Cumulative Learning with Causal-Relational Models

Kristinn R. Thórisson[1,3] and Arthur Talbot[1,2(✉)]

[1] Center for Analysis and Design of Intelligent Agents, Reykjavik University, Reykjavik, Iceland
[2] Ecole normale supérieure Paris-Saclay, Cachan, France
arthur.talbot@ens-paris-saclay.fr
[3] Icelandic Institute for Intelligent Machines, Reykjavik, Iceland
thorisson@ru.is

Abstract. In the quest for artificial general intelligence (AGI), questions remain about what kinds of representations are needed for the kind of flexibility called for by complex environments like the physical world. A capacity for *continued* learning of *many domains* has yet to be realized, and proposals for how to achieve general performance improvement through continuous *cumulative learning*—while seemingly a necessary feature of any AGI—remain scarce.

In this paper we describe a cumulative learning mechanism that produces causal-relational models of its environment, to predict events and achieve goals. We show how such models, coupled with an appropriate *modeling process*, result in knowledge whose accuracy increases over time and can run continuously throughout the lifetime of an agent. The methods have been implemented, demonstrating learning of complex tasks and situated grammatically-correct natural language by observation. Here we focus on key theoretical principles of the modeling method and explain how effective cumulative learning is achieved.

Keywords: Artificial intelligence · Artificial general intelligence
Cumulative learning · Cumulative modeling · Causal relations
Models · Knowledge representation · Autonomous learning
Task-environment · Knowledge acquisition

1 Introduction

We see the existence of intelligence in nature as a practical solution for *limited time and resources* [17], and our efforts target practically viable methods for building artificial general intelligence (AGI) systems. While in this paper the primary focus is on theoretical aspects of *cumulative modeling*, which itself is a subset of cumulative learning, the larger context for this work is AGI systems that can handle the complexities of the physical world.

This work was funded by Reykjavik U. and IIIM, Iceland.

M. Iklé et al. (Eds.): AGI 2018, LNAI 10999, pp. 227–237, 2018.
https://doi.org/10.1007/978-3-319-97676-1_22

An environment E into which a goal-oriented agent is introduced can be seen to consist of a (potentially large) set of variables $\mathcal{V} = \{v_1, v_2, \ldots, v_{\|V\|}\}$ that represent all the things in the world that may hold a particular value and change over time, along with relations $\Re$ (causal, meteorological, etc.) between (some) of these variables, and dynamics functions $\mathcal{F}$, that together determine how those changes happen. Subsets $\{e_1 \ldots e_n \subset E = \langle \mathcal{V}, \mathcal{F}, \Re \rangle\}$ can be identified,[1] where relations, dynamic functions, values and value ranges, are of a special kind or of particular interest, representing what we collectively call *domains*, $\mathcal{D} \subset E$.[2]

As proven by Conant and Ashby's Good Regulator Theorem [1], to be an efficient survivor in a complex world, a learning mind must *necessarily* proceed by *modeling* its task environment. This means that any AGI system, being a learning controller of the most capable kind, will need a significant amount of models to operate effectively in the physical world. A *model set* $\mathcal{M}_D$ of $\mathcal{D}$ contains models of (parts of) the environment – information structures that together describe $\mathcal{D}$, to some level of accuracy.

A *cumulative modeler CM* in our conceptualization is a controller[3] that, guided by one or more top-level (internalized) goals G_{top}, implements a process whereby regularities are recursively extracted from E to construct models $\mathcal{M}_D$ of it [14] for the purposes of (a) making predictions about $\mathcal{D}$, and (b) achieving goals with respect to $\mathcal{D}$. In our approach models are explicit, and this is done via *forward* and *backward* chaining, respectively.

The kind of models we are talking about the agent creating are *bi-directional* in that a single model serves both purposes of prediction and goal achievement, and whenever a model is triggered (considered relevant for a situation) its usefulness for both purposes may be tested and evaluated. The ability of models to be used for predicting events from particular conditions, and planning active intervention, is a key feature of significant importance for the nature of the knowledge thus accumulated, as discussed below. In our approach, models can refer to other models to form hierarchies, so that compound phenomena can be represented, and equally importantly, so that the system can model itself (to implement *reflection* [14]). Knowledge is non-axiomatic and defeasible [13], so any old knowledge—even that which has repeatedly been shown to be useful—may be defeated by new knowledge that is more useful (better predictions and/or better goal achievement) and consistent with other models. New knowledge is automatically reconciled with old knowledge, and learning tends to be sped up due to prior knowledge (transfer learning [7,8]), without catastrophic interference/forgetting [2,4,6].

