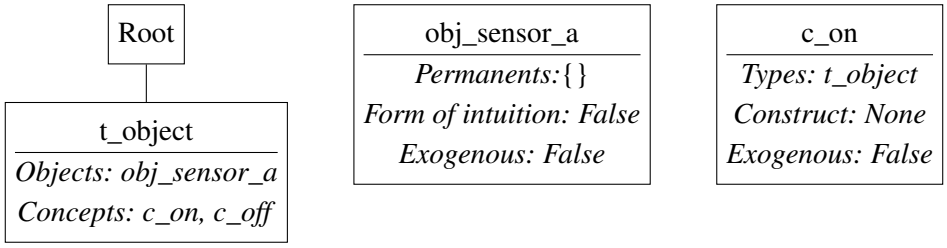


Figure 10: On the left is the AAE concept tree τ constructed using the output of the apperception task, in the middle is an overview of how the obj_sensor_a object is stored in the tree and on the right is an overview of the concept c_on as stored in the tree.

When a type from ϕ is assigned to a node, or is already present as a node in the tree, the objects in O that instantiate this type are added to the node. Here the only object is $sensor_a$, which we add to the t_object node as obj_sensor_a. As seen in the middle figure of figure 10 each object contains multiple values that are kept track of in the tree:
1) The *Permanent facts* it figures in, like the successor or part_of relations, which is empty for obj_sensor_a.
2) If it is added as *Form of Intuition*, thus added as a spatial or temporal 'object', like a cell or time-step. This is important to keep track of as these are not concepts in the usual sense, for they should be added by a process of intuition, but do factor in certain rules or permanent facts. This is set to False in the current case, for while obj_sensor_a is of course a spatial object, it does not constitute space as a form of intuition in this case.
3) If they were added as an *Exogenous* object to the template, thus one that doesn't need to be explained by the Apperception Engine, but still figures in the template. This is set to False here as this object does need to be explained.

After adding the objects instantiating the type to the node the constructor adds the concepts from P in which this type figures to the node. In this case P contains two unary concepts *on* and off, which are added to the node as c_on and c_off. If the system encounters a binary concept, where the second type is a different type from the one currently under consideration and not yet present in the tree it recursively performs the above steps for the newly found type, as it will only add a binary concept to the tree if both its types are present. We will inspect this later in a further example. In the present instance it can simply add both unary concepts to the node, where each of these concepts contains multiple values, as can be seen in the case of c_on on the right side of figure 10. The values are defined as follows:
1) The *Types* that figure in the concept, which is of course most important for binary concepts, as for a unary concept we should already know the type the concept is contained under. Still it is important that we differentiate concepts like *c_on* by the type it applies to, for while *on* and *off* might seem to mean the same for different types of objects, their content, thus their behaviour defined by rules, is not strictly the same. The assignment of rules to concepts will thus always check for both the concept and the type to

[93] When discussing the types, concepts and objects in the concept tree we will respectively prepend $t\_$, $obj\_$, and $c\_$ in front of items in question for the purpose of disambiguation, as sometimes types and objects can have the same name. Permanent facts, even though they are in principle concepts, also get their own prefix $p\_$, as they fulfill a different role than the regular concepts. This notation also reflects the notation in the computational implementation, and will make it easier to navigate the examples.

which the concept applies. We come back to this point in the next section when we incorporate the rules into the concept tree.
2) The *Construct* flag, which indicates that a concept is *not* a permanent concept if it *None*, and if it *is* a permanent concept it indicates if the permanent concept is *Given* - thus if a fact is provided stating that *b*is the *successor* to *a* - or has to be *Constructed* - thus if the *apperception task* has to define the successor relation between *a* and *b*. In this case it is *None*, as c_on is *not* a permanent concept. The other cases we will leave for exposition in the next chapter.
3) The *Exogenous* flag, which just as for the object class above indicates that the concept does not need to be explained by the apperception engine. In this example there are no exogenous concepts, thus this is set to False.

Lastly the variables V in the frame ϕ are not taken up into the tree, but they are renamed to reflect their type, thus going from X to var_object_1, to reflect the fact that it is the first variable for the t_object type. This is to better keep track of which variable corresponds to which type in the rules contained in R.

Now that each item in the frame ϕ has been added to the tree the permanent facts in I and rules in R are processes to be added to their corresponding nodes. First the permanent facts in I are added to the nodes, as these can figure in the rules and are thus conditions for the rules to be properly added. These are added in a similar fashion to the concepts, and we will show this addition in an example in the next chapter, specifically in section 5.1.3.
After the permanent facts have been added the constructor inspects the Rules R and attaches each of the rules under the concepts that figure in the rule. Take the rule $\mathit{off}(X) \ni \mathit{on}(X)$, it contains two concepts $\mathit{on}(X)$ and $\mathit{off}(X)$, where we know that the variable X represents the object type, allowing the constructor to place the rule under concept with the right type. As discussed above distinguishing the concepts by the types it applies to is important, precisely for the sake of the rules that constitute the content of these concepts. For example, in this instance the apperception engine has been able to make sense of the behaviour of the sensor_a object alternating between *on* and *off* by postulating that when sensor_a is in the *on* state it will switch to the off state in the next time-step. By the generality of these rules we make the assumption that in any future case the sensor_a object will exhibit the same behaviour, which allows us to reproduce these rules in a new sensory sequence involving sensor_a in the *on* and off states. This assumption or principle of regularity,[94] thus of the inherent generality of concepts and stability of objects, is exactly what underlies the reproduction of concepts in memory, thus what allows us to assume that we can *reapply* these same concepts in a new situation, making it fruitful to construct a memory for these concepts. Yet it is important to limit this generality of the concepts by a specificity[95] regarding these concepts, as when we encounter the same concept *on* applied to a new object like a $\mathit{desklight}$, we are not warranted to infer that the concept still has the same content, thus that this $\mathit{desklight}$ will alternate between *on* and *off* at each timestep, making it a *strobelight*! The concept of *on* thus needs to be *specified* further according to the object it is applied to, and the other concepts present in the sensory constellation, which is precisely by including the types the concepts apply to in the rules, which make up the content of these concepts. A rule is thus located under the specific concept/type combinations that figure in it, as can be seen in figure 11 for the case of c_on(t_object).
As can be gleaned from figure 11 rules are saved in a special way in the AAE, as they are internally represented as *facts* for the clingo environment, consisting of a head (the consequent) and a body (the conditions), kept track of through a shared rule index. The Datalog$^{\ni}$ representation of the rule seen in the header, $\mathit{on}(X) \ni \mathit{off}(X)$, is thus only presented here for readability, but not present in the computational

[94] Which Kant would call a transcendental condition of systematicity.

[95] Notice that this discussion mirrors the theoretical considerations surrounding Kant's principles of systematicity from section 3.3.1, but then in a practical setting.

environment. A rule is thus saved with the following values:
1) The *Rule head* denotes the consequent of the rule and is a single line saved as an ASP fact.
2) The *Rule body* denotes the conditions of the rule, which can be multiple lines, are saved as ASP facts.
3) The *Concepts* denote the different concept/type combinations present in the rule, allowing us to refer to the other concepts in the rule when inspecting a rule from a specific node.

Now each relevant piece of information produced by the apperception engine has been taken up into the tree, and so we can proceed to the *application phase*, where the AAE concept tree τ is used to *make sense* of the same sensory sequence, without the apperception engine having to synthesize a new theory.

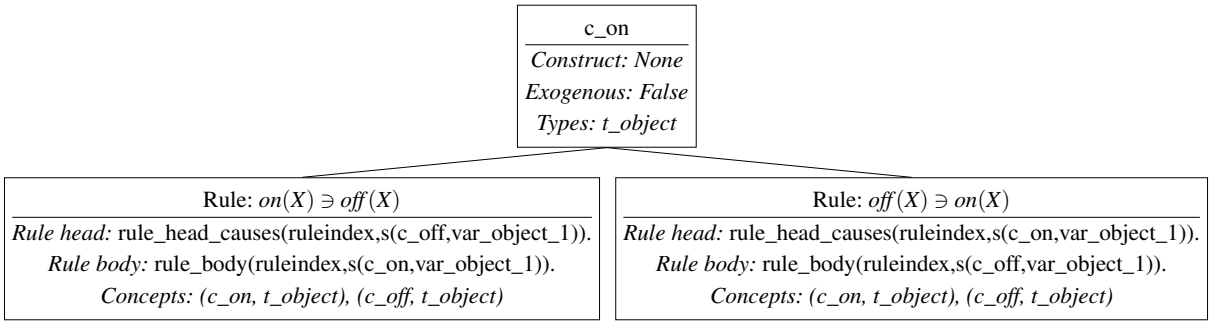


Figure 11: Overview of how the concept c_on is stored in the tree, including the rules stored under the concept.

4.2.3 Application phase

The *application* phase takes a *sensory sequence* and a concept tree τ and will try to construct multiple hypothetical frames and theories for those frames consisting of the permanents and rules learned in previous instances. Then it lets the apperception engine iterate over these frame/theory pairs to check which has the lowest combined cost. For this purpose we have also adapted Evans' cost function to also include how many types and concepts are present in the frame, besides the amount of objects, rules and ground atoms.[96] As we will see in practice later it is necessary to devise multiple hypotheses, for even with both a sensory sequence and a tree τ of low complexity there are multiple possible types applicable to the concepts and objects in the sensory sequence. Furthermore we can not simply stick to only including the concepts and objects seen in the input to the frame, as most sequences require the postulation of concepts and objects beyond the sensory data to *make sense* of the sequence. These speculative concepts can thus also not be deterministically added, but will be part of one of multiple hypotheses to explain a sensory sequence.[97]

We again supply the following sensory sequence to the AAE, but this time it will have to use the (of course still very simple) AAE tree τ built in the previous section to construct a frame ϕ and reproduce the concepts learned in the previous instance.

$$S_1 = \{on(a)\} \quad S_2 = \{off(a)\} \quad S_3 = \{on(a)\}$$
$$S_4 = \{off(a)\} \quad S_5 = \{on(a)\} \quad S_6 = \{\}$$

Now we note that internally this sensory sequence is presented as follows:

```
senses(s(c_on, obj_sensor_a), 1).
senses(s(c_off, obj_sensor_a), 2).
```

[96] Done by adding the weak constraints `:~is_type(A). [1 @ 1, A]` and `:~is_concept(A). [1 @ 1, A]` to `costs.lp`

[97] This is the practical consequence of Wittgenstein's rule paradox discussed in section 1.2

```
senses(s(c_on, obj_sensor_a), 3).
senses(s(c_off, obj_sensor_a), 4).
senses(s(c_on, obj_sensor_a), 5).
hidden(s(c_off, obj_sensor_a), 6).
```

