Studies in Computational Intelligence 1049

Jerry Swan · Eric Nivel ·
Neel Kant · Jules Hedges ·
Timothy Atkinson · Bas Steunebrink

The Road to General Intelligence

Springer

Studies in Computational Intelligence

Volume 1049

Series Editor

Janusz Kacprzyk, Polish Academy of Sciences, Warsaw, Poland

Jerry Swan · Eric Nivel · Neel Kant · Jules Hedges ·
Timothy Atkinson · Bas Steunebrink

The Road to General Intelligence

Springer

Jerry Swan
NNAISENSE SA
Lugano, Switzerland

Eric Nivel
NNAISENSE SA
Lugano, Switzerland

Neel Kant
NVIDIA
Santa Clara, USA

Jules Hedges
University of Strathclyde
Glasgow, UK

Timothy Atkinson
NNAISENSE SA
Lugano, Switzerland

Bas Steunebrink
NNAISENSE SA
Lugano, Switzerland

ISSN 1860-949X ISSN 1860-9503 (electronic)
Studies in Computational Intelligence
ISBN 978-3-031-08019-7 ISBN 978-3-031-08020-3 (eBook)
https://doi.org/10.1007/978-3-031-08020-3

To Doug Hofstadter, for showing that programs can be piano as well as forte.

Foreword by Melanie Mitchell

The idea that machines could one day exhibit 'general intelligence' has both inspired and confounded the field of AI since its inception. 'Inspired' by drawing talented people into the field to achieve one of humanity's grandest challenges. 'Confounded' since there is no widely agreed-upon definition of what 'general intelligence' actually means or a definitive list of the properties it entails.

The pursuit of general intelligence in AI has been, in large part, the story of moving goal posts. Variously called 'human-level AI', 'strong AI', 'AGI', and even 'superintelligence', the criteria for machines exhibiting general intelligence have continually changed over time. Alan Turing, in his classic 1950 paper 'Computing Machinery and Intelligence', proposed that a machine should be considered intelligent (or 'thinking') if it could persuade a human judge via conversation alone. However, the rise of ever more capable chatbots (and the surprising propensity of humans to assign intentionality to machines) has shown that AI conversationalists that clearly lack general intelligence can easily fool humans in many cases. Early AI proponents believed that something like general intelligence could be captured in systems that heuristically followed symbolic rules. Many of the early founders of AI predicted in the 1960s that human-level AI was only 10, 15, or at most 20 years away. However, symbolic AI approaches, such as the optimistically named 'General Problem Solver', turned out to be far from general and often disastrously brittle.

Intellectual board games like chess and Go have long been seen as a grand challenge for AI systems, and many people believed that conquering them would require something like general intelligence. In 1958, AI pioneers Newell and Simon declared, 'If one could devise a successful chess machine, one would seem to have penetrated to the core of human intellectual endeavor'. And in 1997, the year that IBM's Deep Blue defeated world chess champion Garry Kasparov, the New York Times conjectured about Go, 'When or if a computer defeats a human Go champion, it will be a sign that Artificial Intelligence is truly beginning to become as good as the real thing'. But although Deep Blue and its counterpart AlphaGo are extraordinary achievements, neither is anywhere close to a general intelligence.

Since the 2010s, the rise of deep learning (and its myriad successes) has once again encouraged optimism among some in the AI community that 'true AI' is close at hand. More recently, though, cracks have begun to appear in deep learning's facade of intelligence, and many have expressed serious doubts about the prospects of 'big data' approaches for developing general AI.

This book is an attempt to clarify what kind of knowledge representation and information processing is needed for general intelligence in machines. The authors, inspired by theories of semantics as well as programming-language theory, stress the need for representations that exhibit both 'algebraic compositionality' and 'strong typing'—properties of programming languages that allow for explicit propagation of arbitrary constraints, rapid adaptation to new inputs, and reflection (in which a process can examine its own behavior). This is in contrast to the representations formed by today's deep learning systems, which seem to have none of these properties.

Most people in AI would agree that even when machines exhibit what seems like intelligent behavior (for example, producing coherent translations between languages) these machines don't have anything like the kind of understanding of their inputs or their own behavior that humans have. The notion of 'understanding' in machines is hard to pin down, but the authors of this book note that, at the very least, understanding must entail the ability to transfer what one has learned to new tasks, a capacity that today's state-of-the-art AI systems still struggle with, but is highly desired by the world of automation. The authors frame this notion as 'Work on Command'—the ability to respond, in a reasonable time, to dynamic changes in the specification of the task one is faced with.

This book includes extensive discussion of additional abilities required for general intelligence—for example, abduction, analogy, and hypothesis generation. Capturing such abilities in AI systems in a general and humanlike way has been the subject of much research but little progress to date, in part due to the lack of progress in capturing the causal knowledge and reasoning that underlies them. Here the authors describe how such abilities have been implemented in a reference system that exhibits what they call Semantically Closed Learning (inspired by the concept of semantic closure in open-ended evolution, and incorporating further ideas from category theory).

In short, this book provides an intriguing and provocative framework for thinking about what general intelligence is, and how its essential abilities might be attainable by machines in an economically viable manner. The philosophy behind both programming-language theory and category theory plays key roles in the formalization and development of the main ideas. The authors also provide pointers to what research challenges lie open. Given the complexity and intricacy of the destination, the road to general intelligence will be a bumpy one. This book gives a thought-provoking view of one pragmatic direction toward this goal.

Santa Fe, NM Melanie Mitchell

Foreword by David Spivak

It’s no coincidence that companies have departments, that furniture has drawers and shelves, that bodies have organs, that code bases have modules: things organize for a reason. As Herbert Simon points out in ‘Sciences of the Artificial’, organizing offers exponential compression in the search space for solving problems. By carving nature at its joints, by elegantly articulating and factoring the space of possibilities, and by finding the right abstractions, we enable ourselves to handle new situations in stride.

Mathematics is the marketplace for humanity’s clearest and most reliable abstractions. And within mathematics, category theory is unparalleled in its ability to factor ideas into constituent parts that fit together frictionlessly. In this book, the authors emphasize a subdiscipline of category theory called bidirectional transformations, also known by names such as polynomial functors and optics, which I join them in regarding as essential for expressing the crucial role of feedback in intelligent systems. Finding categorical abstractions to handle new situations, including creating more intelligent systems, requires a widespread research program that is only now starting to emerge. As far as I know, this is the first book to confidently assert this need and make a real stab at solving it.

Berkeley, CA David Spivak

Contents

Chapter 1
Introduction

Theories may be equivalent in all their predictions and are hence scientifically indistinguishable. However, different views suggest different kinds of modifications which might be made and hence are not equivalent with respect to the hypotheses one generates from them.

Richard P. Feynman, Nobel Lecture 1965

The rise of civilization is synonymous with the creation of tools that extend the intellectual and physical reach of human beings [133]. The pinnacle of such endeavours is to replicate the flexible reasoning capacity of human intelligence within a machine, making it capable of performing useful work on command, despite the complexity and adversity of the real world. In order to achieve such Artificial Intelligence (AI), a new approach is required: traditional symbolic AI has long been known to be too rigid to model complex and noisy phenomena and the sample-driven approach of Deep Learning cannot scale to the long-tailed distributions of the real world.

In this book, we describe a new approach for building a situated system that reflects upon its own reasoning and is capable of making decisions in light of its limited knowledge and resources. This reflective reasoning process addresses the vital safety issues that inevitably accompany open-ended reasoning: the system must perform its mission within a specifiable operational envelope.

We take a perspective centered on the requirements of *real-world* AI, in order to determine how well mainstream techniques fit these requirements, and propose alternative techniques that we claim have a better fit. To reiterate: by AI we mean the property of a machine that exhibits general-purpose intelligence of the kind exhibited by humans, i.e., enjoying the ability to continually adapt existing knowledge to different domains. The endeavor to create intelligent machines was definitively proposed as such in the 1950s [220], although the concept of a humanoid automaton recurs throughout recorded history. Due to the sheer magnitude and ambition of the project, there have naturally been many bumps in the road: not only the infamous 'AI winter' [202], but also periods where the endeavor's vision and direction have been clouded by the prospects of short-term success.

J. Swan et al., *The Road to General Intelligence*,
Studies in Computational Intelligence 1049,
https://doi.org/10.1007/978-3-031-08020-3_1

AI for Automation

Given that substantial resources are required to create AI, it cannot be done on a whim. Therefore the shape of AI (at least in its initial incarnation) will be strongly influenced by the return anticipated by those investing in it. That is, to answer "How to build AI?", we must ask why we want AI in the first place, i.e., what is the *business case* for a machine with general intelligence?