[1] We mean *any* sub-division of E, $e_n \subset E$, including sub-structures, component processes, whole-part relations, causal relations, etc.

[2] In any complex environment such as the physical world there will be innumerable ways of domain sub-divisions. The range of domains created from human-centric perspectives (e.g. transportation, electronics, home, commerce, clothing, etc.) demonstrate the utility of such sub-division.

[3] A controller is the process that dynamically couples knowledge and goals to obtain actions (or inaction) in an environment [14].

A *good* cumulative modeler is one which does not build its models haphazardly or randomly but in a way that achieves goals and predictions efficiently and effectively. Elsewhere we have argued that to do so the models must by necessity capture causal relations[4] Here we show why this must be the case.

Two forces are at work when improving knowledge represented as bi-directional models: (a) Improving the *precision* of the atomic models and, (b) increasing its *scope* by covering as many variables as possible with the models.

The mechanisms we present show that the models created by a good cumulative modeler will, over time, increase their usefulness in guiding the system's behavior – the cumulative nature of the modeling means that while it is ongoing the system continues to improve its knowledge of the environment [15]. A cumulative learner's capabilities, as a result, grow incrementally over time, in relatively small but frequent steps, to ultimately cover a wide range of tasks.[5] This is important because when the modeler makes itself the subject of the modeling, it can potentially also improve itself—implementing what has been called *bounded recursive self-improvement* [9]—in a safe, incremental fashion.

2 Agent and Task-Environment

In the physical world, some of the environment's variables are observable, $\mathcal{V}_o \subset \mathcal{V}$, some are manipulatable, $\mathcal{V}_m \subset \mathcal{V}$, and some are related via causal links such that changing one variable will affect another.

An agent situated in an environment (Fig. 1) consists of a controller (c) that hosts a modeling process (P_M) capable of generating causal-relational models $\mathcal{M}$ of the environment through experience, testing their validity through observation and direct intervention in the environment. The agent has a perception cone through which it can receive input from observable variables ($\mathcal{V}_o$) and actuators through which it can affect the state of manipulatable variables ($\mathcal{V}_m$). At any point in time these will be limited to a subset of the total set of observable and manipulatable variables due to I/O bandwidth and its specific location. If the world is highly asymmetric—that is, features of any of its part in one area are highly dissimilar to features in other areas—make any acquired knowledge

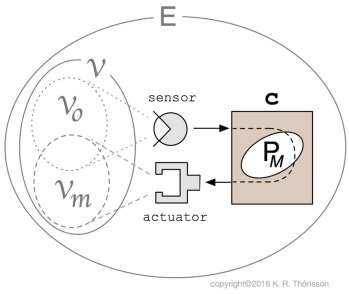

Fig. 1. An agent situated in an environment (E) consists of a controller (c) that hosts a modeling process (P_M) capable of generating causal-relational models of the environment through experience.

[4] Any reliable and repeatable regularity in a world is considered a causal relation, irrespective of whether it is observable or not, or truly deterministic or not [3].

[5] The speed of accurate model building is of course of critical importance for any real world implementation, determined in part by the details of the implementation methods and the nature of the task-environment; in this paper, however, the primary focus is on theoretical aspects of the modeling process.

heavily dependent on an agent's localization; highly self-similar worlds will not make knowledge generation dependent on agent position.

In our conceptualization, a cumulative learner will proceed to create models of the relations between observed variables. If V_1 and V_2 are both variables of the world, linked by the relation $V_1 R V_2$, with $R \in \Re$, what the model "claims" is a causal link between O_1 and O_2, the observable parts of V_1 and V_2. For instance, if A is the result of the observation of O_1 and B is the result of the (subsequent) observation of O_2, then the model represents a (hypothesized) causal relation between A and B, i.e. $A \Rightarrow B$ (and indirectly, that V_1 caused V_2).

Of the set of variables $\mathcal{V}$ in any complex environment such as the physical world, only a fraction are observable, and even a smaller fraction of those are observable in any given time interval. An event involving an observed relationship between any two or more variables, where for that time interval some relevant variables are unobservable, will make it indistinguishable from a partially random relationship, even if it is fully deterministic "under the hood". Same goes with the manipulatable variables $\mathcal{V}_m$: they only represent a fraction of the variables of the world, and may not be a strict subset of $\mathcal{V}_o$, as actions of the agent may have unobservable effects.