When presented with the above sequence the AAE checks for each time-step if the concept/object combination, thus for the first time-step (c_on, obj_sensor_a), is contained in any of the nodes, thus can be applied to any of the types in the concept tree τ. Now there are several distinct possibilities that arise here, each of which could lead to a different hypothesis for the frame. If there are multiple options the AAE postulates both options as hypotheses, and recursively continues its walk through the time-steps for each hypotheses, branching off at every new split. The AAE can split or continue in the following cases:
1) Both the concept and the object are contained in the concept tree τ *and* they figure in the same type/node and are not found in any other type/node. In this most straightforward case there is only one possible option for the inclusion of a type in the frame, which means this is directly included into the frame ϕ and the AAE can go onto the next time-step without branching off. Now keep in mind that because our conceptual system is a genus-species tree, when an object is a subtype, like t_sensor, of another type, like t_object, it is at the same time that higher type, thus both t_sensor and t_object should be included in frame ϕ.[98] This can be relevant when a previously learned rule has been observed figuring the higher type t_object, but not the lower type t_sensor, yet could still be included in the theory for the present instance. Thus each of the types above the current type under consideration in the hierarchy is included in the frame ϕ when a hypothetical theory is constructed from the tree. The problem here is that including each supertype can add a lot of redundant information to the frame, slowing down the *apperception task*, thus we also generate hypothetical frames without the supertypes.[99]
2) The concept is is found in the tree, yet the object has not been encountered yet. In this case there are usually multiple types to which a concept can be applied, and thus we create hypothetical frames for each of the different types the object can take. If the different types are part of the same branch we always choose the lowest one, thus these hypotheses are made up only of the leaves of the tree. There is of course the case where none of types in the leaves can be properly used to explain the sensory sequence, but the proper handling of this requires the addition of the *dummy concept* to solve, which is outside the scope of this thesis. The current implementation takes a simpler route and refers to the solution for possibility 4. We will come back to this in section 4.3 on the justification of the changes made to the AAE tree.
3) The object is found in the tree, but the concept is not yet known. This case is fairly straightforward, as the type the object instantiated can simply be added to the frame, as well as the concept, which when the frame is sufficient to explain the sensory sequence can be added to that type.[100]
4) Both the concept and the object can not be found in the tree. Here, in the absence of a more complex solution fitting it at specific part in the tree, the system simply assigns a newly made type falling directly under the Root node to the object and concept.

Once every time-step has been inspected and each concept and object assigned a type in one of the hypothetical frames ϕ, each of these hypothetical frames is again branched into further hypotheses. First

[98] Note that the internal representation of the template contains a field to indicate the hierarchy of the different types, which is used here to construct and apply such a hierarchy.

[99] This is of course a question of relevance, for is it always relevant to include in our explanations of the behaviour of a dog to include the fact that it falls under the genus of multi-cellular organisms? Including such a supertype only becomes relevant when we can include a concept and a rule from that supertype in our explanation of the object in question.

[100] In contradistinction to the possible complexity of accommodating the conceptual system to a new object, which might not fit any previous type, we see that adding a new concept to an old object is quite straightforward. Yet we should not forget that this is a simplification, and the addition of a new concept to an old object can of course also lead to the wholesale restructuring of the conceptual system. This lies outside the scope of our proof-of-concept.

from each type the speculative concepts and their postulated objects are added, thus concepts not found in the input but previously postulated to explain the sequence, branching into minimal (using only input concepts) and speculative hypotheses. Then for each type the supertypes in the hierarchy are added, again branching into minimal and expanded hypotheses. Lastly we add possible objects constituting space, like grid and cell objects, which have not been seen yet in the input, again branching into minimal and expanded hypotheses.[101] Finally variables are added for each type, with two variables for a type if there is a binary concept containing twice the same type, like the successor relation for the letter type.

Our system thus reproduces the frame $\phi = (T, O, P, V)$ from the tree τ seen in figure 10. In this case there is only a single hypothetical frame, because the tree is small enough that the process is completely deterministic:

$$\phi = \begin{Bmatrix} T = \{object\} \\ O = \{sensor_a : object\} \\ P = \{off(object), on(object)\} \\ V = \{OBJECT_1 : object\} \end{Bmatrix}$$

Once a frame ϕ has been constructed we construct hypothetical theories for each frame. First for each object its permanent facts are added if the conditions of application apply, thus for example p_successor(a,b) is added if the successor concept is in the frame, and both objects are in the frame. In this case no permanent facts need to be added.
After the permanent facts have been added we check for each concept in the frame if any of the rules contained in the concept can be applied, thus if each of the types, concepts and objects present in the rule can be found in the frame. For the rule $on(X) \ni off(X)$ we see that both concepts c_on and c_off are present, and that there is an object of the correct type t_object present, allowing the system to add it to the theory. The same applies of course for the other rule.[102] Now the AAE has produced the following theory for the sensory sequence, before it even needs to start the costly *apperception task*. The AAE reproduces the theory $\theta = (\phi, I, R, C)$, where:

$$\begin{aligned} I &= \{\} \\ R &= \begin{Bmatrix} on(OBJECT_1) \ni off(OBJECT_1) \\ off(OBJECT_1) \ni on(OBJECT_1) \end{Bmatrix} \end{aligned}$$

Now we notice first that the constraints have not been added, for reasons we will divulge upon in the next section. Secondly we see that the initial conditions I are still empty, as they should be, for this can not be determined a priori through the system of concepts, but must be synthesized from the sensory sequence. The beauty is though that to check this frame and theory we still need to run the same *apperception task*, which can then trivially add the initial conditions if the theory works to explain the sensory sequence. And in this case, because we can see that the two theories are identical, when the *apperception* task is run on the sequence with the frame and theory provided by the system of concepts, it shows that this theory does indeed *make sense* of the sequence.

We have timed, taking the average of 10 runs for each setting, starting with the simple apperception task and a premade frame, without accessing the memory tree. The second setting reproduces the frame and theory from the memory tree, without iterating over hypotheses. Our default hypothesis in this case is

[101] In future work, as we will explain in the justification section and the discussion, it would be better to procedurally synthesize space and time, thus combining our approach with that of Soeteman [2022].

[102] Now there is a problem that incompatible rules can still be added, which will be discussed in the next section and is reserved for future work.

one without speculative concepts, supertypes or forms of intuition. This works here of course because the tree and sensory sequence are still very simple. Lastly the third setting goes through the iteration of hypotheses as described above, resulting in 8 hypotheses, as we currently branch off into minimal and expanded hypotheses three times. The results are presented in table 3 below.

Process	**Time in seconds**	**Cost**
Synthesizing a theory with a premade frame (baseline)	0.07	8
Frame and theory from memory, with a single hypothesis	0.07	8
Frame and theory from memory, with 8 hypotheses	0.28	8

Table 3: Table showing the averaged time over 10 runs for making sense of the alternating object sequence.

We can see that there is little difference between the time taken to reproduce a theory or to synthesize it anew in this simple case, but this is itself a positive outcome as the most we could hope for in the simple case is that the AAE is not slower than the baseline. Once we factor in iteration over hypotheses the AAE does of course become slower than the baseline, but this will factor out in later, more complex cases. In regards to optimization each setting finds the exact same solution, thus the cost of each solution is the same. With this example we have shown that and how the Architectonic Apperception Engine works to learn and reproduce concepts and theories, solving in part the **Memory Problem** by implementing a rudimentary form of the *System of Concepts* layer. For the first time, the Apperception Engine can be called a true machine *learner*, retaining and reproducing the concepts for the purpose of synthesizing the manifolds presented to it. Before we will go on to expand on these results, and show its ability to tackle the **Search Problem** it rests us to justify the changes made when transitioning from the theoretical idea of the system of concepts to the computational implementation in the AAE, and to provide a quick overview of the code used to effect this implementation.

4.3 Justification of implementation

Several changes have been made to the system of concepts presented in chapter 3 to be able to implement it in the computational setting. In this section we will list and give justifications for these changes, as well as indicating how they could be tackled in future work. We will go throught these in order.

4.3.1 Marks

The different marks expanded upon in the theoretical section, thus the *sufficient*, *essential* and *accidental* marks have not explicitly been implemented in the system of concepts used in the AAE. Of course implicitly we have used *sufficient marks* to decide which types will be included in the frame ϕ, namely the combination of concepts and objects observed in a time-step of the sensory sequence now determine if a type can be included in the frame; the appearance of both in the type is a *sufficient* condition for that type to be included. Once such a type is included the system also includes its supertypes, and the conceptual content in the form of rules that follow upon the addition of these types, concepts and objects, thus the functionality of the *essential marks* is also at play in our implementation.
Yet our system is a lot more rigid now that we don't use the explicit categories of the marks, as it is not possible to dynamically change which attributes of the concept can act as *sufficient marks* for its inclusion, and which aspects are merely *accidental marks* related to its appearance. Including this behaviour would necessitate the on the fly rearrangement of the locations of specific nodes in the system of concepts, as changing for example an *accidental mark* into an *essential mark* could mean creating a new concept that needs a new position in the tree. If we consider it accidental to the essence of an apple if it is *red* or *green* we can make do with the single concept apple that contains *red* and *green* as *accidental marks*. But once

we postulate that a *green* apple does not only appear different from a *red* apple, but actually constitutes a different *species* under the *genus* of apple, with its own essentially different conceptual content, like a *sour taste* over against a *sweet taste*, we need two add *green apple* and *red apple* as their own nodes under the *apple* node. The *greenness* of the apple then transforms from an *accidental mark* into an *essential*, even a *sufficient* mark, denoting a different species. This would also entail the possibility of having nodes that are not types, which is also not possible in the current computational implementation.
This process of dynamically updating the tree for the purpose of specification and differentiation is an essential part of the full-fledged system of concepts, but is not yet necessary for our proof-of-concept of this system in the apperception engine environment, as we can already reproduce concepts and theories with a more static tree. We term the tree a static tree instead of a dynamic tree, because once an item has been added to the tree it will stay at this position forever, as there is not yet any functionality to alter, move or delete a node. This will be reserved for future work.

4.3.2 Constraints

As discussed earlier we do not consider constraints when constructing the system of concepts for the simple pragmatic reason that in the current apperception engine environment these constraints are supplied upfront in the input, instead of learned. This seems contrary to how it is reported in [Evans, 2020] as there it is stated on pages 60-61 that the constraints are synthesized by the apperception engine in the *apperception task*, yet they are always supplied in the input, and when I tested it by removing the constraints in the input the AE did not generate the constraints on it own. I suspect that Evans removed this feature in the final code for optimization purposes, as the apperception engine should in principle be able to generate these constraints. Amending this was beyond the scope of the thesis, and because the constraints are in the current setup not generated in the *apperception task* I have kept to the convention of supplying them in the input, foregoing their inclusion in the system of concepts.

4.3.3 Iteration over amount of body atoms and rules

As this is in its simplest form a problem of iteration we have not included any consideration to the amount of body atoms and rules in the system of concepts. For ease of use we have simply supplied it up front in the cases where no larger iteration is needed to find new types, as iterating this from scratch can be done but takes too much time in practice. This is why I don't report these amounts when discussing the frames produced by the AAE. For future work it would be interesting to see if this iteration can be improved by giving starting estimates based on how many body atoms and rules are included by the AAE into the theory, allowing us to also cut down on the iteration in this case when compared to the base apperception engine, but for now this is beyond the scope of this thesis.