Philosophical considerations aside, intelligent machines are ultimately tools for implementing a new leap in *automation*. In practical automation settings, the generality of a system is measured as the inverse of the cost of its deployment and maintenance in a given environment/task space. At the low end of this spectrum are systems that depend on full specifications of their environments and tasks. Such systems are very costly to re-deploy when facing specification changes, possibly incurring the highest cost: that of a complete rewrite. At the high end are more general systems that re-deploy autonomously through continual open-ended adaptation and anticipation.

The main functional requirement of general intelligence is therefore to *control the process of adaptation*. In this work, we claim that this can be achieved in a unified, domain-agnostic manner via the ability to ground arbitrary symbols (whether arising from end-user vocabulary or being synthesized by the system) in an explicit learned semantics. Hence, throughout this work, when we discuss *symbols* in reference to our proposed architecture, it is not in the sense of the a priori opaque logical predicates of 'Good Old-Fashioned AI', but rather follows in the footsteps of a collection of cyberneticists, psychologists and systems theorists [8, 67, 218, 253, 261, 269, 299] for whom *"symbols are merely shorthand notation for elements of behavioral control strategies."* [49].

In practical terms, the endeavor of creating general intelligence therefore consists of building a template for a *learning control system* which can be re-targeted at an arbitrary environment, bootstrapping the control mechanisms with as little latency as possible, starting from small amounts of (incomplete or even faulty) knowledge. The system is then expected to discover further constraints on the fly—be it from a corpus of ready-made knowledge; from experience acquired with and without supervision; perhaps by interacting in the environment, possibly under the sporadic guidance of teachers and end-users.

Notwithstanding these business considerations, the creation of AI still relies on good science, especially with regards to requirements engineering, with the initial focus illustrated in Fig. 1.1. Although we set aside those requirements that are mostly issues of hardware, paperwork, or procedures (e.g., constructing curricula for teaching the system as well as its eventual operators), the fact that they must be addressed and fulfilled then imposes constraints on which scientific techniques can even be considered. The requirement-centric perspective dictates which properties are important for a technique to exhibit or avoid. For example, even legal requirements impinge on techniques, such as when GDPR[1] demands transparency in automated decision

[1] General Data Protection Regulation (EU) 2016/679.

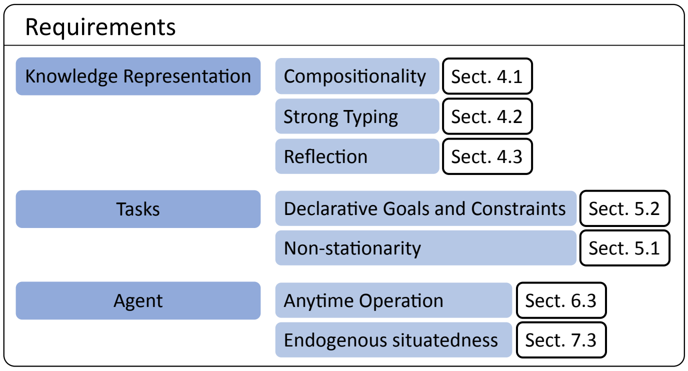


Fig. 1.1 Theme development in Part I: a summary of the most pertinent engineering requirements for constructing a general intelligence system of real-world use. Their importance is established throughout the first part of this book and then leveraged to construct our proposed framework: Semantically Closed Learning

making, which is more easily fulfilled when knowledge representation and reasoning are not intrinsically black-box components or processes.

The Structure of this Book

This book begins with a survey of historical (Chap. 2) and contemporary (Chap. 3) AI methodologies, discussing their strengths and weaknesses, from the perspective of their potential to support general intelligence. Machine learning (ML), notably deep- and reinforcement learning, has emerged as the dominant AI paradigm. There are certainly many valuable applications for which ML offers functionally good solutions, in particular for industrial applications where such techniques are used to build control systems beyond the reach of traditional software engineering. Nevertheless, it remains a feat of imagination to ascribe any meaningful notion of intelligence to any of these systems: the constraints and ambitions of machine learning and general intelligence research are simply orthogonal. Although machine learning is a valuable *engineering technique*, this fact is not to be confused with a *claim* that it might offer a path toward general intelligence. In Chaps. 4 and 5, we make a critical appraisal of this claim, by contrasting deep learning and reinforcement learning techniques against key requirements of general intelligence—from the perspective of automation engineering, these are reified by the notion of 'Work on Command' in Chap. 6.

The second part of the book is concerned with an alternative framework that we claim fulfills better these requirements. There is increasing consensus that it is necessary to combine the strengths of both symbolic and connectionist paradigms [59, 210]: the main advantage of symbolic approaches is the ready injection of domain knowledge, with the attendant pruning of hypothesis space. In contrast, the main advantage of connectionism is that it is (at least in principle) a *tabula rasa*.

As has been argued by Marcus for many years [214], we also hold the view that general intelligence requires the recursively algebraic capacities of human reasoning. This motivated the research and associated reference architecture implementation we present in this book. This architecture has been implemented, and prototypes have been developed, addressing the domains of medical diagnosis, service robotics, and industrial process automation—empirical demonstrations will be the topic of subsequent works. In Chaps. 7–10, we define a framework for 'Semantically Closed Learning' which:

- Describes an explicit (but nonetheless 'universal') recursive interpreter for a highly generalized notion of algebraic reasoning.
- Represents the hierarchical causal structure of hypotheses as first-class objects.
- Defines a fine-grained and resource-aware attention mechanism, driven to favor highly-structured and stable hypotheses.
- Describes key reasoning heuristics using the generic and compositional vocabulary of category theory, from the emerging perspective of 'Categorical Cybernetics' [42, 140].
- Defines a novel compositional mechanism, using *lenses* [88], an approach which unifies conventional backpropagation, variational inference, and dynamic programming, for the purpose of abductive reasoning over hybrid numeric-symbolic expressions.
- Describes a minimal viable implementation design for 2nd *order automation engineering*—system identification, synthesis, and maintenance—with *guarantees* relevant to safety.

Finally, in Chap. 11, we summarize our contribution, discuss research avenues, and conclude.

Part I
Requirements

Chapter 2
Background

It's all these black boxes you can't open—see how each spends most of its time trying to defeat the other.

Knuth [368]

Recent years have seen an explosion in academic, industrial, and popular interest in AI, as exemplified by machine learning and primarily driven by the widely-reported successes of deep- and reinforcement learning (e.g. [314, 315, 351]). Deep learning is essentially predicated on the notion that, with a sufficiently large training set, the statistical correlations captured by training will actually be causal [310]. However, in the absence of convergence theorems to support this, it remains a hypothesis. Indeed, insofar as there *is* evidence, it increasingly indicates to the contrary, since the application of enormous volumes of computational effort has still failed to deliver models with the generalization capability of an infant. There is accordingly increasing discussion about what further conceptual or practical insights might be required [57]. At the time of writing, the very definition of deep learning is in flux, with one Turing Award laureate defining it as "a way to try to make machines intelligent by allowing computers to learn from examples"[1] and another as "differentiable programming".[2] We argue in the following that deep learning is highly unlikely to yield intelligence, at the very least while it equates intelligence with "solving a regression problem". Specifically, we claim that it is necessary to adopt a fundamentally different perspective on the construction of inferences from observations, and that this is in accordance with a fundamental revolution in the philosophy of science: Karl Popper's celebrated solution to 'The Problem of Induction' [268].

[1] https://blogs.microsoft.com/ai/a-conversation-ai-pioneer-yoshua-bengio.

[2] https://www.facebook.com/yann.lecun/posts/10155003011462143.

© The Author(s) 2022
J. Swan et al., *The Road to General Intelligence*,
Studies in Computational Intelligence 1049,
https://doi.org/10.1007/978-3-031-08020-3_2

In subsequent chapters, we first describe the significant challenges for machine learning. We then argue that current approaches are unlikely to be able to address them unless the scope of the learning framework is considerably widened, and that it will ultimately be necessary for this framework to be both situated and to support reflective reasoning. We compare our proposed conceptual framework for learning with that of reinforcement learning, with respect to the features necessarily associated with general intelligence. We propose a roadmap towards general intelligence, progressing via the notion of *work on command* to the property of *semantic closure*. This property is described fully in Chap. 7, but in précis, it equips an agent with the ability to determine causality and induce hierarchical representations in a manner that is absent from traditional machine learning approaches.

2.1 What we Mean by General Intelligence

As convincingly argued by Wang [357]: before any in-depth discussion of roadmaps and obstacles, we must define the desired destination of general intelligence. Legg and Hutter [194] give a comprehensive and insightful tour of various definitions of intelligence. They distill many of these into the following observations:

> *Intelligence is not the ability to deal with a fully known environment, but rather the ability to deal with some range of possibilities which cannot be wholly anticipated. What is important then is that the individual is able to quickly learn and adapt so as to perform as well as possible over a wide range of environments, situations, tasks and problems.*

These observations along with others culminate in the mathematical formalism of *universal intelligence*. This defines the intelligence of an agent as being equal to the average of the returns (sum of rewards) it can obtain across all possible environments, weighted by the complexity of those environments.