This means that for any world with a very small ratio of $\mathcal{V}_o/\mathcal{V}$, a large number of relations between observable variables will seem "probabilistic" to an agent—even if that world is fully deterministic (i.e. all related variables are truly deterministically coupled). Therefore, an agent in such environments will neither be able to model its task-environment perfectly nor fully, and a number of its models will be incorrect some of the time.[6]

3 Models and Modeler

Before even beginning to model, our cumulative modeler must be given a well-defined *seed*, composed of (at least) one top-level goal G_{top}, one model connecting this goal to an observable variable, one model acting on a manipulative variable, and one primitive action to take. This is really the theoretical minimum – practical implementations will contain quite a bit more; the more complete and thorough the seed is the easier it will be for the system to bootstrap and start learning. If the seed does not reference anything that the system can measure the system won't be able to evaluate the results of its actions, and thus cannot grow its model set. The same goes for goals (the seed must contain at least one concrete objective to attain) and executable actions (there must be at least one

[6] It should be noted that causal relations cannot be replaced by probabilities. Pearl [12] (p. 36) states: *"...causality deals with how probability functions change in response to influences (e.g., new conditions or interventions) that originate from outside the probability space, while probability theory, even when given a fully specified joint density function on all (temporally-indexed) variables in the space, cannot tell us how that function would change under such external influences. Thus, 'doing' is not reducible to 'seeing', and there is no point trying to fuse the two together."*.

action the system can perform). The smaller the seed, other things being equal, the longer the system will take to bootstrap its knowledge.

When the agent is inserted into the environment its modeling process starts generating models using information in the seed. This proceeds by noting correlations in the stimuli coming in through the senses (including sensations related to the agent's own end-effectors), and for any $A \Rightarrow B$ pair, where A precedes B in time, creating a model that predicts what may happen when you see an A, and suggesting that when you want a B you may want to make A happen.

The modeler must have the ability to round up the relevant models at any point in time. This is done by the agent's *scheduling process*. If there are too many models models for the controller to sort through, the threshold for any model to be considered relevant to the present moment could be increased, thus decreasing the total number of models the system must work with at any point in time. The speed at which this can be done directly affects a controller's ability to bring the right knowledge to bear on any situation, as it determines how many models the controller can consider during forward (deduction) and backward chaining (abduction). This is important because it specifies, respectively: the controller's capacity to consider (a) alternative futures (by triggering various different models that look promising for predicting next events based on e.g. the current state) and (b) alternative ways of achieving goals (by searching through models abductively to find plausible paths through which any state may have been formed). While these do not bear directly on our arguments in this paper, they are of great importance when considering practical implementation of these mechanisms.

3.1 Structure of Causal-Relational Models

Our concept of a *causal-relational model* is used here in a specific way; prior work provides detailed examples of how such models may be implemented [9,11], while here we are more concerned with the theory of such models.

Models are executable information structures encoding procedural knowledge and are either provided up front (by the designer) in a seed or created by the modeler – with the latter set becoming much larger than the former over time. The models our cumulative modeler creates are composed of a left-hand term (LT) and a right-hand term (RT). The LT (the "input") refers to a preconditional pattern, composed of values, variables, ranges, etc., that make the model relevant[7] for a particular situation (via pattern matching); the RT represent the post-conditions of the terms the model refers to. When the LT pattern is observed, a prediction based on the RT pattern is produced. In this forward-chaining process a set of models compute predictions on given LT inputs via deduction. For instance, if a model takes two consecutive $\{x,y\}$ coordinates of the path of a Pong ball and computes its next position using a linear transformation formula, this is a prediction of a future state of a particular entity

[7] Relevance is determined at "the top" by top-level goals, and at the "bottom" by incoming stimuli through sensors; in between the pattern matching on the models' LT and RT determines their relevance.

(the ball) in particular circumstances (moving along a path, i.e. correct until the ball hits something). A good model will thus produce a true prediction using valid deduction.

Reciprocally, when an instance of a model's RT pattern is observed that is a goal, a sub-goal patterned after its LT pattern is produced. In this case backward-chaining answers the question "how could RT be achieved?". Models produce sub-goals when super-goals match their RT pattern, these sub-goals in turn match other models' RT pattern until – unless the chaining is halted for some reason – a sub-goal produces a command for execution by I/O devices.