4.3.4 Forms of Intuition

As discussed in section 3.3.3 on of the *a priori* section of Kant's system of concepts it is important to distinguish the objects and concepts that constitute the structure of the Forms of Intuition, thus space and time, from the *a posteriori* synthesized by the apperception engine. These *a priori* concepts and objects behave in a different fashion then the *a posteriori* concepts, in that they are needed to frame the environment of the application of the other concepts, but can not acquire full-fledged conceptual content: they are in an important sense *empty* concepts, or more precisely empty *intuitions*.
They are empty in the sense that while one can determine relations between for example cells in Cartesian space, thus defining cell_1_1 to be above cell_1_2, yet beyond their mutual relations with other *a priori* "objects" they can can not stand in a definite relation to any *a posteriori* object or concept. Cell_1_1 is

conceptually indistinguishable from cell_1_2, as it is a category mistake[103] to assert that some object has *this* behaviour when in cell_1_1 and *that* behaviour in cell_1_2, except when speaking in a specific context beyond the forms of intuition, like on a chessboard. We can thus not simply save these *a priori* objects and their relations in the system of concepts, nor should we, as the Kantian way to handle the problem of space is it procedurally generate it according to the necessitation of the sensory sequence. The problem is namely that when get a sensory sequence as input like with sokoban, the sensory facts only specify the spatial locations, the cells, of the objects, but not the spatial cells between and beyond these objects. We can not simply save all the spatial cells we have previously seen and add them to the frame when we encounter one of the spatial cells in this web, as when the tree grows larger we would have to include an unworkably large amount of spatial cells or concepts in the frame, not knowing when to stop adding cells as these cells can not have definite conceptual content, for in one situation cell_4_4 might indicate the end of the grid, while in another it is just a cell among cells. Merely conceptually we can thus never determine where the end of our grid should be.[104] Space and time should thus be procedurally generated by the Imagination, as in [Soeteman, 2022], instead of conceptually constituted.
Still in the present case we have not yet combined the AAE with the FAE proposed in [Soeteman, 2022], which means we needed to find a practical work-around. Right now the spatial structure is thus in some cases simply given in the input, and in other cases in a limited form provided by the system of concepts. We will come back to this when we encounter examples in the next chapter.

4.3.5 Specification and dummy concepts

As was noted earlier, the addition of new types and objects is less sophisticated in the AAE then our proposal using the *dummy concept X*[105] in the theoretical section. Instead of advancing continually through the tree by disjunction until we can not assign any of the subconcepts to the unknown object in question, necessitating us to consider a new node and type at that specific location for the unknown object, we simply generate a new type and place it directly under the Root node. Our system of concepts is thus in some sense not fully engaged in the process of specification, as it can not straightforwardly generate new species for old genera, for now it just generates a new species directly under the Root node. While in principle implementable through a more involved generation of new types this is left for future work, as it is unnecessary for the proof-of-concept. This concludes our discussion of the changes made in the computational implementation as compared to the theoretical exposition of the system of concepts ventured upon in chapter 3.

4.4 Code

In this section we will provide a quick overview of the code employed to implement the system of concepts in the AAE.[106] In the course of this overview I will continually refer to the code diagram drawn in figure 12 on the next page, where the *green* units indicate files containing data, the *red* units refer to the programs of the base apperception engine running the *apperception task* and finally the *blue* units refer to programs constituting the system of reason part of the AAE. The code for the AAE is built on top of the original apperception engine[107], which is written in Haskell and ASP, where the ASP programs are solved

[103] Or for Kant a mistake in ones transcendental *topic*, as one seeks to relate to the understanding what must be related to intuition.

[104] This is the practical implication of Kant's problem of the Antinomy of Space in CPR, which he of course solves by treating space not as constituted by the Understanding, but by Intuition. These same concerns lead us to suggesting combination of the AAE with the FAE presented in [Soeteman, 2022]

[105] Representing Kant's singular judgements

[106] The code of which is accessible at https://github.com/afalbrecht/apperception

[107] Accessible at https://github.com/RichardEvans/apperception

by Clingo.[108] Our system of reason is written in Python,[109] with the AAE tree τ consisting of a tree of nested dictionaries, able to be converted to JSON and stored in pickle. When handed an `[input].lp` file by `solve.hs` the program in `memory_in.py` reads in the AAE tree from `mem_tree.pickle` and generates hypothetical templates and theories in the *application phase* of the system of reason. The templates generated by the system of reason which are meant as input to the the Clingo solver are stored in simple `.txt` files, parsed by Haskell functions executed from `solve.hs`, which hands it over to Clingo to run an *apperception task*. At the same time the theories generated by the system of reason are written out as ASP programs in `.lp` files, which can be directly accessed by Clingo. Finally `solve.hs` parses the best theory and template provided by Clingao and writes these into the same file format for `memory_out.py` to use them in the *construction phase*, updating the tree with the newly learned concepts and storing it again in `mem_tree.pickle`.

Now that we have a clear view of the program loop with the code diagram we can make digress into a philosophical diversion to make the relation between Reason and the Understanding more concrete, as it is now enacted in the relation between our addition of the system of concepts and the original apperception engine. From a global view at figure 12 we can observe the full structure of the AAE, namely that the system of concepts acts in some sense separately from the base apperception engine, mirroring the interaction between Reason and the Understanding described by Kant. The system is prompted by a sensory sequence given as input,[110] which in the base apperception engine would be taken up by the Understanding to synthesize into a transcendental unity of the Understanding, which in the computational environment translates to entering into the Haskell handler,[111] where it is either assigned a prebuilt template, or several iteratively constructed templates, which are then passed Clingo to run an *apperception task* as an ASP program.[112] Here Sensation passes directly into the Understanding, which 'mechanically' constructs templates and synthesizes the manifold or sensory sequence into a transcendental unity.[113] Once it has *made sense* of the sensory sequence it promptly forgets all the concepts it has arduously synthesized, to repeat the process wholly anew when prompted with another sensory sequence. Now ultimately this process has not changed drastically, but in the AAE we have added another faculty described by Kant, namely Reason, which as discussed seeks to synthesize these concepts generated by the understanding into a higher unity, namely the Unity of Reason, which we to be a systematically structured memory.[114]

The faculty of Reason, when taken in its theoretical aspect as memory can only start acting after the Understanding has already performed an *apperception task*, so we properly have to begin at the end, namely at the moment `solve.hs` feeds the best template[115] and theory found in the *apperception task* to `memory_out.py`, starting the *construction phase* of the system of reason. Here the generated concepts are added to the AAE tree τ, resulting for example in the tree in figure 10 we discussed in the previous chapter. Reason thus sifts through the concepts generated by the Understanding in its Unity of the Understanding

[108] [Gebser et al., 2014]

[109] Specifically Python 3.11.6

[110] Thus entering the faculty of Sensation in Kant's view.

[111] Whose executor is `solve.hs`, as can be seen in figure 12.

[112] Consisting in the basic case of `judgement.lp`, `constraints.lp` and `costs.lp` as can be seen in figure 12.

[113] As distinct for the power of judgement for Kant, which can be modelled as a heuristic algorithm through the memory tree, seeking to apply concepts to sensation: "The reflective power of judgement thus works with given appearances to bring them under empirical concepts of determinate natural things not schematically, but technically, not merely mechanically, like a tool controlled by the understanding and the senses, but artistically according to the universal but nonetheless indeterminate principle of a purposive and systematic ordering of nature." (CJ20:213/4)

[114] This is of course the theoretical side of Reason, practical Reason is more than just memory.

[115] Note that here it does not really matter if the template was generated by the AAE, found through diagonalization or provided by us, as without anything in memory the template generation of the AAE is identical to diagonalization (though with more readable types), which are both equivalent to the handmade template by construction.

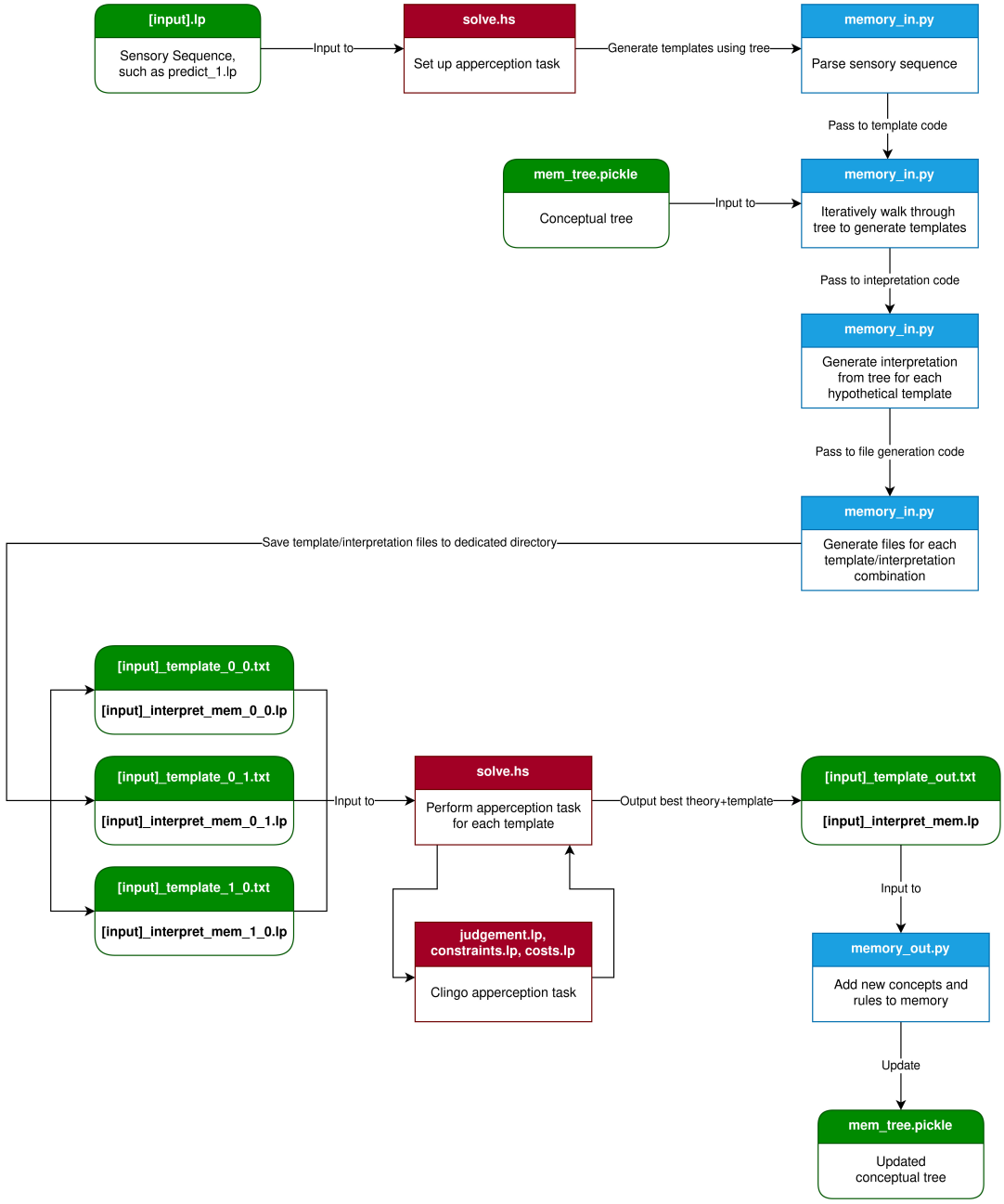


*Figure 12: Schematic overview of the code. The **green** units indicate files containing data, the **red** units refer to the programs of the base apperception engine running the apperception task, finally the **blue** units refer to programs constituting the system of reason part of the AAE.*

and takes these up, abstracts them[116] into higher concepts and principles and synthesizes these into its own Unity of Reason, instantiated here as the AAE tree. Now that we the AAE tree has content, when we feed a similar or identical input into `solve.hs` at the top, the *application phase* operated in `memory_in.py` can kick in, prompting it to generate templates and theories from memory based on the sensory facts it encounters in the sensory sequence. The movement through the faculties is thus from an agitation of Sensation to a manifold for the Understanding, at which point Reason steps in to save the busy Understanding the work of iterating through every possible template,[117] looking through its Memory

[116] Our implementation does not do this yet, but this should be done in future work.