The preceding definition is problematic in that it assumes the existence of an *a priori* reward function. In natural organisms, any such reward function is assumed to be provided by a combination of innate and cultural mechanisms. For artificial systems, the implied complexity of a constructed reward function becomes a concern. In the extreme, if the reward function were to be completely arbitrary, then it effectively characterizes the environment as pure noise, with no useful features for a learner to exploit. Conversely, once the reward function is structured so as to reflect regularities in the environment, then at least some (and potentially a great deal) of the purported intelligence of the agent is actually provided by the reward function itself.

In subsequent sections, we argue that some of the issues with ML are actually an artifact of this kind of ‘narrow framing of the problem’, in which (1) human expertise is required to represent each specific problem in a manner amenable to ML and (2) this framing then only makes a highly impoverished form of feedback available the learner, typically in the form of a scalar numeric reward. We argue that it is necessary to replace black box reward functions with richer representations that are suitable for reflective reasoning.

In pursuit of greater generality than can be provided by an *a priori* reward function, we must therefore adopt the wholly pragmatic perspective of the following value proposition.

The Value Proposition for General Intelligence

For all practical purposes, general intelligence is a necessary property of a system which:

- Performs *work on command*
  i.e., responds with tolerable latency to dynamic changes in goal specification and environmental conditions.
- Scales to real-world concerns.
- Respects safety constraints.
- Is explainable and auditable.

2.2 Science as Extended Mind

There is clearly a huge gap between the software which enables facial or gait recognition and the yet-to-be-realized technology which will allow safe and trustworthy autonomous vehicles or factories. One can likewise consider the reality gap between audio-activated digital assistants and fully-fledged household robots. There exist countless other examples of roles that current AI techniques are incapable of fulfilling. In roles where humans are currently irreplaceable, what *traits* enable them to meet the demands of these roles? The generality of human intelligence is evident in many ways:

- Humans can handle multiple objectives simultaneously and can typically order activities so as to meet these objectives relatively efficiently.
- Humans can learn skills without forgetting those previously learned. They can also make efficient use of related skills to bootstrap their learning process and minimize this effort.
- Humans can explain their decision-making in terms of relevant causal factors and 'locally consistent' frameworks of thinking, which means that a recipient of the explanation (perhaps also their subsequent self) can understand, verify, and possibly rectify the steps taken to reach conclusions.
- Humans can be told what is desired directly as a goal rather than needing to iteratively try behavior in the hope of optimizing some sampled metric.
- Humans can be told what is forbidden and/or constrained and they can avoid such situations without needing to physically interact with (i.e. 'sample') the environment, assuming relevant grounded world knowledge.

- Humans can gracefully adjust their cognitive resource usage between perception, action, and learning rather than having rigid boundaries between them.[3]
- Humans can operate in multi-agent settings, mostly through being able to effectively model other agents' trajectories based on their perceived intentions and behavioral patterns.
- Humans can do all of the above in the real world, perhaps with a curriculum, but not needing a high-fidelity resettable/reversible simulation within which to learn.

For the purposes of this work, the above list of traits will be considered as a set of necessary emergent capabilities of general intelligence. With this motivation, we claim that a system which exhibits these traits must satisfy the requirements summarized in Fig. 1.1.

While the human mind is the most immediate exemplar for general intelligence, we believe there are strong reasons to consider that the *scientific method* is better suited to provide a template for its implementation. As Rodney Brooks has famously observed [37], insights obtained via mental introspection might cause us to be deeply misled about the nature of intelligence. In contrast, the adoption of the scientific method yields falsifiable statements about the physical world. This can be seen as providing an 'extended mind' [50]—an externalized artifact with *verifiable properties* that can directly inform the design of general intelligence architectures. Given the inevitable concerns about 'AI alignment',[4] such verifiability is of particular importance in obtaining measures of *safety*. Hence, we believe that the path to general intelligence (at the very least, in a form capable of respecting safety concerns) lies in the attempt to automate the scientific method, from the perspective of an embodied reasoner with real-world concerns of deadlines and resource availability.

Recent years have seen increasing emphasis on causality in machine learning. Causality is essential for building reasoning systems as it is a stronger criterion than merely statistical correlation. Originally having been convincingly argued for by Pearl [254], the relevance of causality to AI has been since agreed upon by Schölkopf, Bengio, and others. One of the key ideas is the 'ladder of causality,' which is framed as inference situated on three 'rungs': the observational, interventional, and counterfactual settings. Statistical learning from fixed datasets operates solely the observational rung: training on data generated only by an external process which the model does not affect. Interventions imply the ability to set values of certain variables despite the natural external processes in order to generate informative data; for example, double-blind experiment design with control groups. The most demanding yet powerful application of causality is counterfactual reasoning, where inferences are drawn based on variable values which were never observed but generated through interventions in a model; for example, alternate history timelines.

Pearl also introduced the *Structural Causal Model* (SCM), which is a directed acyclic graph structure specifically designed to enable users to operate on all three rungs of the ladder of causality. A key idea in the SCM formalism is that dependencies

[3] Whilst not necessarily under conscious control, nonetheless a property of human cognition overall.

[4] The quest for confidence that a general intelligence won't attempt to turn everything into paperclips [30].

between variables are framed as probabilistic *functions* rather than simply statistical dependence. This is better aligned with a physical interpretation of observations, namely that they are caused by physical processes over time. The distributions of a set of variables X_i are given by the formula:

$$X_i = f_i(\mathrm{PA}_i, U_i), \quad (i = 1, \ldots, n)$$

where PA_i are the parent nodes of X_i. The functions are probabilistic due to U_i, which are exogenous noise variables which are jointly independent of one another. If there were dependencies, they could be explained by forming yet more causal relationships (as per the common cause principle), and so noise must be modeled as independent.

Interventions in SCMs are defined as (temporarily) setting f_i to be a constant. Importantly, the distributions of the parent nodes are unaffected by interventions on children since their relationships are effectively severed, which is different from standard Bayesian networks. Instances of the latter only denote conditional independence relationships as undirected graphs, and so dependence between nodes persists even if the value of one is set. The ability to perform interventions also allows for principled counterfactual reasoning in SCMs. If we had observed some value for a node X_i, we can use abduction to estimate the value of U_i, and then after intervening on its parents PA_i, re-apply the observed exogenous noise in order to produce a counterfactual inference. There are ubiquitous problems which require counterfactual reasoning that are consequently intractable for purely statistical models [254]. Given their apparent completeness for causal modeling, the modern problem of causal discovery involves deducing the topology of an SCM which can accurately describe the system.

Despite widespread interest in the use of SCMs, it is vital to appreciate that, within scientific practice, causality is best understood as being only part of a contextualized process of situated, bidirectional inference. Hence we take the deeper view of science as the construction of statements which provide a concise and consistent description of *possible worlds*. As recently observed by David Deutsch:

> *Finding causal theories is necessary but not sufficient. We need* explanatory *theories. "Mosquitos cause malaria" is essential and useful but only 1% of the way to understanding and curing malaria.*

The essence of the scientific method is, of course, the interleaving of problem formulation, hypotheses generation, experimentation, and analysis of results. Hence, a core aspect of the proposed approach is the requirement for a reflective expression language. Reflection is the property that statements can themselves be treated as data, meaning that hypotheses about knowledge in the language can be evaluated as first-class objects. This becomes salient for the process of *hypotheses generation*.

Concretely, for our purposes hypotheses are some (sub)graph of inferences in the system's transition model. Mappings from sensor inputs to effectors (or in the opposite direction, in the case of abductive inference) are just specific fragments of this overall model, starting or terminating in appropriately designated sensor or effector dimensions. Naturally, it is desired that the only hypotheses that are entertained by the system are those which (1) actually describe a possible world and (2) are relevant to the task at hand. Considered from a 'traditional symbolist' perspective, the latter is of course equivalent to the well-known 'Frame Problem' [219], which, as discussed in subsequent chapters, is increasingly understood to have been an artefact of coarse-grained and disembodied inference.

As we discuss in detail in Chaps. 7 and 9, it also follows that any reasonable candidate architecture for encoding knowledge for general intelligence will be *compositional*, so as to obtain a semantics for compound hypotheses. Another essential property is the notion of *strong typing*, which enables inheritance/taxonomy and the explicit denotation of goals and constraints as regions of a (prospectively open-ended) state space. These properties jointly enable structured updates to working knowledge that retain the self-reinforcing nature of a scientific theory [89].