When an input other than a goal or a prediction matches its RT, an assumption is produced, based on the LT. In either case the task of the scheduler is to find the model(s) whose RT matches the state, and reading the model "in reverse" by looking at its LT. This constitutes generating abduction hypotheses, answering the question "how might the RT have come about?".

After a model is used we check to see whether the predicted outcome was correct or not; each model stores the number of correct uses over the total number of uses, whose ratio is used to determine the preference for which model should be chosen when an input matches the LT (when forward chaining, RT when backward chaining) of two or more models.

Models further contain two sets of functions that compute values for variables featured in the RT, from the values held by variables in the LT (one set for forward and another for backward execution).

4 Modeling Process for Cumulative Learning

Now we describe the basic operations of a canonical cumulative learner, and show that by generating small modifications to existing models and testing these through observation and manipulation, the modeling process implements cumulative learning.

The modeling process consists of two sets of models $\mathcal{M}$ and $\mathcal{M}_{hyp}$, and their interactions with the environment E (Fig. 2). The first set, $\mathcal{M}$, is used to compute and predict the state of the system. This set of models interacts directly with the environment: it receives informations from some observable variables in the world and can act on some variables (the manipulatable ones), as explained in the preceding section.

The second set, $\mathcal{M}_{hyp}$, is the "experimentation lab" of the modeler, whose purpose is to test new models without interfering with the environment. Models of this set are variations of those from $\mathcal{M}$. These variations are based on alternative *contextualization*, hypothesized *generalization*, and proposed *compression* of existing models. Models in $\mathcal{M}_{hyp}$ are tested as the environment and $\mathcal{M}$ interact. When a model $M_1 \in \mathcal{M}$ is triggered, we will test what would have happened if it would have been replaced by its variation, $M'_1 \in \mathcal{M}_{hyp}$. A comparison of the model set using the new model versus the original one determines whether the new model is deleted (if it produces no improvement or is simply wrong), or is added in $\mathcal{M}$ (if a performance improvement over the original set is detected).

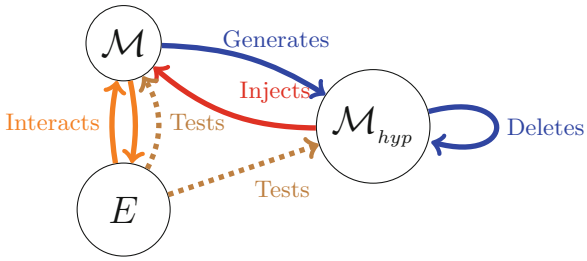

Fig. 2. Interactions between the different sets of the system (see text).

The original model M_1 may then be deleted from the sandbox or kept (if, for some contexts not addressed by the new model, the original one is still useful).

The initiation to generate a new model M_0' is the triggering of a model M_0 in $\mathcal{M}$ (i.e. it is considered relevant for the present state). Depending on the actual usefulness of M_0 in the present situation (for prediction and/or intervention), the modeler will create (at least one) variation of M_0 in $\mathcal{M}_{hyp}$. If M_0's prediction is correct, the modeler will try to make it more general by removing some of the constraints in its LT, in an attempt to make it more general. If M_0's prediction is incorrect, models will be created with the observed outcome as output (targeting the same observation) and more specific input than M_0, making it less general. In this case M_0 may still be relevant and is not deleted.

When several models are used in series they may be compressed into a single model with a more detailed or specific input. While not improving $\mathcal{M}$'s coverage (scope), reducing model count may increase the modeler's runtime efficiency.

4.1 Description of Modifications

The key to the idea of proving that our cumulative modeler is a good one is to show that each of the small modifications will result in a small improvement of the system. To do so, we will detail each of the modifications and explain why—should the modification be shown to be a good one—injecting it in $\mathcal{M}$ will improve the ability of the system to predict and achieve goals in $\mathcal{D} \subset E$ due to the models in $\mathcal{M}$ matching more closely actual relationships between variables in $\mathcal{D}$.

Contextualization. New models will be created when a triggered model $M_0 \in \mathcal{M}$ produces incorrect predictions (they could be incomplete, partially correct, or totally wrong). What we want in this case is, if I is the set of all inputs that M_0 was triggered on, to partition it into two sets I_0 and I_1. Then the original model M_0 will be transformed in M_0^0. It will be the same as M_0 with a LT that will only match inputs in I_0. A new model M_0^1 will also be created, with a LT that match I_1, and the RT that was just witnessed.