[117] Which would in any realistically complex setting require infinite compute, at least if we agree with Kant that Intuition is a singular infinity, thus extends infinitely broad and deep.

at the previous concepts the Understanding has generated to present it with several hypothetical theories it can use to start *making sense* of this new situation. Reason thus sets the Understanding to work in a quasi-mechanical fashion, providing it the concepts with which it will then try to synthesize the sensory manifold, or solve the *apperception task*! Once it has either confirmed the previous theory or found a new theory to *make sense* of the sensory sequence this whole loop starts again at the end, continuing further and further to specify the concept of an *object in general*, aiming to ultimately construct the Total Unity of Reason as the end of science.[118]

5 Results

In this section we will present the results obtained by the Architectonic Apperception Engine on different apperception tasks to showcase its ability to partially solve the **Memory Problem** and the **Search Problem**, as well as highlighting the foundations and limitations it presents for solving the **Theory-Choice Problem**. In the previous chapter we already showed the first result, using it as an example to show how the AAE partially solves the **Memory Problem**. This we will continue to expand upon here with more results, thus the first section will consist of learning concepts for *simple tasks* and reapplying these concepts to the same task again, thus exemplifying simple *reproduction*. Then in the next section we discuss the application of the AAE to the *complex task* of Sokoban, showing both the ability to apply concepts to different but similar environments and the the speed-up it achieves by applying the concepts it has learned in previous situations. Let us then start with presenting the results.

5.1 Simple Memorization

In this section we will continue the gradual construction of a conceptual tree as begun in the previous chapter using simple yet real examples. As a start we take the tree in figure 10 constructed in the last chapter, expanding it with each new example. Some sensory sequences will require us to first run an *apperception task* with a pre-built template, to then store the generated concepts in memory, which can only then be used to generate a template and theory to explain the sensory sequence again. In other cases the AAE tree might already have the right concepts to construct a template for a sensory sequence, thus allowing it to not just simply reproduce a template for an identical sequence, but to produce a new template for a similar sequence to the ones it encountered previously. The first cases we discuss will be of the first variety, thus simple reproduction, but gradually we will expand this behaviour with different results. Let us then start presenting the results without further ado.

5.1.1 Two objects

We continue building the tree with a sensory sequence containing two objects, where the first object is the alternating object from the previous example, and the second object a simple sensor always set to *a*. The objects themselves don't pose a new challenge, but the fact that there are two together means the AAE will need to postulate a spatial structure to keep them apart. As our current approach does not yet implement a more sophisticated way to constitute space, we need to provide a pre-built template

[118] But as we noted earlier even this view of the role of Reason is not strong enough, as when we take practical reason into account we realize that Reason does not simply doze off until the Understanding is prompted with a random manifold by Sensation, but it actively directs the Understanding through the field of Sensation for its own ends, ends inherent in Reason, ultimately culminating in the Final End of Humanity, the realization of the Kingdom of Ends as the Highest Good. The end of science turns out to thus be just one of the self-imposed ends of Reason, prompted by its ownmost desire for the True, the Good and the Beautiful, united in the Kingdom of Ends. At least that's if we believe Kant of course!

including a spatial structure. Once this template and theory is included in the AAE tree we can reproduce them for the same situation. This is shown below.

5.1.1.1 Construction phase using pre-built template

Given the following input:

time	$sensor_a$	$sensor_b$
1	?	?
2	*off*	*a*
3	*on*	*a*
4	*off*	?
5	?	?
6	*off*	*a*
7	?	?

Table 4: Input for the two objects problem

We provide the frame $\phi = (T, O, P, V)$, where:

$$\phi = \left\{ \begin{array}{l} T = \{cell, grid, object, sensor_1, sensor_2\} \\ O = \{sensor_a : sensor_1, sensor_b : sensor_2, grid : grid\} \\ P = \{a(sensor_2), b(sensor_2), c(sensor_2), off(sensor_1), on(sensor_1), part(cell, grid)\} \\ V = \{S1 : sensor_1, S2 : sensor_2, C : cell, G : grid, X : object\} \end{array} \right\}$$

Our system finds the theory $\theta = (\phi, I, R, C)$, where:

$$\begin{aligned} I &= \left\{ \begin{array}{l} a(sensor_b) \\ on(sensor_a) \\ part(sensor_a, grid) \\ part(sensor_b, grid) \end{array} \right\} \\ R &= \left\{ \begin{array}{l} on(S1) \ni off(S1) \\ off(S1) \ni on(S1) \end{array} \right\} \\ C &= \left\{ \begin{array}{l} \forall X : sensor_2,\ a(X) \oplus b(X) \oplus c(X) \\ \forall X : sensor_1,\ on(X) \oplus off(X) \\ \forall X : cell,\ \exists! Y : grid,\ part(X, Y) \end{array} \right\} \end{aligned}$$

Here the rules R remain the same as in the previous instance, as $sensor_b$ can simply be explained by indicating that its initial state is a, due to Evans' Frame Axiom stating that any state holds each consecutive time-steps if there is no rule changing this state.[119] This new template and theory is then added to the previously built tree, to produce the tree shown on the left side of figure 13.

The important additions to the tree are of course also the spatial concepts, thus t_cell and t_grid, which as noted earlier we can not yet neatly divide under their own *a priori* part of the tree, as they are not generated yet *a priori*. To keep track of which types constitute forms of intuition, we ourselves inject the information that the *t_grid* objects and *t_cell* objects constitute Forms of Intuition using the earlier

[119] As defined in 1.3.1

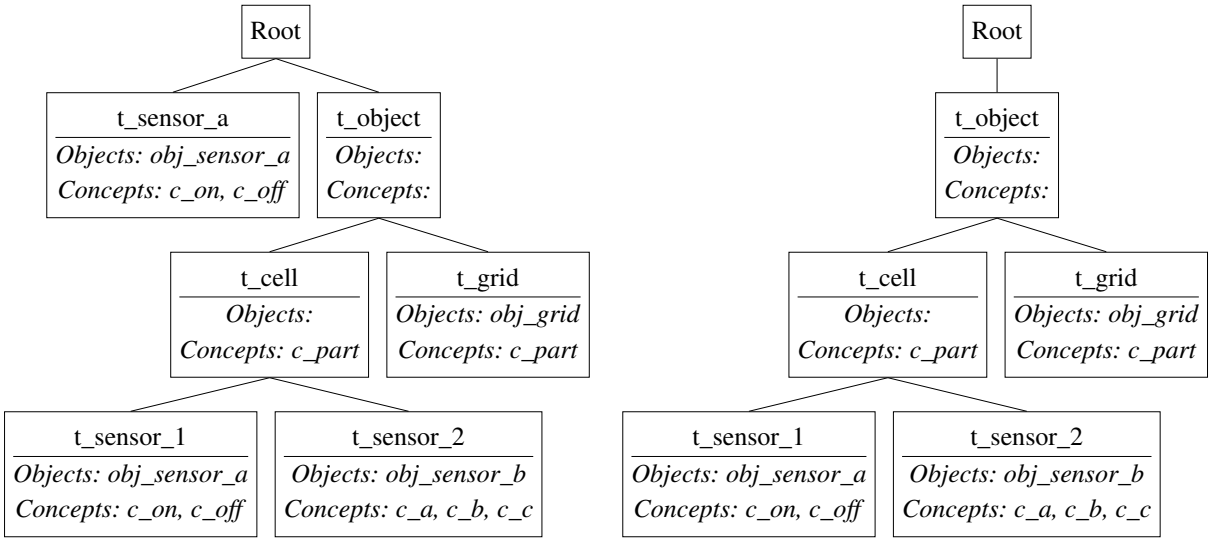


Figure 13: On the left is the tree produced when adding the current concepts to the tree built in the alternating object problem. On the right is the tree produced when taking just the current problem into account.

mentioned flag. This will allow the AAE to include these beyond the input concepts in the *application phase*. Another thing to note is that because in the previous example the AAE produced the type t_sensor_a for obj_sensor_a itself, and as here we provide a pre-built template with type t_sensor_1 for the same object our tree now has two types for the same object, which is of course inefficient space-complexity wise. To amend this an update mechanism would need to be present, which is not included as of yet. Still another solution rests us, which is a manual way of how the update mechanism could work if the AAE could also store the sensory sequences it had seen in the past in the *RawSensorySequencesLayer* proposed in section 2.1. This is because one way to approach it could be to first construct a tree using the current more complex sequence, and then to apply this tree to the previous problem. This tree produced using only the concepts learned in the current problem is shown on the right side of figure 13. Both these trees produce the resulting template and theory as can be seen below, but importantly the second tree is also able to solve the *alternating object* problem, and as it is a smaller tree we should prefer to use this second tree. In future work it is imperative to construct a mechanism for the AAE to actually produce multiple trees and compare their costs exactly for the current problematic, but we will return to this in the discussion. Now we present the *application phase* for the current problem using the second tree.

5.1.1.2 Application phase without iteration

Without iteration the system keeps to the minimal template, which as it excludes the *a priori* objects does not produce a working template. Neither can our current implementation generating a template on its own produce the right template, as it can not in an integrate yet produce objects for the template beyond the ones given in the input, let alone generate spatial structure. Iteration, as done in the next section, will thus be necessary. Before showing the working template we show here the minimal frame $\phi = (T, O, P, V)$ produced without iteration:

$$\phi = \left\{ \begin{array}{l} T = \{sensor_1, sensor_2\} \\ O = \{sensor_a : sensor_1, sensor_b : sensor_2\} \\ P = \{a(sensor_2), off(sensor_1), on(sensor_1)\} \\ V = \{SENSOR_{1\_1} : sensor_1, SENSOR_{2\_1} : sensor_2\} \end{array} \right\}$$

5.1.1.3 Application phase with iteration

The AAE produces the frame $\phi = (T, O, P, V)$, where:

$$\phi = \left\{ \begin{array}{l} T = \{cell, grid, object, sensor_1, sensor_2\} \\ O = \{sensor_a : sensor_1, sensor_b : sensor_2, grid : grid\} \\ P = \{a(sensor_2), off(sensor_1), on(sensor_1), part(cell, grid)\} \\ V = \{SENSOR_{1\_1} : sensor_1, SENSOR_{2\_1} : sensor_2, CELL_1 : cell, GRID_1 : grid\} \end{array} \right\}$$

The AAE reproduces the theory $\theta = (\phi, I, R, C)$, where:

$$R \quad = \quad \left\{ \begin{array}{l} off(SENSOR_{1\_1}) \ni on(SENSOR_{1\_1}) \\ on(SENSOR_{1\_1}) \ni off(SENSOR_{1\_1}) \end{array} \right\}$$

This is the same theory as when the prebuilt frame is provided, except that it excludes the $b(sensor_2)$ and $c(sensor_2)$ concepts as these are not found in the input. This explains why in table 5 below we see that its cost is 2 points lower. As the system has successfully reproduced the theory and template for this situation we can move on to the next example.