Finally, the modern scientific method requires that all working knowledge should be *in principle falsifiable* via empirical observation. Hence, we include as a requirement to our approach that the base symbols of the expression language must include denotations which are grounded in this way. Causal modeling also stipulates the ability to intervene directly in the environment to learn the effects of one's own agency. Thus, the system and representation language must both support primitives for interacting with the environment.

We claim that it will not be possible (or economical) to automate human labor in the general case until AI also possesses these properties. As such, this is the context in which we will highlight the challenges and shortcomings of deep learning, reinforcement learning, and other existing AI approaches. To place contemporary approaches in the appropriate context, we proceed via a brief historical recapitulation of the rise and fall of the traditional symbolist approach.

2.3 The Death of 'Good Old-Fashioned AI'

The key figures at the inaugural AI conference at Dartmouth [220] were split across the nascent symbolist and connectionist divide, with Simon, Newell, and McCarthy in the former camp, and Shannon, Minsky, and Rochester in the latter [174]. However, the symbolist approach became the prevailing one, not least because of the widespread confusion surrounding the solvability of the XOR problem by perceptrons [226]. The following decades saw concerted effort in symbolic AI. Many of the languages used to construct AI originated from the synthesis of procedural and logical programming styles, respectively exemplified by LISP and resolution theorem provers. Hewitt's 'PLANNER' language [142] was a hybrid of sorts, being able to procedurally interpret logical sentences using both forward and backward chaining. It was used to construct SHRDLU [366] which was hailed as a major demonstration of natural language understanding. This inspired other projects such as CYC [197], an ongoing attempt to create a comprehensive ontology and knowledge base that seeks to capture 'common-sense knowledge'. Less ambitious and more successful were the various expert system projects that started in the 1960s and became prevalent in the following two decades. These included the MYCIN expert system for diagnosing infectious disease [349], Dendral for identifying unknown organic molecules [38] and other well-known systems such Prospector [135]. Collectively, such systems became known as 'Good Old-Fashioned AI' (GOFAI) [137].

Common Attributes and Roadblocks

In general, GOFAI fits the template of a knowledge-based system. Such systems may be decomposed into two components: a knowledge base and an inference engine which answers queries and/or enhances the knowledge base by applying inference rules to the current state of the knowledge base. These systems typically required users to create the symbolic primitives and inference rules a priori. It gradually became understood that such information was difficult to obtain—the so-called 'knowledge elicitation bottleneck'.

Two other key longstanding GOFAI problems are the Qualification Problem and the Frame Problem [219], both of which contributed to the perception of scalability issues. The Qualification Problem is concerned with the preconditions needed for a logical deduction to be valid. The Frame Problem is concerned with the difficulty of fully specifying conditions related to invariants of state space transformation.[5] Another key obstacle to scalability is the 'cognitive cycle' of the 'sense–think–act' paradigm [240], in which it is assumed that the world evolves in lockstep with the system. This quickly became a formidable obstacle for early projects such as the General Problem Solver [238] and PLANNER [142]. At that time, computation was also far more expensive than it now is—together, these two things meant that GOFAI was destined for a reckoning.

The End of GOFAI

((The prospect of general intelligence using such rule-based systems was not highly appraised by observers. In the early 1970s, the Lighthill Report [202] lead to a drastic reduction in AI research funding by the UK government and DARPA cut funding to academic AI research in the USA. In the late 1980s, the United States' Strategic Computing Initiative, which was essentially formed to participate in an AI/computing race with Japan, cut funding for new AI research as its leaders realized that the effort would not produce the full machine intelligence it desired. Simultaneously, the market for large-scale expert system hardware collapsed and more affordable general-purpose workstations took over. These developments formed part of what is now colloquially known as the 'AI Winter'.

In hindsight, many of these challenges and the accompanying demise of GOFAI could be said to be a function of the hardware of the time. Computing power and mem-

[5] It was eventually concluded that solutions exist to these problems, including default logics [347] and answer set programming [201].

ory were obviously more expensive and software design decisions (such as deciding between the use of LISP or C/C++) had a correspondingly disproportionate impact on what could be computed in practice. A number of companies built upon LISP-based expert systems (such as Symbolics, LISP Machines Inc., Thinking Machines Corporation, and Lucid Inc.) went bankrupt. Ambitious undertakings were common, such as the Fifth-Generation Computer Systems (FGCS) project in Japan during the 1980s. The massive investment in building highly parallel computing systems was ultimately in vain as simpler, more general architectures such as workstations from Sun Microsystems and Intel x86 machines became favored for such roles. Some of those forward-looking ideas have been reinvented in the early 21st century, such as the emphasis on highly parallel programming from FGCS now commonplace in general-purpose GPU programming with CUDA and OpenCL.

The Problem of the 'Sense–Think–Act' Loop

Regardless of technological advances, the GOFAI paradigm still does not present a viable path to general intelligence. For the architectures relying on ungrounded knowledge representation, there is no prospect of deploying them to address tasks in the real world of complex and noisy data streams. More fundamentally, the absence of grounding precludes the understanding of causal relationships of the real world—a core aspect of operationalizing the scientific method. Even if a GOFAI system were hypothetically to achieve symbol grounding, there would still be a fatal flaw: GOFAI never matured sufficiently to escape the scalability problem inherent in the 'sense–think–act' loop. As the system's body of knowledge grows, the time required to make plans and predictions must also increase. As engineers put it, the system 'lags behind the plant' and faces two options: either to deliver correct action plans but too late, or to deliver on time plans that are incorrect [241]. This issue arises essentially from the *synchronous* coupling of the agent and its environment, i.e., the latter is expected to wait politely until the agent completes its deliberations. Technically speaking, synchronicity means that the agent computes in zero time from the environment's perspective.

Machine learning has failed to acknowledge the significance of this problem and has even adopted GOFAI's synchronous coupling as one of the fundamentals of reinforcement learning; see Chap. 3. For now, computation is scaling at a rate that can sustain the synchronous abstractions used in large-scale projects (see Sect. 5.2). For lower-level routines (such as reactively handling sensory streams of data at a fixed frequency) this may suffice. On the other hand, there are certainly aspects to cognition which are slower, more deliberative and explicitly logical, and this is where the 'sense–think–act' approach breaks. Some believe that given more resources and innovations in model architectures, deep learning may be able to encode knowledge effectively enough to empower this sort of cognition and meet the requirements for general intelligence. In the next chapters, we shall see why this, in fact, cannot be the case.

Chapter 3
Where is My Mind?

It was like claiming that the first monkey that climbed a tree was making progress towards landing on the moon.

Dreyfus, 'A History of First Step Fallacies' [73]

The research field of AI is concerned with devising theories, methods, and workflows for producing software artifacts which behave as intelligent subjects. Evidently, intelligence, as the property of an agent, is not of necessity inherited from the methods used to construct it: that a car has been assembled by robots does not make it a robot.

Unfortunately, even this obvious distinction can sometimes be erased in some prominent published work. To wit: the statement, "an agent that performs sufficiently well on a sufficiently wide range of tasks is classified as intelligent" was recently published by DeepMind [273] to give context to a paper claiming to have developed "the first deep RL agent that outperforms the standard human benchmark on all 57 Atari games" [14]. This invites the inference that the range of the tasks (57 games) that have been achieved warrants calling the advertised agent 'intelligent'. However, careful reading of the paper reveals that the authors have in fact developed 57 different agents. Granted, this was achieved using the same development method and system architecture, but 57 agents were nonetheless trained, rather than the claimed single agent. Here is a prime example of distilled confusion: a property (applicability to 57 tasks) of one construction method (instantiating the Agent57 system architecture) has just been 'magically' transferred to some 57 artifacts produced by the method.

This only fuels what Marcus terms the "epidemic of AI misinformation" [211] from which the world has been suffering for some years [165]. As a minimum, this damages public understanding (impinging on business expectations/validation), trust (in scientific deontology), and education at large. Indeed, in common with others [75] we are of the opinion that such 'misinformation at scale' steers both governance and research the wrong way: it gives credence even among some seasoned researchers—

J. Swan et al., *The Road to General Intelligence*,
Studies in Computational Intelligence 1049,
https://doi.org/10.1007/978-3-031-08020-3_3

or worse, the next generation of researchers—to claims that machine learning is the (only) root of general intelligence, a myth we debunk in the next two chapters. But first, to give the matter a proper grounding, we must return to the roots of ML in order to objectively assess its domain of application, appreciate its evident achievements, and delineate the boundaries of its potential.

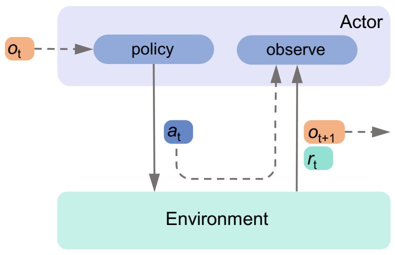
3.1 A Sanity Check

With the demise of GOFAI came a renewed effort to explore connectionist learning paradigms. Over time, the field shifted from the symbolic languages of GOFAI to 'end-to-end' feature learning. This has loosened the constraints on knowledge representation, namely moving away from an expression language with discrete symbols to something more representationally amorphous. Learning the parameters of a function approximator then becomes a fitting and regularization problem, as conceptualized by the bias-variance trade-off.