Whether these new models are effective or not becomes known when a state arises where they are considered relevant. Two properties must be verified to say that this results is an improvement.

- If $i \in I_1$, (M_0 correct on i) $\Rightarrow$ (M_0^1 correct on i)
- $\exists i \in I_1$,(M_0^1 is correct on i and M_0 is wrong on i)
- We already have that if $i \in I_0$, (M_0 correct on i) $\Rightarrow$ (M_0^0 correct on i)

If one can find a couple of models (M_0^0, M_0^1) such that the above properties are verified, then one has found a couple of models that are strictly better than the original one M_0.

Generalization/Induction. Another case of new model creation is when a model M_0 from $\mathcal{M}$ is triggered, and its prediction is correct. New models that are more generic will then be created, with the same RT output as M_0, but with LT inputs that are less specific (suppression/deletion of input variables, greater range of values, variables replacing values, etc.). To be injected into $\mathcal{M}$ a model thus created should be as efficient on the LT input where M_0 is correct, and work well on other inputs not specified in M_0's LT. The new model should be verified to not interfere with existing models in $\mathcal{M}$. If that's the case, and the model is failing on inputs that are caught by other models, it should be modified to prevent it form competing with such existing models.

When it is verified that the new model is more general than M_0, it will then simply replace it in $\mathcal{M}$. When evaluating this change in $\mathcal{M}$, what we want to verify is simply that the new model works "better" than the old one. Here, working "better" has three implications:

- M_0 is correct on input $i \Rightarrow M_0'$ is correct on input i. This is ensured by the fact that M_0' LT is simply a less specific version of the one on M_0.
- M_0' is correct on input i, and i did not trigger M_0. This is ensured by the fact that if M_0' was moved from $\mathcal{M}_{hyp}$ to $\mathcal{M}$ it has been tested to be correct on such inputs.
- There is no input i on which M_0' will make incorrect predictions that would have been caught correctly by other models in $\mathcal{M}$. This is ensured by checking that M_0' does not interfere with existing adjacent models (that have variables in common).

Compression. The last case of creating new model(s) is when several small models $(M_i)_{i\in[|0,n|]}$ are compressed into a single bigger one, and when a model sequence (M_0, then M_1, etc.) is replaced with a new model that has as its input a mix of the input variables from $M_0, \ldots, M_{n-1}$, and the same output as M_n. Models should be added in $\mathcal{M}$ only if they are at least as correct as the original ones. The objective of such compression is not to achieve a better model than the original ones but rather to reduce computational requirements.

This modification category is somewhat less important than those above and should only be used if it is desired to have a greater number of more specific models, rather than a lower number of more generic ones. This represents, however,

a general way to tune resource usage [5] and can be done when faster results are desired. The original models can of course in all cases be kept and strategically retrieved, e.g. should more precision be needed or more computational resources become available.

4.2 Producing Causal-Relational Models

Due to the bi-directionality of the models they will tend towards capturing true causal relations between variables of the environment. To see why, consider the situation where a cause A has two effects, B and C (Fig. 3). We assume that to the modeler A appears before B and C, but B and C appear together. Four models could be used to describe what is observed every time we see an A:

1. Model M_1: $B \Rightarrow C$
2. Model M_2: $C \Rightarrow B$
3. Model M_3: $A \Rightarrow B$
4. Model M_4: $A \Rightarrow C$

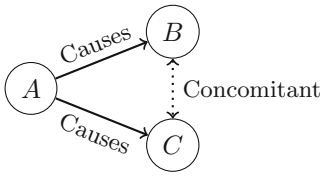

Fig. 3. Example causal relations between A, B and C.

Any of these models will predict correctly: If you see B you will see C, and vice versa; if you see A you will see B and C. However, not all of them can be used to achieve goals in the domain of A, B and C: If you want to stop seeing C it does not help to remove B, or vice versa – due to the causal structure of this task-environment, only model M_4 will help. Thus, when each of these models is used for both prediction and goal achievement models M_1 and M_2 will be deleted due to their incorrect predictions. What remains are M_3 and M_4, the only models that capture actual causal relations in the domain, to the extent that this can be represented as relationships between *observable* variables.

Each modification of the model set in $\mathcal{M}$, using the methods above, makes them more reliable within a given sub-domain $\mathcal{M}_\mathcal{D}$. Repeated usage and testing of the models increases the overall reliability in small steps, as they capture the target phenomena. The system is continuously trying to improve each of its models, hypothetically reaching the maximum precision allowed by the environment and the allowed time and resources. When this point is reached, every phenomenon is modeled as well as possible.