Process	Time in seconds	Cost
Synthesizing a theory with a prebuilt frame (baseline)	0.09	19
Synthesizing a theory by generating a frame (baseline)	x	x
Frame and theory from memory, with a single hypothesis	x	x
Frame and theory from memory, with 8 hypotheses	0.62	17

Table 5: Table showing the averaged time over 10 runs for making sense of the two objects sequence.

5.1.2 Simple sequence

The following is a simple sequence, which we will simply use to showcase the ability of the system to learn a new concept for an old object, as well as providing a good environment to showcase the production of a template from scratch. We go through several possible ways of solving the input, with the first using a prebuilt template as a baseline, then we let the AAE generate a template from scratch and finally we use the memory tree constructed in the *two objects* section, to show how concepts are added to a type. Below is the input we use in table 6:

5.1.2.1 Using prebuilt template

We provide $\phi = (T, O, P, V)$, where:

$$\phi = \left\{ \begin{array}{l} T = \{object\} \\ O = \{sensor_a : object\} \\ P = \{a(object), b(object), c(object), d(object), off(object), on(object)\} \\ V = \{OBJECT_1 : object\} \end{array} \right\}$$

time	$sensor_a$
1	a
2	b
3	c
4	d
5	a
6	b
7	c
8	d
9	a
10	b
11	c
12	?

Table 6: Input for the simple sequence problem

Our system finds the theory $\theta = (\phi, I, R, C)$, where:

$$
\begin{aligned}
I &= \left\{ \; a(sensor_a) \; \right\} \\
R &= \left\{ \begin{array}{l} a(OBJECT_1) \ni b(OBJECT_1) \\ b(OBJECT_1) \ni c(OBJECT_1) \\ c(OBJECT_1) \ni d(OBJECT_1) \\ d(OBJECT_1) \ni a(OBJECT_1) \end{array} \right\} \\
C &= \left\{ \; \forall X : object, \; a(X) \oplus b(X) \oplus c(X) \oplus d(X) \; \right\}
\end{aligned}
$$

Here we see the baseline, producing the 4 simple rules that govern the consecutive behaviour of the sequence. Now we move to the template generated by our system.

5.1.2.2 Construction phase: AAE generating a template from scratch

With this result we show that our system can produce a similar result as the baseline by generating a template from the input, on which it then will run the *apperception task* to find the same theory. The only difference of course being the naming of the type.

Using the input the AAE generates the following frame $\phi = (T, O, P, V)$, where:

$$
\phi = \left\{ \begin{array}{l} T = \{sensor_a\} \\ O = \{sensor_a : sensor_a\} \\ P = \{a(sensor_a), b(sensor_a), c(sensor_a), d(sensor_a)\} \\ V = \{SENSOR_{a\_1} : sensor_a\} \end{array} \right\}
$$

This template of course allows the apperception engine to find the same theory. Now we move on to generating a template from memory.

5.1.2.3 Application phase: generating a template from memory without iteration

With the following result we show that the AAE can add concepts to an already existing type in an AAE tree, by using the tree generated in the *two objects* section, as shown on the right side of figure 13. This

tree does not yet have the right rules, nor does it have the concept c_d and furthermore it has the letter concepts stored at a different object than the one found in the input, namely under obj_sensor_b instead of obj_sensor_a. To be able to add this to the tree it first has to be seen in a working interpretation of the sensory sequence, so first the AAE will try to *make sense* of the input with a template partially reproduced from memory and partially adapted to the input. Once it has found the best interpretation it will add this new information to its memory, effecting a nice application/construction loop.

In this case the AAE produces the frame $\phi = (T, O, P, V)$, where:

$$\phi = \left\{ \begin{array}{l} T = \{sensor_1\} \\ O = \{sensor_a : sensor_1\} \\ P = \{a(sensor_1), b(sensor_1), c(sensor_1), d(sensor_1)\} \\ V = \{SENSOR_{1\_1} : sensor_1\} \end{array} \right\}$$

Which results in the same theory, but more interestingly produces the following tree:

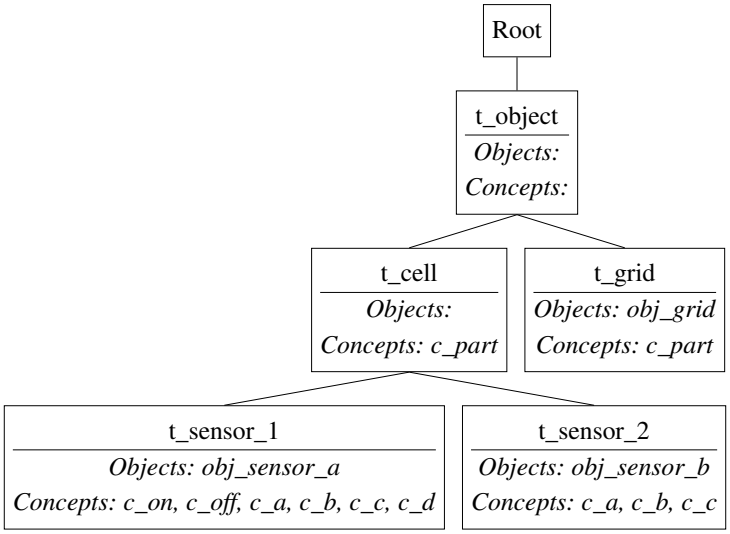


Figure 14: AAE tree resulting from the addition of new concepts to a previous tree.

Here we see that while the letter concepts where already associated with t_sensor_a the system chose to add these concepts to t_sensor_1, as in this implementation the AAE prefers to add new concepts to the same objects instead of adding new objects to old concepts/types. It would be interesting in future work to investigate which of these two options results ultimately in the better tree, and if there criteria can be devised for when which option is the better choice.

5.1.2.4 Application phase: generating a template from memory with iteration

Lastly we have produced a template/theory combination from the updated tree in figure 14, as it now stores the correct rules and concepts to fully reproduce both the theory and the template from memory. This results in the exact same template/theory combination as the previous section, but is added to results table 7 to show the time it costs to reproduce this template and theory from memory with iteration. Only with more complex sequences will we see the serious performance gains this produces. Before going to these larger examples we will discuss a case in the next section where the AAE can learn permanent facts.

Process	Time in seconds	Cost
Synthesizing a theory with a prebuilt frame (baseline)	0.05	15
Synthesizing a theory by generating a frame	0.05	15
Frame and theory from memory, with a single hypothesis	0.08	15
Frame and theory from memory, with 8 hypotheses	0.40	15

Table 7: Table showing the averaged time over 10 runs for making sense of the simple sequence.

5.1.3 Successor sequence

In this section we will show the acquisition of a permanent fact, namely the successor relation between letters. For this we can not let the AAE generate a template from scratch, as we need to indicate in the template which permanent concept it will need to give content to by learning permanent facts. It remains a challenge for future work to find a way to let the system of concepts designate certain conceptual relations as permanent facts, for example by turning the rules learned in the last section, which fulfill the same role as the successor relation here, into permanent facts. Yet as this has not been added yet this means that in this case we need to provide the template up front in the *construction phase*, the result of which we will use to update the tree generated in the last section. As input we use the following sequence shown in table 8: Given the following input:

Given the following input:

time	*sensor*
1	$letter_a$
2	$letter_b$
3	$letter_c$
4	$letter_d$
5	$letter_e$
6	$letter_f$
7	$letter_a$
8	$letter_b$
9	$letter_c$
10	?

Table 8: Input for the successor sequence problem

5.1.3.1 Construction phase using prebuilt template

We provide the following prebuilt frame $\phi = (T, O, P, V)$, where:

$$\phi = \left\{ \begin{array}{l} T = \{grid, letter, object, sensor\} \\ O = \{sensor : sensor, grid : grid, letter_a : letter, letter_b : letter, letter_c : letter, \\ letter_d : letter, letter_e : letter, letter_f : letter\} \\ P = \{letter(sensor, letter), part(sensor, grid), succ(letter, letter)\} \\ V = \{L : letter, L2 : letter, S : sensor\} \end{array} \right\}$$

It is important to note in this frame the addition of the permanent concept *succ(letter,letter)*, which in this representation is just one among the other concepts in P, but in the internal representation of the frame

has its own location and syntax, written as follows `(p_succ,Constructed,[t_letter,t_letter])`. The `Constructed` flag announces to the apperception engine that it is to construct permanent facts using this concept, the result of which we can observe in the initial conditions defined in the generated theory $\theta = (\phi, I, R, C)$:

$$
\begin{aligned}
I &= \left\{ \begin{array}{l} letter(sensor, letter_a) \\ part(sensor, grid) \\ succ(letter_f, letter_a) \\ succ(letter_a, letter_b) \\ succ(letter_b, letter_c) \\ succ(letter_c, letter_d) \\ succ(letter_d, letter_e) \\ succ(letter_e, letter_f) \end{array} \right\} \\
R &= \left\{ succ(L, L2) \wedge letter(S, L) \ni letter(S, L2) \right\} \\
C &= \left\{ \forall X : sensor,\ \exists! Y : letter,\ letter(X, Y) \right\}
\end{aligned}
$$

In the above theory we can see that the apperception engine has been able to learn several permanent facts relating to the letter objects, which we can then save in memory to be reused in future situations. This updated AAE tree is shown below in figure 15, with a closeup of the storage of permanent facts under objects in figure 16. With this result we have shown the ability of the AAE tree to learn and store permanent facts for specific objects, yet we must reiterate the caveat that we as researchers have to point out which concepts qualify as resulting in permanent facts. Future work will have to amend this to make the AAE more autonomous. Now we move on to the next example, where we will use the updated tree to reproduce the template and theory for the present sequence in the *application phase*.

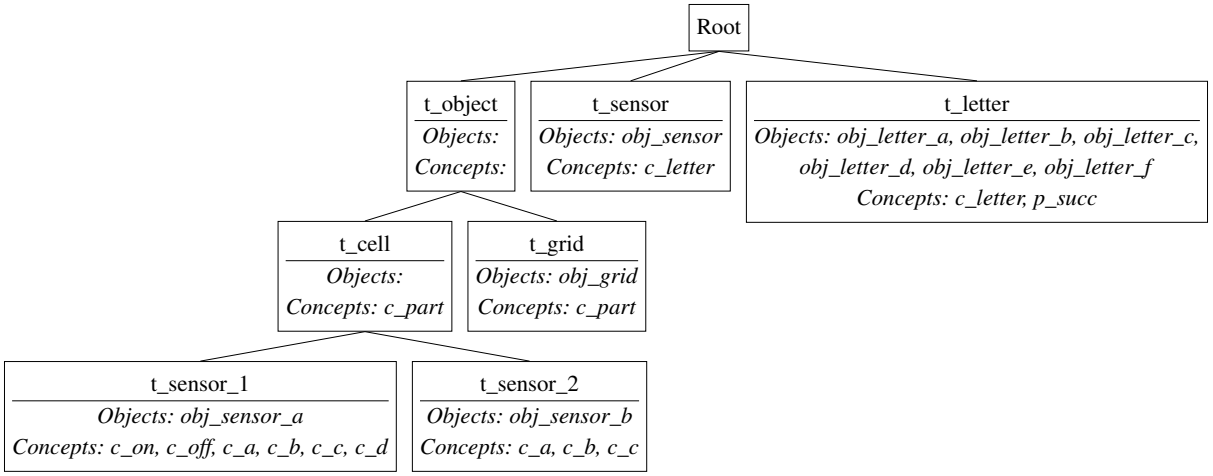


Figure 15: AAE tree after the construction phase for the successor sequence

5.1.3.2 Application phase from memory without iteration

In the application phase from memory we end up with a template/theory combination of lower cost, because in this instance the grid and cell types, objects and concepts where unnecessary, as the input consisted of a single sensor. As these were not seen in the input they were not added to the minimal template generated from memory without iteration, resulting in a smaller template, yet the resulting

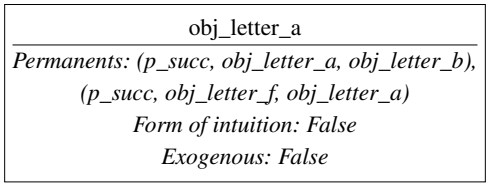


Figure 16: Example of the storage of permanent facts in the Object class, here for obj_letter_a.

theory remained the same. This theory was wholly reproduced from memory, and correct for the sequence. Next we look at the case with iteration.