Inspired by trial-and-error learning in animals, reinforcement learning (RL) developed from work in optimal control, which is concerned with minimizing some metric of a dynamical system over time. The techniques developed for solving these problems became known as dynamic programming, and also produced the formalism of the Markov Decision Process (MDP), which is now essential to RL [331]. It provides the abstractions of states $s \in S$, actions $a \in A$, and reward function $r\colon S \times A \to \mathbb{R}$. The evolution of states is assumed to progress in discrete steps according to the transition function $P\colon S \times A \to \Delta(S)$, with a scalar reward $R_t = r(s_t, a_t)$ produced at each timestep t. These ingredients can be modified to accommodate additional complexity such as partial observability, continuous spaces, and multiple agents.

The purpose of RL is to optimize a value function based on sampled rewards which are stationary and specified a priori, in order to produce an agent (called a controller) consisting essentially of a policy that maps states to actions. It does so by shaping the policy encoded in a neural network[1] generally either via gradient descent over its weights or by 'neuro-evolution', i.e., optimizing the fitness of a policy within a population thereof. Training is unsupervised and uses for its ground truth a corpus of trials and errors logged from an number of sessions of interaction (episodes) with a simulated world—the number of episodes grows with the complexity of the task/environment distribution and is generally enormous (in other words, the sample efficiency is inordinately low). Note that, although it is possible in principle to perform trials and errors in the real world instead of in a simulator, this is rarely the case in practice, for fear of wear and tear of equipment and safety risks (on sample inefficiency and safety concerns, see Sect. 5.2).

[1] In early days, policies were encoded as mere lookup tables (as in Q-learning [208]), but this could not scale with the increasing dimensionality of the problems, hence the need to compress said policies via deep neural networks, hence 'deep RL'.



```
1: while True do
2:   $t \leftarrow 0$; $e_t \leftarrow$ False
3:   Reset environment to get $o_0$
4:   Observe $o_0$
5:   while not $e_t$ do
6:     $a_t \leftarrow \pi(o_t)$
7:     Take action $a_t$; obtain step $\eta_t = (r_t, o_{t+1}, e_{t+1})$
8:     Observe $(a_t, \eta_t)$
9:     Update the policy $\pi$
10:    $t \leftarrow t + 1$
11:  end while
12: end while
```

Fig. 3.1 State-of-the-art high-level RL training procedure (Hoffman et al. [143]). The fact that, in practice, the procedure is implemented in sophisticated ways does not change its fundamentals

As illustrated in Fig. 3.1, the training process is synchronously coupled with the environment. The procedure it implements consists of successive sessions of interaction, each composed of a number of cycles: a cycle starts by putting the simulator on pause, invoking a subprocedure for guessing the next action based on the previous world state, and feeding the action to the simulator which responds immediately with a reward—the collected rewards are ultimately used to adjust the policy. The simulator is then resumed, it updates its state and a new cycle starts. At some point, another subprocedure decides to stop the cycle and a new session is initiated by resetting the simulator. When the controller performs well enough over selected samples of the training distribution, the procedure halts—note that the test distribution is required to be the same as the training one.

This is the basic procedure. Depending on the needs, the procedure is generally augmented with various support systems such as short-term memory, episodic memory, improved guessing subprocedures (such as intrinsic motivation heuristics and meta-learning, i.e., in-training adjustment of the guessing subprocedure), world models (model-based RL), hierarchies of policies (hierarchical RL), and so on. Regardless of its sophistication, the procedure is never learned nor performed by the controller itself: it is instead designed and performed (using a dedicated tool chain) by *human workers*. Once deployed in production, the controller matches, repeatedly, the same learned policy against sensory inputs and directs the resulting actions to its actuators (‘inferencing’ in ML parlance). In the main, such ‘inferencing’ merely amounts to applying the neural network encoding the policy to an input vector thus producing an output vector, i.e., a single mathematical operation. Regardless of how this operation is implemented, the controller is purely reactive (one could say ‘blind’) and does not make any situated deliberation worthy of the name, let alone taking any initiative. As Hervé Bourlard (a prominent ML researcher, head of the Swiss Idiap Research Institute) recognized recently in Forbes, “Artificial intelligence has no intelligence” [44].

In technical terms, a typical RL policy is a 'curried planner', where it is curried[2] with respect to a predefined goal/environment coupling. This stands obviously at the antipodes of what one expects from a system that learns to respond to arbitrary orders of its user in a world over which system designers have little or no control. Such a discrepancy is not contingent: it bears witness to the orthogonality of the respective ambitions of RL and general intelligence.

The purpose of RL is essentially to automate the production of software dedicated to handling a specific task in a specific environment, both defined a priori. From an engineering perspective, RL amounts to a compilation procedure. Granted, it compiles interaction logs instead of, say, C++ code, but compilation it does nonetheless ('compression' being the preferred nomenclature in ML). Assuming the resources required to build a fast[3] simulator are available (domain knowledge, funds, etc.), if one can guarantee that the world and the task will forever remain as they were during training,[4] then RL constitutes a valid cost-efficient alternative to standard engineering procedures for some classes of problems.

3.2 Real-World Machine Learning

ML has been successfully applied to automate tasks of increasing apparent complexity such as playing video games, albeit demonstrating very little with regards to real-world applications where much higher levels of complexity are the norm.

In contrast, the much less publicized application of ML to industrial automation is a far better and more rigorous exemplar of the value of ML in generating fast and accurate controllers for complex machinery in unforgiving environments. Examples abound: improving cooling in data centers, optimizing network routing for high-performance computing, optimizing chip layout, predictive maintenance, robot manipulation, digital twinning, etc. In this significant business-relevant and constrained domain, ML enjoys steady progress towards broader applicability—again, the fundamental principles are not challenged, nor do they need to be for the time being. In particular, motivated by industrial requirements, it is of critical importance to keep policy-defined behaviors within prescribed operational envelopes and this is an area of ongoing research [48, 100, 231, 270, 275, 346].

To re-emphasize: according to the expectations of the ML method, models ('policies' in case of RL) are produced in the lab according to client-provided specifications, then these models are frozen to constitute the core of deployed controllers. When the environment or task changes or whenever there is a need for improvement, new models are trained in the manufacturer's lab to replace the old ones, following

[2] https://en.wikipedia.org/wiki/Currying.

[3] Because the RL procedure is synchronous and highly sample-inefficient, simulators must be fast enough to keep the training time under manageable limits.

[4] The task is fixed and the deployment environment is drawn from the same distribution as the one used for training.

the standard procedure of software update. This is unsurprising: like any other software engineering method, ML addresses predefined controlled conditions—a claim of 'intelligence' is neither made or required; client-side engineers know what to expect.

Note that, in compliance with the continuous integration/continuous delivery paradigm (CI/CD) that pervades modern software engineering (for better or worse), it is possible to accelerate the rate of the updates. In principle, this is achieved by expanding the training distributions via the parallel aggregation of multiple data sources (as long as they are deemed, by humans in the loop, to pertain to one same task/environment), then training new policies in the background and updating the deployed controllers asynchronously.

Beyond direct application in standalone deployments, ML-generated controllers are also used as components of larger systems. For example, some of the building blocks of the Multi-level Darwinist Brain architecture (MDB) [20] consist of policies operating on world models. Reminiscent of hierarchical RL, the selection of a policy among the repertoire is itself a policy, subjected to a satisfaction model (intrinsic motivation). World models and the satisfaction model are initially given by the designers, with some guarantees that they are general enough to support a determined set of tasks and environmental conditions. Learning of policies and models is performed offline via neuro-evolution, subject to hand-crafted value functions. In that sense, an MDB controller still depends critically on the foresight of its designers, which limits its potential for autonomy. Yet it can be altered after deployment (incurring some inevitable downtime) in an incremental fashion at a cost arguably lower than that of re-engineering from scratch. This line of work culminates for example in the DREAM architecture [68]. Whereas MDB representations are tuned, ad hoc, to individual control policies, DREAM introduces a generic slow refactoring loop (called 're-representation') that continually extracts common denominators from existing representations in order to form more abstract ones to eventually facilitate transfer learning, in support of a user-defined training curriculum.

There is no contention about the value of ML as long as claims remain aligned with established principles and proven capabilities. As we have seen, whereas an agent is learned, the deployed agent does not itself learn or act autonomously. This is fully consistent with both the principles of ML and its goals and makes perfect sense from an engineering/economic point of view. What is obviously more debatable is the claim that a process designed for manufacturing purely reactive software will eventually produce thinking machines. The next two chapters will question, in depth, the plausibility of such a prophecy.