5 Conclusion

We have presented a modeling process that implements a good cumulative modeler, improving continuously to eventually be able to operate in a wide range of task-environments. Based on combined abduction and deduction over causal-relational models, the methods described have been implemented and tested [9–11,16], producing notable results not demonstrated by other learning systems. While important questions about practical issues remain to be investigated,

e.g. the extent of its generality and scaling, this paper demonstrates that the basic principles are relatively clear and concise and meet the criteria for cumulative learning. Our results so far indicate that these ideas are significantly different from most other approaches, and potentially a promising approach to achieving AGI.

References

1. Conant, R.C., Ashby, W.R.: Every good regulator of a system must be a model of that system†. Int. J. Syst. Sci. **1**(2), 89–97 (1970)
2. French, R.M.: Catastrophic forgetting in connectionist networks: causes, consequences and solutions. Trends Cogn. Sci. **3**(4), 128–135 (1999)
3. Halpern, J.Y., Pearl, J.: Causes and explanations: A structural-model approach – part 1: Causes. CoRR abs/1301.2275 (2013). http://arxiv.org/abs/1301.2275
4. Hasselmo, M.E.: Avoiding catastrophic forgetting. Trends Cogn. Sci. **21**(6), 407–408 (2017)
5. Helgason, H.P., Nivel, E., Thórisson, K.R.: On attention mechanisms for AGI architectures:a design proposal. In: Bach, J., Goertzel, B., Iklé, M. (eds.) AGI 2012. LNCS (LNAI), vol. 7716, pp. 89–98. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-35506-6_10
6. Kirkpatrick, J., et al.: Overcoming catastrophic forgetting in neural networks. Proc. Nat. Acad. Sci. **114**(13), 3521–3526 (2017)
7. Lazaric, A.: Transfer in reinforcement learning: a framework and a survey. In: Wiering, M., van Otterlo, M. (eds.) Reinforcement Learning. Adaptation, Learning, and Optimization, vol. 12, pp. 143–173. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-27645-3_5
8. Lu, J., Behbood, V., Hao, P., Zuo, H., Xue, S., Zhang, G.: Transfer learning using computational intelligence: a survey. Knowl.-Based Syst. **80**, 14–23 (2015)
9. Nivel, E., et al.: Bounded Recursive Self-Improvement. Technical RUTR-SCS13006. Reykjavik University Department of Computer Science, Reykjavik, Iceland (2013)
10. Nivel, E., Thórisson, K.R., Steunebrink, B., Schmidhuber, J.: Anytime bounded rationality. In: Bieger, J., Goertzel, B., Potapov, A. (eds.) AGI 2015. LNCS (LNAI), vol. 9205, pp. 121–130. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-21365-1_13
11. Nivel, E., et al.: Bounded seed-AGI. In: Goertzel, B., Orseau, L., Snaider, J. (eds.) AGI 2014. LNCS (LNAI), vol. 8598, pp. 85–96. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-09274-4_9
12. Pearl, J.: Bayesianism and causality, or, why I am only a half-Bayesian. In: Corfield, D., Williamson, J. (eds.) Foundations of bayesianism. Applied Logic Series, vol. 24, pp. 19–36. Springer, Dordrecht (2001). https://doi.org/10.1007/978-94-017-1586-7_2
13. Pollock, J.L.: Defeasible reasoning and degrees of justification. Argum. Comput. **1**(1), 7–22 (2010)
14. Steunebrink, B.R., Thórisson, K.R., Schmidhuber, J.: Growing recursive self-improvers. In: Steunebrink, B., Wang, P., Goertzel, B. (eds.) AGI 2016. LNCS (LNAI), vol. 9782, pp. 129–139. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-41649-6_13

15. Thórisson, K.R., Kremelberg, D., Steunebrink, B.R., Nivel, E.: About understanding. In: Steunebrink, B., Wang, P., Goertzel, B. (eds.) AGI 2016. LNCS (LNAI), vol. 9782, pp. 106–117. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-41649-6_11
16. Thórisson, K.R., et al.: Autonomous acquisition of situated natural communication. Comput. Sci. Inf. Syst. **9**(2), 115–131 (2014). outstanding Paper Award
17. Wang, P.: Insufficient knowledge and resources - a biological constraint and its functional implications. In: AAAI Fall Symposium: Biologically Inspired Cognitive Architectures. Citeseer (2009)