Without iteration the AAE thus produces the frame $\phi = (T, O, P, V)$, where:

$$\phi = \left\{ \begin{array}{l} T = \{letter, sensor\} \\ O = \{sensor : sensor, letter_a : letter, letter_b : letter, letter_c : letter, \\ \quad letter_d : letter, letter_e : letter, letter_f : letter\} \\ P = \{letter(sensor, letter), succ(letter, letter)\} \\ V = \{SENSOR_1 : sensor, LETTER_1 : letter, LETTER_2 : letter\} \end{array} \right\}$$

5.1.3.3 Application phase from memory without iteration

The template/theory combination produced from memory *with* iteration results in the same template/theory combination as the one produced without iteration, as any added types, concepts or objects do not contribute to the theory in this example. The testing of each theory did take more time in this case, which resulted in a serious decrease in performance with iteration, as can be seen in table 9. While we have shown that the AAE can now store and reproduce the concepts and theories it produces to *make sense* of sensory sequences, this has with the simple examples only resulted in equal performance, or performance loss with respect to time-complexity. This will change drastically when we look at more complex results, thus in the next section we will go on to present the functionality of the AAE with regards to the previously discussed Sokoban environment. Finally it is important to note that the AAE tree built up over these 4 examples is now still able to be *applied* to the previous problems without a change in accuracy or significant reduction in speed, the AAE can truly act like a memory.

Process	**Time in seconds**	**Cost**
Synthesizing a theory with a prebuilt frame (baseline)	0.16	18
Synthesizing a theory by generating a frame	x	x
Frame and theory from memory, with a single hypothesis	0.20	9
Frame and theory from memory, with 8 hypotheses	1.40	9

Table 9: Table showing the averaged time over 10 runs for making sense of the successor sequence.

5.2 Sokoban: speed-up and similar situations

In this section we will present the results the AAE achieves in the context of the Sokoban environment, which allows us to show both the tremendous and expected speed-up from being able to remember and reproduce templates and theories across sensory situations, as well as showing its behaviour when presented with similar yet different sequences. Before expanding on the results we will specify the current Sokoban problem, which is in essence the same problem as the Symbolic Sokoban problem discussed in section 3.2, but then without the explicit shape predicates, as the essence/appearance distinction does not

play a major role in the computational implementation of the AAE and would only introduce unnecessary complexity. In this version of the problem we follow Evans' way to distinguish the *Player* and *Block* pieces by using two different *in* predicates, namely in_1 and in_2, which are applied to respectively the *Player* and the *Block* type.[120] Also changed is that *Player* and *Block* now simply have their own types, instead of both being of the *object* type and differentiated by their *shape* predicate. We will give an example of this new problem set-up in the following section, where we discuss the first result.

5.2.1 Sokoban general

Here we represent the sensory sequence in figure 17, with a translation to the ASP input given in figure 18, where one can note the differentiation of *Player* and *Block* by the in_1 and in_2 predicates, instead of by the shape predicates as done in the Symbolic Sokoban setup in section 3.2. We use here a general sequence that exhibits the behaviour for each piece, thus showing the *Player* as both moving and pushing the *Block*. Yet this starting sequence can not be too small, as then the apperception engine starts to learn malformed rules that take the spatial positioning into account. To prevent this the sequence continues another 13 steps beyond the 5 steps shown here, which move the *Player* back and forth and lets it push the *Block* from multiple sides.[121] We do not show the complete sequence here as beyond step 5 it is mostly redundant, but refer the reader to the `data/sokoban/predict_e8_17.lp` file in the code if the whole sequence is desired. Now we move onto the construction phase using a prebuilt template.

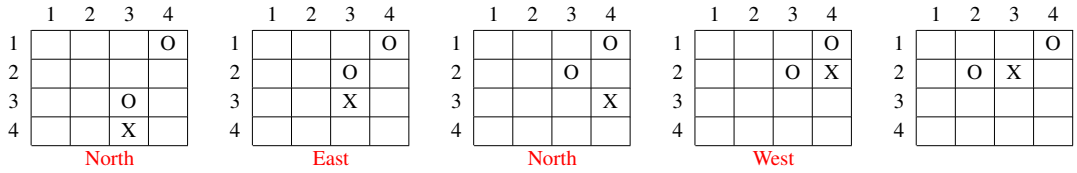


Figure 17: The general Sokoban sequence with the player actions printed below the grid for each timestep.

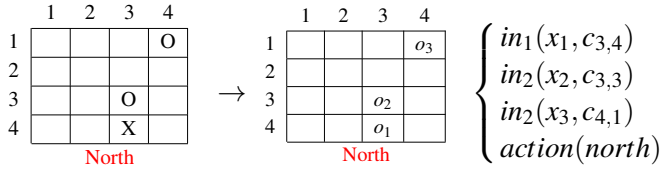


Figure 18: An example of the symbolic state translated from the Sokoban board state, which is taken as input for the apperception task.

[120] To note is that in the internal representation I have also changed the names of the types for the Sokoban problem from Evans' *t_1* and *t_2* to *t_player* and *t_block* for readability. This may cause issues when trying the AAE on new Sokoban sequences, so please make sure these types are changed when trying out the AAE with new instances, especially with the constraints given in the input.

[121] This is an interesting showcase of the systematic investigation principle discussed earlier, as we have to systematically built up the right input with *our* faculty of Reason to let the apperception engine generate the proper concepts. An example of such a malformed rule generated by the apperception engine when given just the 5 steps below is this: $is_w all(CELL_1) \wedge p1(BLOCK_1) \rightarrow in_2(BLOCK_1, CELL_1)$, a rule that assigns the predicate $p1$ to a *Block*, which is supposed to indicate that it has been pushed by the *Player*, simply when the *Block* is in a specific spatial location next to one of the gridwalls. This show why it is imperative in future work to devise a way for the apperception engine to systematically generate sensory sequences (thus move through a world) that allow it to learn properly general concepts.

5.2.1.1 Construction phase using prebuilt template

For the *construction phase* of the small Sokoban sequence we feed the system a prebuilt frame $\phi = (T, O, P, V)$, where:

$$\phi = \left\{ \begin{array}{l} T = \{block, cell, player\} \\ O = \{x1 : player, x2 : block, x3 : block, c_{x,y} : cell \mid (x,y) \in \{1,2,3,4\} \times \{1,2,3,4\}\} \\ P = \{in_1(player, cell), in_2(block, cell), p1(block), p2(block), p3(block), p4(block), \\ \quad south(player), west(player), east(player), noop(player), north(player)\} \\ V = \{PLAYER_1 : player, CELL_1 : cell, BLOCK_1 : block, CELL_2 : cell\} \end{array} \right\}$$

This template includes the predicates *p1, p2, p3, p4* discussed in section 3.2 and the spatial information consisting of a 4 by 4 grid of cells needed as a spatial framework for the problem. Using this template for the *apperception task* the apperception engine *makes sense* of the sequence using the theory $\theta = (\phi, I, R, C)$, where:

$$I = \left\{ \begin{array}{l} in_1(x1, cell_{3,4}) \\ in_2(x2, cell_{4,1}) \\ in_2(x3, cell_{3,3}) \end{array} \right\}$$

$$R = \left\{ \begin{array}{l} below(CELL_1, CELL_2) \wedge in_1(PLAYER_1, CELL_2) \wedge in_2(BLOCK_1, CELL_1) \rightarrow p1(BLOCK_1) \\ below(CELL_1, CELL_2) \wedge in_1(PLAYER_1, CELL_1) \wedge in_2(BLOCK_1, CELL_2) \rightarrow p1(BLOCK_1) \\ right(CELL_1, CELL_2) \wedge east(PLAYER_1) \wedge in_1(PLAYER_1, CELL_1) \wedge in_2(BLOCK_1, CELL_2) \rightarrow p4(BLOCK_1) \\ right(CELL_2, CELL_1) \wedge in_1(PLAYER_1, CELL_1) \wedge in_2(BLOCK_1, CELL_2) \rightarrow p4(BLOCK_1) \\ below(CELL_1, CELL_2) \wedge p1(BLOCK_1) \wedge south(PLAYER_1) \wedge in_2(BLOCK_1, CELL_1) \ni in_2(BLOCK_1, CELL_2) \\ right(CELL_1, CELL_2) \wedge east(PLAYER_1) \wedge p4(BLOCK_1) \wedge in_2(BLOCK_1, CELL_1) \ni in_2(BLOCK_1, CELL_2) \\ right(CELL_2, CELL_1) \wedge p4(BLOCK_1) \wedge west(PLAYER_1) \wedge in_2(BLOCK_1, CELL_1) \ni in_2(BLOCK_1, CELL_2) \\ below(CELL_1, CELL_2) \wedge south(PLAYER_1) \wedge in_1(PLAYER_1, CELL_1) \ni in_1(PLAYER_1, CELL_2) \\ below(CELL_1, CELL_2) \wedge north(PLAYER_1) \wedge in_1(PLAYER_1, CELL_2) \ni in_1(PLAYER_1, CELL_1) \\ right(CELL_2, CELL_1) \wedge east(PLAYER_1) \wedge in_1(PLAYER_1, CELL_2) \ni in_1(PLAYER_1, CELL_1) \\ right(CELL_1, CELL_2) \wedge west(PLAYER_1) \wedge in_1(PLAYER_1, CELL_2) \ni in_1(PLAYER_1, CELL_1) \\ below(CELL_1, CELL_2) \wedge north(PLAYER_1) \wedge p1(BLOCK_1) \wedge in_2(BLOCK_1, CELL_2) \ni in_2(BLOCK_1, CELL_1) \end{array} \right\}$$

$$C = \left\{ \begin{array}{l} \forall X : player,\ \exists! Y : cell,\ in_1(X, Y) \\ \forall X : block,\ \exists! Y : cell,\ in_2(X, Y) \end{array} \right\}$$

Here the apperception engine has produced concepts which are able to fully explain each Sokoban instance,[122] but while previously we could not reproduce these general concepts as there was no way to store them, forcing the apperception engine to generate the same or even deficient concepts in a new