Chapter 4
Challenges for Deep Learning

Are you a good noticer*? Do you notice things well? I mean, for instance, supposing you saw two cock-starlings on an apple-tree, and you only took one good look at them — would you be able to tell one from the other if you saw them again the next day?*

Hugh Lofting, 'The Voyages of Doctor Dolittle' [205]

Deep learning (DL) has emerged as the dominant branch of machine learning, becoming the state of the art for machine intelligence in various domains. As discussed in the previous chapter, this has led some researchers to believe that deep learning could hypothetically scale to achieve general intelligence. However, there is increasing consensus (e.g. [57, 210, 230]) that the techniques do not scale as well as was anticipated to harder problems.

In particular, deep learning methods find their strength in automatically synthesizing distributed quantitative features from data. These features are useful insofar as they enable mostly reliable classification and regression, and in some limited cases also few- or zero-shot transfer to related tasks. However, it is increasingly questionable whether deep learning methods are appropriate for autonomous roles in environments that are not strongly constrained. While there are still countless use-cases for narrow artificial intelligence, many of the truly transformative use-cases can only be realized by general intelligence.

We recall from Sect. 2.2 that, while we do not know the internal mechanisms of human general intelligence, we observe that 'science as extended mind' is a pragmatic description of a general intelligence model of the environment. However, neural network representations are not readily interpretable, either to humans or more importantly—as we subsequently argue at length—to the learning process itself.

This chapter has two purposes: the first is to explain what properties are wanting (their relationship to the entire book is shown in Fig. 4.1) and to elicit fundamental

J. Swan et al., *The Road to General Intelligence*,
Studies in Computational Intelligence 1049,
https://doi.org/10.1007/978-3-031-08020-3_4

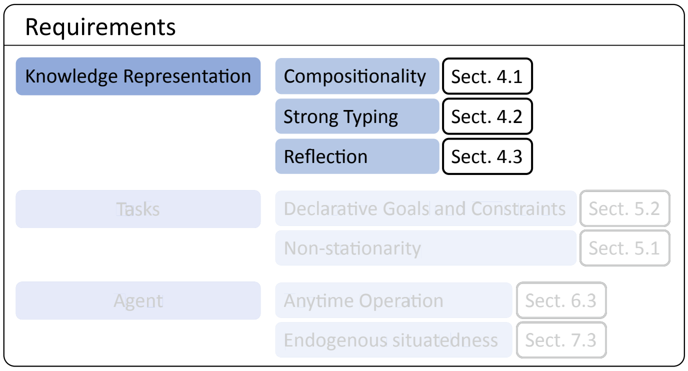


Fig. 4.1 In this chapter we argue that the lack of structure in representation languages created via deep learning are in conflict with the requirements of general intelligence

obstacles posed by deep learning. The second purpose is to argue that 'science as extended mind' offers a more effective perspective for designing the desired system. The latter is further developed in Sect. 7.2 and is the foundation of our proposed inference mechanisms in Chap. 9. The following claims contrast deep learning with the requirements for operationalization of the scientific method:

- Representations are *not compositional*, which make them inefficient for modeling long-tailed distributions or hierarchical knowledge.
- Representations are *not strongly typed*, which prevents verification against adversarial scenarios and hinders generalization to new domains.
- Representations are generated by models which *do not support reflection*, which restricts model improvement to gradient-based methods.

4.1 Compositionality

There has recently been increasing emphasis on the importance of compositionality [120] for machine learning. To take a famous example from AI history [149, 304], humans do not require an a priori hypothesis to react to the outlier case of 'goat enters restaurant'. Knowledge about goats can be freely composed with knowledge about restaurants; for example, the sudden arrival of a goat would not generally be expected to preserve a tranquil dining atmosphere or good standards of hygiene [281]. It has recently been stated by some of the most renowned exponents of deep learning [21] that:

> *We believe that deep networks excel because they exploit a particular form of compositionality in which features in one layer are combined in many different ways to create more abstract features in the next layer.*

However, this notion is far weaker than is actually required, in particular for purposes of AI safety but (as we subsequently argue) also for scalable inference via greater sample efficiency. The weakness of this notion of compositionality is evidenced by numerous challenges for deep learning (discussed in more detail in this and subsequent chapters):

- Adversarial examples.
- Weak generalization capability.
- Inability to explicitly induce or propagate more than a small number of types of invariant (translation, rotation, etc.).

Indeed, it could be said that DL is closer to the merely syntactic notion of *composability* than the semantic notion of compositionality. In its most degenerate form, the syntactic notion is merely the observation that feature ensembles are instances of the 'composite' design pattern [101, 369] and hence hierarchically aggregated features are syntactically substitutable for individual ones. However, that does not impose any intrinsic *constraints* on what the features represent or what the ensembles compute. In contrast, compositionality is defined as [207]:

> *The* algebraic *capacity to understand and produce novel combinations from known components.*

The term 'algebraic' here effectively means 'having well-defined semantics', in the sense that the behaviour of a composite exhibits constraints that are a function of those of its component parts. The role played by the alleged compositionality of DL is lacking in almost every respect of this definition: in algebraic terminology, the feature representations in DL layers can be '*freely* composed'. In contrast, in Chap. 9 we describe a mechanism for imposing a *denotational semantics* on composite representations.

Hence, the only *property* in ML for which there is a guarantee of generalized, end-to-end compositionality is differentiability [90]. If, as seems likely, it is necessary to express more directly whether or not some desired property is compositional, then this requires extending DL far beyond 'differentiable programming'. In common practice, composability in DL consists of assembling a network from constituent parts which may be trained 'end-to-end'. Usually, this follows the *encoder-decoder* pattern where the encoder is responsible for generating vectorized features, and the decoder maps the features for classification or regression. This paradigm is common in deep learning applied to sequential data or labels. Example domains are text-to-text [66, 276, 330, 370], image-to-text (and vice-versa) [169, 173, 283], and program synthesis [46, 123, 124, 284]. When *explicitly tasked* with the generation of compositional representations, neural networks have been observed to exhibit better generalization performance [5]. However, as observed throughout the literature (e.g. Liska et al. [204]), more complex architectures tend not to scale well, showing limited scope. We argue the following as the most salient consequence:

Claim 1: DL and Compositionality

Deep learning appears to be fundamentally limited in its ability to create compositional knowledge representations. This severely inhibits the effective formation and use of structured, hierarchical knowledge, which in turn results in weak performance in domains with *long-tailed distributions*.

Hierarchical Behavior and Representations

In advance of more detailed discussion in the following chapter, we briefly consider here some work from deep reinforcement learning (DRL), so-called for its use of DL for knowledge representation. Deep reinforcement learning has seen much work on encapsulating learned behaviors into skills [94, 199, 222, 312] and options [13] to be composed in a hierarchical fashion. Researchers are motivated by the potential for hierarchy to reduce planning horizons, branching factors and improve sample efficiency. Despite interesting results that point to progress in these directions (e.g. [170]), it is not clear whether these approaches scale to more difficult problems. Nachum et al. [235] study these methods in particular and find that the benefits of hierarchical policy composition have more to do with exploration than with the imposed structure, and that the same benefits can be obtained with a modified exploration technique and a 'flat' policy.

Other work [65] explores embeddings for tasks which can be composed arithmetically, in a similar manner to deep word embeddings [223]. However, subsequent work on sentence and document embeddings [58, 125, 256] suggests arithmetic compositionality of properties encoded via embeddings is a difficult constraint and that little besides differentiability is scalable for compositional representations. In certain settings, recursion can effectively be used to hierarchically compose the interfaces of deep learning architectures [41, 244]. However, this composition is still at a coarse granularity, and it appears unlikely that arbitrary properties can be composed by this means. As such, it cannot be said that DL gives scalable solutions for building hierarchical knowledge, and this will most certainly limit DL's overall scalability toward general intelligence.

Robustness in Long-Tailed Distributions

It is important to note that practically all domains of interest contain long-tailed distributions, particularly if they are grounded in the real world. Indeed, it can be argued that if long tails are not encountered in the data, then that represents a limitation of the dataset and subsequently the evaluation of the model. For example, an autonomous vehicle in an unconstrained environment will need to deal with an endless Borgesian list of edge cases [274]:

- Telling the difference between a shallow puddle and an impassable flood.
- Obeying signage written in human language for an intelligent human being to understand.
- Figuring out what another driver means when they flash their headlights.
- Badly scuffed or missing road/lane markings.

- Pulling on to a kerb or through a red light to allow an emergency vehicle to pass.
- Correctly determining from context whether a traffic light is knocked askew or genuinely pointed at a different lane of traffic from this one.
- …and so on, ad infinitum.