[122] Evans notes this on page 126 in [Evans, 2020]: "While most of the trajectories do not contain enough information for the engine to extract a correct theory, three of them are able to achieve 100% accuracy on the held-out portion of the trajectory. Of course, getting complete accuracy on the held-out portion of a single trajectory is necessary, but not sufficient, to confirm that the induced theory is actually correct on all possible Sokoban configurations. We checked each of the three accurate induced theories, and verified by inspection that one of the three theories was correct on all possible Sokoban maps, no matter how large, and no matter how many objects." Here we see also Evans noting the importance of the right input, which when found the AAE can now use to easily explain the other situations. Taking in mind the **Theory-Choice Problem** we notice the emergence of a better criterion for the hypotheses the apperception engine produces than the simple cost-calculation, which is reproducibility in similar but different situations. While the concepts learned in the most general sequence can be applied again in the smaller situations, the reverse is not possible. With the full Memory stack discussed in 2.1, which would allow the AAE to save both previous hypotheses, concepts and sequences we could reassess the generality of certain hypotheses and concepts based on their

but similar situation, now the AAE tree allows the apperception engine actually learn these concepts. We present the newly constructed tree below in figure 19. Here we have presented the tree with the previously learned concepts folded away, except for t_cell, as the complete tree is too large to display informatively.[123] Yet it is important to note that the AAE still functions properly with a tree of this size.[124] This tree we can now use to solve a myriad of other Sokoban problems, but we will start by reapplying

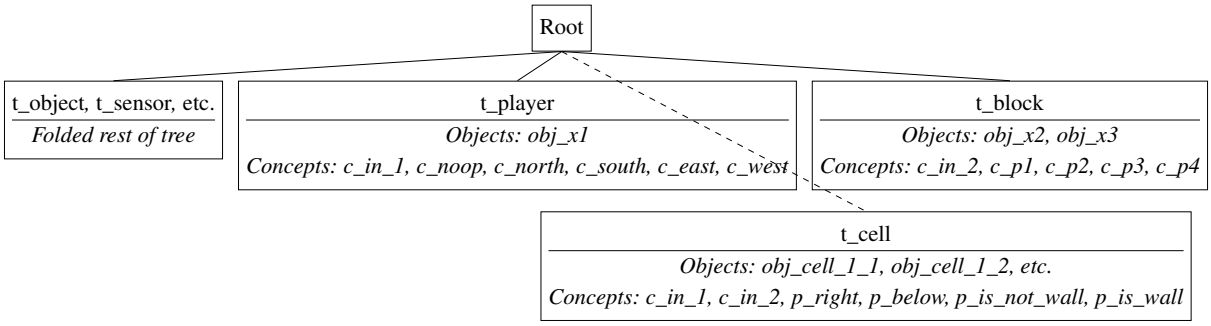


Figure 19: AAE tree built using the concepts generated for the small Sokoban problem

the concepts to the general sequence.

5.2.1.2 Application phase

In the *application phase* the AAE reproduces the same template and the same theory, but iteration over hypotheses is needed as the minimal template does not include the predicates *p1, p2, p3 ,p4*, for these are not seen in the input, thus *speculative* concepts. It is also important to note that as we discussed earlier the AAE does not have a sophisticated way of dealing with spatial structures yet, so to make the Sokoban spatial structure work we simply import it from the input into the template. To show the gain in speed that is in principle possible, with a potentially more sophisticated iteration, we have included a row in the table of results using the maximal template, thus including all speculative concepts, supertypes and a priori objects. This can be seen as the lower bound for the time-complexity of the AAE in the Sokoban case, with the upper bound being given by the full iteration. We present the results below in table 10

Process	**Time**	**Cost**
Synthesizing a theory with a prebuilt frame (baseline)	60 minutes	83
Synthesizing a theory by generating a frame	x	x
Frame and theory from memory, with a single hypothesis	x	x
Frame and theory from memory, with a single hypothesis (optimized for Sokoban)	25 seconds	83
Frame and theory from memory, with 8 hypotheses	minutes	83

Table 10: Table showing the averaged time over 10 runs for making sense of the general Sokoban sequence.

applicability in a new situation, and when presented with a properly general sequence that allows the apperception engine to produce the properly general concepts it could reinterpret the past sequences in light of this better theory. We will return to this in the next section.

[123] I have added an representation of the complete tree in the on the github page: https://github.com/afalbrecht/apperception

[124] Still we expect that when it grows larger it will start to present defective behaviour without update and cleaning mechanisms. We will present such a problem in section 5.2.2.

From the table we can see that original apperception engine using a prebuilt template can find a working theory in 60 minutes,[125] while when this theory has been learned once the computation-time drops to at most 6 minutes. This figure of 6 minutes is even an upper-bound, namely the time it takes to check all 8 hypotheses, as the first working hypothesis of the 8 is found after 1 minute.[126] The absolute lower-bound for the reproduction of the template and theory is 25 seconds, taking the maximal template as optimized for Sokoban. This speed-up is to be expected, since the complexity of finding an optimal (lowest-cost) solution for a sensory sequence *given* a template is in Σ_2^P,[127], while checking such a solution is polynomial, with the traversal of the tree to produce hypotheses being trivial compared to the checking of the solution. Reproducing the theory with the template thus allows us to seriously cut down on complexity, as is to be expected.[128] Note that Σ_2^P is the complexity class *given* a template, which when we take into account the diagonal iteration over the templates would only represent a lower-bound, as each iteration of the template would need to perform a Σ_2^P task. This is not enough to bump it into a higher complexity class, but does indicate that the result of 1 hour for a solution[129] as the baseline for the general Sokoban problem is only possible when *given* the proper template up front, where a proper diagonal *search* through the templates would increase the time it takes to find a solution from hours to days, which is why this was not attempted in this thesis. Yet it does show that for the AAE to be able to not only reproduce the theory but also the template allows for very significant speedups, lowering computation time from hours or days to mere seconds or minutes after a solution has been found for the first time.[130] With this result we then state that the AAE is able to propose a significant first step for a solution to the *Search problem*, which is that by simply storing and reproducing templates and theories we can in the case of the reproduction of theories cut down the complexity class from Σ_2^P to P,[131] and in the case of the templates cut down the iteration to at most 8 hypotheses in the current implementation. Future work could improve the generation of hypotheses to provide a better solution to the template iteration, yet this proof-of-concept has already been able to show a significant improvement to the **Search Problem**. In the next section we will show more results to back up our speedup claims, as well as showing the ability of the AAE to apply previously learned concepts to different but similar situations.

5.2.2 Application to different but similar sequences

In this section we apply the AAE tree to different Sokoban sequences, which are similar but distinguished from the general Sokoban sequence by slight alterations. Besides the baseline time needed to solve the problems when provided just the template we have only provided the lower-bound of the AAE tree, thus

[125] Which is a lower-bound, because this is when the first working theory is generated, which had a higher cost than the one presented here. The theory we have shown above was constructed after 4 hours of computation time, though with a minimal drop in cost, which is why we choose to present the 60 minutes mark as the computation-time.

[126] Better iteration would thus be very useful in this case.

[127] [Evans, 2020, p.64], [Brewka et al., 2003]

[128] We now also see why checking 8 hypotheses lasts 6 minutes instead of 3.5 minutes, which we would expect when we simply multiply the time to check a correct solution, 25 seconds, by 8. The slowdown is of course caused by the fact that checking a correct solution takes less time than checking a wrong solution, becoming even worse when checking a partial solution for which the apperception engine can generate new concepts to create a correct solution. In future work it would thus be imperative to add time-limits to the hypothesis checking, and only after each hypothesis has been checked to allow the apperception to run full-blown *apperception tasks* on these hypotheses.

[129] Or 4 hours for a more optimal solution

[130] Or potentially even when concepts learned for different situations already allow the AAE to produce a correct template for the new situation, implying that a general understanding of the world explicitly codified in a system of concepts can allow us to *make sense* of new situations much faster.

[131] If we of course add some time-limits on checking hypothetical theories, to prevent the AAE from slipping into a Σ_2^P problem before properly completing all P tasks.

using a single hypothesis with optimized setting[132] for clarities sake, as the iteration over hypotheses can now start to perform the larger apperception task, as a proper time-limit has not been implemented yet. The results are shown in table 11 below, after which we discuss some peculiarities of the different inputs.

Process	**Baseline time**	**AAE time**
1. Small General Sokoban	30 minutes	14 seconds
2. Horizontal Line Sokoban with 1 Block	30 minutes	14 seconds
3. Sokoban with 3 Blocks	80 minutes	45 seconds
4. Smallest General Sokoban	15 minutes	6 seconds
5. Sokoban with 1 Block	20 minutes	9 seconds
6. Sokoban sequence no Blocks and 1 Player	10 minutes	10 seconds
7. Vertical Line Sokoban with 2 Blocks	15 minutes	6 seconds

Table 11: Table showing the averaged time over 10 runs for making sense of different Sokoban sequences.

1. The first input to be considered is the small variant of the general Sokoban sequence, thus taking only the first 5 time-steps, which it can obviously solve in less time than the general Sokoban sequence using the AAE tree. Important to note is that as we discussed earlier the solution found in the baseline case takes a shorter time to produce, yet is not general enough to be able to explain the larger general Sokoban sequence. This still holds true if we let the apperception engine continue longer in its search to find a more optimal solution, as of course the more optimized the solution becomes the more it is geared to the specific case. This is why a properly general input is important for constructing the AAE tree.
2. The second input is long horizantal line on which the Player pushes a single Block to the right, which is interesting because it has a very different spatial structure. Yet as discussed earlier our current implementation simply reads in this spatial structure, making it only significant for the rules that are included. This it still handles correctly.
3. This third sequence adds a Block to the sequence, which the AAE tree can add on the fly under the correct type t_block, as it has enough information from the concept c_in_1 to decide on which type it can belong to. It is not done now, but in principle it could also iterate over this uncertainty by both trying a known type and a newly created type for this object, yet here the minimal case is already correct. As it includes more objects and a larger field, this sequence of course takes longer to solve, both in the baseline case and the theory generated from the AAE tree.
4. This input is an even smaller case of the general Sokoban problem to show the decrease in time spent solving and checking for smaller sequences.
5. This Sokoban sequence simply includes the Player moving in every direction, without any Blocks present. To solve this the AAE tree needs to reproduce only the 4 rules defining the movement of the Player, which it is shown to be able to do.[133]

[132] Kept the same for every sequence

[133] This is an interesting case not for any specific conceptual reason, but because it lays bare some odd behaviour in the ASP program. Namely when this line: `rule_body(r5,isa(p_is_not_wall,var_cell_1)).` is included in one of the rules, it refuses to find a solution, even if on the face of it the line is an inconspicuous check for if the cell moved to is not a wall. Curiously it does work when the index, thus here `r5`, is changed to 5 instead of the expected 4. This is note because an extra rule is produced, and the precise reason for the influence of the in principle arbitrary indexes I have not been able to deduce. As there is no principled problem to be found here I have solved it by hardcoding the AAE to never include a line containing `p_is_not_wall`, as this does not change the results of any other problem, except simply fixing this one. It is noted here only for further potential improvement to the computational environment, as it has no conceptual upshot.

6. This input consisting of a vertical line with 2 Blocks that the Player both moves is on the face of it inconspicuous, but when the AAE tree produces a theory for it it acts like a monkey wrench to the functioning of the tree, by adding a contradicting part to the following rule:
$right(CELL_2, CELL_1) \wedge east(PLAYER_1) \wedge in_1(PLAYER_1, CELL_2) \ni in_1(PLAYER_1, CELL_1)$,
which has now become:
$right(CELL_1, CELL_2) \wedge right(CELL_2, CELL_1) \wedge east(PLAYER_1) \wedge in_1(PLAYER_1, CELL_2) \ni in_1(PLAYER_1, CELL_1)$.
The new rule now paradoxically expects $cell_1$ to be to the right of $cell_2$, as well as vice versa, before it can move move the Player to $cell_1$! I have not yet been able to figure out why exactly it does this, but I can attest to the fact that once it has added this rule to the AAE tree it becomes unable to solve any of the Sokoban problems, including this one. In future work this would be a good place to start for investigating the potential generation of conflicting rules, possibly caused by the communication between the system of concepts and the base apperception engine. For now we simply note the problem and move on to the conclusion.

6 Conclusion

In this thesis we have sought to make concrete Kant's notion of the Architectonic of Reason, both theoretically and computationally by implementing a proof-of-concept into the Apperception Engine, creating the Architectonic Apperception Engine. This was successful, as the AAE can now learn concepts and theories generated in an *apperception task* by constructing a system of concepts, and reapply these concepts in similar and different situations, decreasing computation time for *making sense* of a similar sensory sequence by at least 10 to 100 times, depending on the setup of the AAE.

Now we will report more precisely the different advancements made through the implementation of memory into the apperception engine, which we deemed a necessary addition as we argued that for the AE memory in the form of a conceptual system is needed to be able to learn and reuse concepts in different situations, to be able to provide a criterion for the truth of its theories and the generality of its concepts and to decrease the time-complexity of the its costly computations. These three problems were termed respectively the **Memory Problem**, the **Theory-Choice Problem** and the **Search Problem**, and we will list the results obtained by our implementation for each of these problems.

6.1 Memory Problem

In the case of the **Memory Problem** we have successfully implemented a version of the *System of Concepts Layer* described in section 2.1 that can: 1) store concepts and theories generated in *apperception tasks*, 2) reapply these to the same situation that they were learned in, 3) apply these to different but similar situations, 4) still do this when the system of concepts contains a myriad of other concepts learned in situations of differing complexity.
The other layers have not been implemented, thus the AAE can not yet on its own reassess its past interpretations in the light of new knowledge, but as the *System of Concepts Layer* is the most essential layer for the functioning of memory we can say that the AE has now acquired a basic form of memory, allowing it to *learn* the concepts it generates, store these in *system of concepts* and *reapply* these to new situations. This shows us both the theoretical validity of Kant's notion of a system of concepts as memory, and a way to construct an *explicit memory* in the context of Machine Learning, over against the Neural paradigm of *implicit memory*. This allows for human inspectability of the concepts the Machine Learner has acquired, improving on the inspectability of the concepts of the base apperception engine in situations where we as researchers are not aware of the underlying rules ourselves, as the rules and concepts applied

to the new situation can now be put into context with concepts learned in previous, known contexts. It also allows us to start attacking the **Theory-Choice Problem**, described in the next section.

6.2 Theory-Choice Problem

In the case of the **Theory-Choice Problem** we have been unable to present a full-fledged implementation of the apperception engine with a more sophisticated way to decide on the truth of theories, as for that it is necessary to implement a method of *systematic investigation*, for which the AAE would need agential features which it does not presently have. We have shown advancements in the solution to the **Criterion Problem**, as we now have a way to check the generality of the concepts in a theory without human inspection. The example for this is the problem reported with the theory learned for the Small General Sokoban problem described in section 5.2.1, which was able to *make sense* of the small problem in question, but not of the General Sokoban problem, while we could test that reverse was not true, as the theory produced for the General Sokoban problem could make sense of all the others. Evans already discovered this by human inspection, but with the addition of memory we could now let the AAE test it, which would thus work as a *criterion* for the generality of a theory if in future work a method of *systematic investigation* is added.
With respect to the **Relevance Problem** we can report no major advancements, for as we discussed earlier this relies even more on the inclusion of agential features in the AAE.

6.3 Search Problem

Lastly in the case of the **Search Problem** we can conclude that the AAE has solved the problem of the costliness of the *apperception task* to a large extent, as when it is able to reuse a theory or parts of a theory learned previously the complexity of the *apperception task* drops from Σ_2^P to P, resulting in a 10 to 100 times speedup depending on the setup of the AAE. As regards to improving the template iteration over the diagonalization we can report also report an improvement, as a complex template like the one for Sokoban does not need to be provided as a prebuilt template anymore, but can be reproduced by the AAE once it has stored the right concepts in memory. Using diagonalization this could also be done in principle, but in practice this takes days, making it an unviable option.

7 Discussion and future work

Here we will discuss any limitations of the implementations and give suggestions for future work.

7.1 Systematic investigation

The first and most important limitation and thus possibility for future work is a method of systematic investigation, enabled by a notion of agentiality. To implement this one would first have to add the other layers in the **Memory Problem**, to be able to reassess and compare previous situations, as well as a way to store hypothetical trees in memory, which would allow the AAE to keep different solutions to a specific sensory sequence in its memory, to then check which of these is best by trying to apply the different hypothetical trees to new situations. Preferably it would then also be able to systematically traverse a world, in a manner conducive to learning relevant and general concepts. Systematic investigation implies that the AAE is able to notice that staring at a blank wall for hours is not fruitful to to developing its

system of concepts,[134] and to be able to move to situations more conducive to furthering its understanding. This is of course a far way off, but the first steps could be made using the Sokoban case, where there definite distinctions in the generality of the sensory sequences, allowing it to systematically check which of the sensory sequences can best be used to explain the others by iteration through the different situations.

7.2 Update and maintenance mechanisms

As noted in section 5.2.2 regarding the Sokoban problem that acts as a monkeywrench to the system of concepts by learning a paradoxical rule, it is necessary once the tree grows larger to be able to update, maintain and change the tree. If for example a large number of previous problems do not work anymore after the addition of a new concept the AAE should be able to have a mechanism to prune or change specific nodes. If we were to implement a notion of systematic investigation into the AAE an update and maintenance mechanism is an absolute precondition for the proper dynamic behaviour of the system.

7.3 Forms of Intuition: Space and Time

A serious limitation of the current approach is its engagement with the forms of intuition, which can not be learned like normal empirical concepts, but are relevant in almost every situation. A proper approach would combine the implementation of the Imagination in the FAE,[135] which can procedurally build up space and time, with the AAE, which can then learn and apply concepts in the spatially and temporally specific framework of the FAE, without needing to store any of the spatial concepts. Though certain objects would probably imply certain spatial structures, which the AAE could then inject as an 'expectation' in the mechanisms of the Imagination captured by the FAE.

7.4 Cumulative Learning

One of the advantages of Memory discussed earlier was the possibility for cumulative learning, thus learning permanent facts such as the successor relation between letters without which a Seek-Whence sequence would not be able to be interpreted, as it is impossible to learn the successor relation between the different letters from most Seek-Whence problems. This we have tested, and we have succeeded to learn the successor facts in one sequence, to then inject these via an `interpret_mem.lp` file to the Seek-Whence problem, but difficulties relating to the specific implementation of the Seek-Whence templates prevented me from making this work using only the tree. Yet I am sure that with one more week this would be possible, thus the AAE is not far off from cumulative learning, as in principle nothing should prevent the current system of concepts from doing this, beyond the specific problems encountered with the Seek-Whence templates.

7.5 Raw Sensory Sequences

I have actually tested if it is possible to save the template and theory for the Raw Sokoban Sequences[136] manually and reapply these, and this is indeed possible, driving the computation time down from 10 hours to 20 seconds. Yet a serious obstacle prevents the current AAE from doing this using the tree, which is that it reads in the symbolic sensory sequence input to determine the shape of the template and theory according to the concepts and objects present in the input, but in the raw sensory sequence these are of

[134] Although it might be useful for it to attain nirvana, if we are to believe the account of Bodhidharma, the disseminator of Buddhism into China, who gazed at a wall in a cave for nine years and supposedly cut off his eyelids in anger when he accidentally dozed off in the seventh year.

[135] The version of the apperception engine created in [Soeteman, 2022]

[136] As described by Evans starting on page 119 of [Evans, 2020]

course not present in the input, but given as output by the Binary Neural Network, which is implemented in ASP. A serious restructuring of the program loop would thus be needed to determine which types can be gathered from the tree, as instead of directly reading the input the system of concepts would have to communicate directly with the Binary Neural Network and inspect its outputs before handing a template to the *apperception task*. I have not yet been able to investigate the feasibility of this implementation, but do think it should in principle be possible to implement in not too much time.

References

Theodora Achourioti and Michiel Van Lambalgen. A formalization of kant's transcendental logic. *The Review of Symbolic Logic*, 4(2):254–289, 2011.

Theodora Achourioti, Michiel van Lambalgen, et al. Kant's logic revisited. *IfCoLog Journal of Logics and Their Applications*, 4:845–865, 2017.

R Lanier Anderson. *The poverty of conceptual truth: Kant's analytic/synthetic distinction and the limits of metaphysics*. Oxford University Press, USA, 2015.

Gerhard Brewka, Ilkka Niemelä, and Miroslaw Truszczynski. Answer set optimization. In *IJCAI*, volume 3, pages 867–872. Citeseer, 2003.

Richard Evans. *Kant's cognitive architecture*. PhD thesis, Imperial College London, 2020.

Richard Evans. Self-legislating machines: What can kant teach us about original intentionality? *Kant-Studien*, 113(3):555–576, 2022. doi: doi:10.1515/kant-2022-2030. URL `https://doi.org/10.1515/kant-2022-2030`.

Alfredo Ferrarin. *The powers of pure reason*. University of Chicago Press, 2015.

Martin Gebser, Roland Kaminski, Benjamin Kaufmann, and Torsten Schaub. Clingo= asp+ control: Preliminary report. *arXiv preprint arXiv:1405.3694*, 2014.

Paul Guyer et al. *Kant's system of nature and freedom: Selected essays*. Oxford University Press, 2005.

Douglas R Hofstadter, Marsha J Meredith, and Gary A Clossman. Seek-whence: A project in pattern understanding. Technical report, CRCC Report, 1982.

Immanuel Kant. On a recently prominent tone of superiority in philosophy. *Theoretical Philosophy after 1781*, pages 428–445, 1781.

Immanuel Kant. *Opus postumum*. Cambridge University Press, 1995.

Immanuel Kant. *Critique of pure reason*. Cambridge University Press, 1998.

Immanuel Kant. *Critique of practical reason*. Hackett Publishing, 2002.

Immanuel Kant. *Lectures on logic*. Cambridge University Press, 2004a.

Immanuel Kant. *Metaphysical foundations of natural science*. Cambridge University Press Cambridge, 2004b.

Saul A Kripke. *Wittgenstein on rules and private language: An elementary exposition*. Harvard University Press, 1982.

Béatrice Longuenesse. *Kant and the capacity to judge: sensibility and discursivity in the transcendental analytic of the Critique of pure reason*. Princeton University Press, 2020.

Plato. *Meno*. Liberal Arts Press New York, 1949.

Arie Soeteman. Artificial understanding. Master's thesis, Universiteit van Amsterdam, the Netherlands, 2022.

Peter Strawson. *The bounds of sense: An essay on Kant's critique of pure reason*. Routledge, 1959.

Ludwig Wittgenstein. *Philosophical investigations*. John Wiley & Sons, 2010.

Lea Ypi. *The architectonic of reason: purposiveness and systematic unity in Kant's critique of pure reason*. Oxford University Press, 2022.