It has been observed that symbolic representations are well-suited for long-tailed distributions because of the potential to map recursive expression syntax into complex semantics [214]. In contrast, the issue of long-tailed distributions is not sufficiently emphasized in current deep learning research. This is evident in both the confidence and emerging scrutiny of natural language processing models. Despite more comprehensive benchmarks such as GLUE [354], the combinatorial nature of natural language expressions acts in direct opposition to the notion that a 'representative' training corpus can be reasonably sized. When highly-parameterized models such as GPT-2 [276] are thoroughly analyzed [206, 212], they reveal an understanding merely on the level of association, far from the depth required for anything like human-level understanding of what has been parsed. This problem has also been repeatedly identified in neural program synthesis, where program induction should be robust to all unseen inputs. For example, the Neural GPU [163] was trained to do long addition and multiplication. While results suggested robustness for problems into hundreds of digits in size, further work [272] revealed weaknesses when doing arithmetic involving many consecutive carry operations. Many other examples emerged concurrently of neural networks attempting to do program induction for arithmetic [183, 284, 303, 373] which also had a pattern of being unable to succeed on outlier cases. These examples clearly show that, even in domains which can be formally characterized, deep learning in its current form will not be of much use in the many cases where crucial inputs come from long-tailed distributions.

4.2 Strong Typing

In Chap. 2, we introduced two concepts claimed essential to general intelligence: 'work on command' and 'science as extended mind'. In the previous section we argued for the necessity of compositionality. The primary motivation for that argument is the need for scalability and robustness in the presence of long-tailed distributions. In this section, we argue another claim regarding representations, stated below.

Claim 2: Deep Learning and Types

Deep learning is not designed for generating typed representations. This deficiency is prohibitive for developing general intelligence, since strong typing is essential for invariant propagation, inheritance, verification, and rapid adaptation of existing inferences to new observations.

Types can be used to explicitly delineate subregions of a state space, which is important for specifying constraints and objectives given to an agent as well as the hypotheses constructed by an agent for explaining causal mechanisms. Deep learning essentially concerns itself with only a single type — that of numeric vectors, even for the incredibly large models which are increasingly used. The meanings of intermediate representations remain opaque and, we argue, underconstrained. For example, the statistical and observational nature of supervised learning means that training and test error can converge favorably without the constraint that intermediate representations capture causal relationships. This observation has raised widespread concern for the biases that may be present in deployed models which are used in sensitive situations such as loan approvals and prison sentencing [132, 359].

Adversarial Examples and i.i.d. Assumptions
Instead of type constraints, deep learning is built upon assumptions about the distributions of data to which it is applied. Most consequential is the requirement that training and test data are both independently and identically distributed (i.i.d.). This condition is essential for the strong convergence guarantees which are derived in statistical learning theory. In that context, such assumptions are perfectly reasonable, but are ill-matched for general knowledge representation and learning. The clearest indication of this is the existence of adversarial examples.

The most common formulation of adversarial examples are minutely perturbed inputs specifically designed to severely reduce supervised accuracy on deep learning models [121, 335]. Adversarial training has been developed in response, but new weaknesses have emerged [345] and an all-encompassing solution remains elusive. Meanwhile, this vulnerability has been confirmed in various scenarios related to image classification in the real world [184, 341] and variations applicable to reinforcement learning agents [112, 151] and natural language models [2, 158, 233, 249].

Adversarial examples are intriguing to humans since our perceptual systems are much more robust to such attacks. We argue that typed representations would prevent such catastrophic misrecognitions that have no clearly explainable origin in terms of DL parameter weightings. Objects can essentially be conceptualized as a set of invariances, such as shape being invariant to the brightness of shone light, texture being invariant to orientation, etc. These invariances anchor qualitative descriptions and allow us to construct part-whole and inheritance relationships through deduction [332].

Importantly, we cannot entirely eliminate high-dimensional inputs, since grounding is essential for a science-oriented agent. Instead, we argue that inducing types from raw high-dimensional data should be prioritized to occur at the lowest possible hierarchical level, since any higher level inference would benefit from the stability and clarity of typed language elements. We contrast this proposal with current techniques still bound to the i.i.d. paradigm, such as domain randomization [343]. While this has shown positive results in complex scenarios [31, 247, 255], there are currently no compelling reasons to believe that the method is scalable to levels required by general intelligence, providing only a relatively crude way to learn invariances which does not involve the distillation of new types. This of course is an open challenge and we discuss it further in Chap. 9.

Out-of-Domain Generalization and Meta-learning
An essential trait for general intelligence is the ability to efficiently leverage learned knowledge when facing a novel yet related domain. Existing literature describes techniques for *domain adaptation*, where a model can perform in another domain with few or no labeled examples. Domain-invariant feature learning [102, 232] and adversarial training methods [348, 352] have shown positive results for deep networks. Transfer learning and semi-supervised learning for deep neural networks are also well-studied topics.

Nonetheless, there is consensus that there remains much to be desired from deep learning in this regard. We argue that typed representations are the natural way to address this requirement for general intelligence. Humans are naturally capable of seeing new situations as modified versions of previous experience. In other words, there is an abstract type of which both the prior observation and the current stimuli are examples, but with certain attributes differing. Given enough new observations, it may be appropriate to reify a different type altogether. Rapid domain adaptation can also be modeled as a scientific exercise of determining an unknown type with minimal experimentation. We expand on this perspective in Sect. 7.2.

Meta-learning has emerged as a popular research topic aiming to expand the generalization of deep learning systems [86, 200, 227, 239, 286]. These methods train models to a location in parameter space which allows for efficient adaptation to unseen tasks as opposed to unseen data points. Conceptually, this may appear to expand generalization capacity. However, the framework makes assumptions about tasks coming from the same distribution, much as for individual data points in a dataset. As such, it suffers similar issues, such as inflexibility to non-stationarity in tasks. More related to generalization, meta-learning does not yield transferable abstractions, rather it gives an optimized starting point for creating adaptable models. As argued by Chao et al. [45], meta-learning is not fundamentally all that different from supervised learning. This makes it unlikely to truly resolve the challenges of generalization when the scope or nature of tasks are broadened.

4.3 Reflection

In the previous two sections, we made the case for compositionality and strong typing as necessary properties for representing knowledge for general intelligence. This section is concerned with what is needed in order to adapt that knowledge to make it more accurate and comprehensive. In deep learning, this process is handled as an optimization problem with the target being minimal error, which fits neatly with a purely numerical class of models. The incorporation of symbols via typed expressions complicates this but also offers new opportunities, which we now discuss.

The notion of *scalability* is applied in various ways. In this section we will draw attention to scalability in terms of sample efficiency with respect to training data. We expect that as an agent grows more intelligent, it should be able to evaluate and compare increasingly complex models with roughly the same efficiency as when it was less developed and learning about simpler phenomena. This is one of the merits of the scientific method: given two competing theories for physical reality, e.g. Newtonian mechanics and Einsteinian relativity, a single experiment (indeed, even a 'thought experiment') may suffice to decisively favor one model over the other.[1]

In this section we first characterize what learning looks like for deep neural networks and consider the choices that researchers make in order to support this sort of learning. Ultimately, we find that the resulting formulation is not well suited for kind of rapid, scalable learning that general intelligence requires and present the corresponding claim below:

Claim 3: DL and Reflection

Lacking compositionality and strong typing, deep learning also cannot support meaningful reflection over proposed models of its target domain. The property of reflection allows for direct, structured updates to knowledge, which compares favorably to deep learning's exponentially growing requirements for data and an undesirable dependence on end-to-end model design choices.

Knowledge and Optimization in Neural Networks

Much effort has been directed toward developing networks and optimization procedures which result in reliable training. This reasoning has paid off since deep neural networks are *universal function approximators*: a class of models that can approximate any continuous function to arbitrary precision [56, 198]. Regardless, training remains nontrivial because of the high-dimensional non-convex optimization involved. Consequently, neural networks continue to get larger at a rapid pace (e.g. [313]), often resulting in dramatic overparameterization. Work shows models are actually able to fit to almost any set of input-output pairs [203] including completely randomized ones [374]. Other interesting examples include the ability to compress networks while retaining accuracy [155] as well as the 'lottery ticket hypothesis' [93] which states that large networks regularly have sparse subnetworks of only one tenth the size, which themselves can achieve equal test accuracy. Analysis of neural networks from an information-theoretic perspective [302] shows that generalization follows from a great deal of internal data compression, which is also consistent with the notion that networks are larger than they need to be.

Gradient-based optimization can be considered a very simple form of reflection, wherein credit is assigned to individual parameters with respect to the error. For models using typed and compositional features, changes require maintenance of the associated semantics. Whereas the knowledge of deep learning models is tuned

[1] When contextualized appropriately by scale, of course.

in subtle and unpredictable ways, the ability to reflect on representations provides a basis for ensuring semantic consistency. We argue this stability during learning and amenability to direct updates are necessary properties of learning in general intelligence.

Furthermore, consider that deep learning is designed to result in *convergence* of the values of parameters. After convergence, adaptation to new tasks or modifying the knowledge is difficult and/or costly. Hence, deep learning with gradient descent does not adequately account for the necessity of open-ended learning for general intelligence. A more reflective learning approach is necessary to re-calibrate existing representations and render them actionable with respect to new goals and constraints. After some exposure to the new domain, a reflective approach can consolidate two models which share underlying similarities or abstractions back into a single, self-consistent and more general model. We explore more details and examples of these propositions in Chap. 9.

4.4 Implications and Summary

We reiterate that Claim 1 given at the beginning of this chapter does not detract from the merits of deep learning. To date, these methods have far outshone their predecessors in their ability to learn features from observational data and make valuable predictions from them. Most standard neural network architectures are also very amenable to parallelization and acceleration, making them practical for their current use cases. Hence, for some, the preceding discussion may not provide sufficient impetus to look beyond deep learning. For many practical applications in narrow domains, it suffices to have training methods which are applicable in the presence of relatively massive computational resources. In theory, recent work [81, 99, 236, 278, 309, 375] could yield representations which are causally disentangled and enjoy greater compositionality, but it seems likely that even the exponents of such forms of causal representation learning would admit that much progress is yet to be made.

We therefore recall the motivation which opened this chapter: the challenges for deep learning we have discussed arise in the context that the paradigm was not designed with general intelligence in mind. We can tie the challenges of machine learning together as being the symptoms of a 'narrow framing of the problem'. The most salient part of the framing is that deep learning model parameters are intended to converge to some satisfactory optimum given the dataset and iterative learning procedure. Training a model with a priori knowledge of the desired outcome is fundamentally at odds with the notion of open-ended learning [340], an essential part of general intelligence.

It should be emphasized that, even for the task of constructing general intelligence, we do believe that deep learning may be the most sensible and practical way to implement very basic layers of perception on high-dimensional sensory inputs such as visual and audio feeds. Although we have emphasized the importance of compositionality and strong typing, we also acknowledge that they may not always be applicable at the level of individual pixels or waveform amplitudes. Instead, it should be clear that compositionality and strong typing become increasingly relevant when the subjects being represented can usefully be compressed into 'concepts' or 'expressions' rather than mere sensory samples.

Chapter 5
Challenges for Reinforcement Learning

Theories destroy facts.

Peter Medawar [221]

Reinforcement learning was historically established as a descriptive model of learning in animals [234], [324], [32], [279] then recast as a framework for optimal control [331]. The definition of RL has been progressively expanded to be so broad and unconstrained [143] that virtually any form of learning can ultimately be described as RL. This has led some to consider RL as a plausible candidate for unifying cognition at large, hence subsuming general intelligence. In this chapter and the next, we identify fundamental issues that challenge this view. In summary:

- The notion of a *fixed a priori reward function* acts counter to open-endedness: assumptions of stationarity and the validity of 'once and for all' behavioral specifications are simply not adequate for open-ended behavior in the real world.
- For both efficiency and safety reasons, the notion of *reward sampling* prevents RL from being performed in the real world.
- The notion of *policy* conflates knowledge (world models) and motivation (goals)[1] via the direct mapping from states to actions: this opposes continual world modeling and plan composition.

To its credit, function approximation using 'deep' neural architectures has advanced the capacity of RL substantially beyond the capabilities of the original tabular setting. However, as described in Chap. 4, deep function approximation has serious limitations.

[1] Even though model-based RL leverages world models during learning, its end-product still remains a policy.

J. Swan et al., *The Road to General Intelligence*,
Studies in Computational Intelligence 1049,
https://doi.org/10.1007/978-3-031-08020-3_5

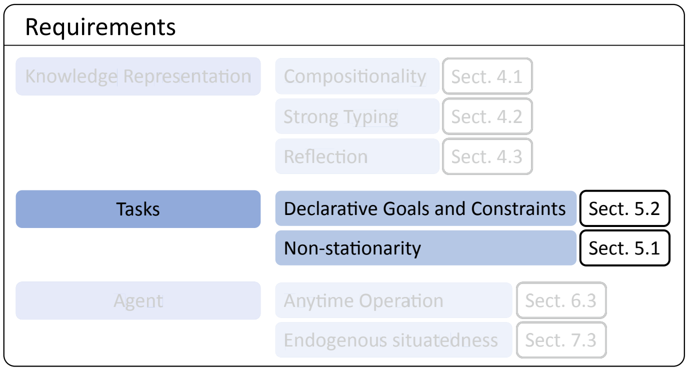


Fig. 5.1 In this chapter we argue that the centrality in RL of a priori fixed reward functions oppose key requirements for general intelligence

Claim 4

Reinforcement learning techniques which use deep neural networks for function approximation will inherit the issues stated in Chap. 4.

As per Sect. 2.1, we center our requirements around the continual adaptation of a world model to achieve a variety of arbitrary goals. We also stipulate that (1) learning without forgetting should be relatively cheap and (2) safety must be certifiable.

5.1 A Priori Reward Specification

Arguably one the most central concept in RL is the *value function* which usually express the value of a state or state-action pair in terms of the expected returns under some policy starting from that state—recall from Section 3.1 that a policy is simply a distribution of actions conditioned on states. The iteration of value functions toward a fixed point makes sense because the reward in RL is specified beforehand and is assumed to remain stationary. If we entirely commit to this strategy, then the only way to become more general is to widen stationarity over broader notions of tasks and environments. Learning is then extended to maximize reward over an entire distribution of MDPs [293], [305], [277] which share a state and action space whereas the reward function is selected from a distribution. These modifications, which are examples of *meta*-RL, purportedly allow for an agent to learn how to succeed over a variety of tasks, and would seem to be the most developed route in fully stationary RL towards general intelligence.

Notwithstanding those extensions, the value function is still an ideal place to scrutinize the foundations of RL. The assumptions of RL make it suitable to apply to problems where there is a naturally occurring quantitative objective to be maximized and there are ‘operationalized’ actions [11], as exemplified by e.g. video games, where even random behavior has non-negligible correlation with success. Even then, the RL engineer may have to do significant work to make a dense, shaped reward from a *sparse* reward, in order to obtain a sufficiently rich reward signal. This is not all that different from having to invest effort into designing the architecture of neural networks to accommodate inductive biases and ease nonconvex optimization. In the context of general intelligence, we therefore present the following claim:

Claim 5

The notion of a *fixed a priori reward function* acts counter to open-endedness: assumptions of stationarity and sufficient human foresight cannot be guaranteed to hold in the domains where intelligent agents are expected to operate.

We argue that since open-ended learning is a necessary trait of general intelligence, there cannot be any true fixed-point optimality condition to achieve. It might be possible that certain primitives can be hard-coded as policies over state observations, such as for basic locomotion. However, it is at the very least counter-intuitive that the higher-order cognition for achieving a variety of goals might be expressible as an optimization over a very wide distribution. Instead, we must dismantle the assumption that rewards are stationary and that states (or state-action pairs) accordingly have a pre-defined value.

Indeed, the literature on prospects for continual learning in RL highlights this same point [171]. In this context, general intelligence may be approached as mastering a sequence of MDPs rather than a distribution. Within each one, there is still an assumption of a priori optimality, and this may perhaps make sense for individual tasks which are suitably framed as optimization problems. The more general scenario, however, involves tasks which the user himself may wish to modify/extend/retract as time goes on. Naturally, a framework for task specification should support the modification of a specification without requiring that it be treated as an entirely new one. This could be partially enabled by having reward functions which are compositional, which is explored to a degree in literature on general value functions and scalarized multi-objective RL [350].

This challenge of modifying a priori reward specifications takes on even greater importance in AI safety literature through the core issues of *misspecification* and *alignment* [4], [82], [128], [292]. Even if we assume that humans have a fixed reward function (or distribution or sequence thereof), the challenge of building a powerful general intelligence aligned with that becomes extremely daunting. Inverse reinforcement learning [1] attempts to learn a (parametric) reward function for situations where an analytic description is a poor fit for human concerns. As with the usual a priori case, this reward function is taken to be essentially stationary and so the same concerns arise [129], [130], [195]. Ultimately, this may prove futile, since humans exhibit preferences which do not conform to the Von Neumann-Morgenstern axioms [353] of rationality, and hence cannot be said to possess a stationary utility function [80]. In any case, it is clearly desirable to avoid a formulation of general intelligence which explicitly requires a *proxy* for desired states and would consequently be vulnerable to Goodhart's Law: "When a measure becomes a target, it ceases to be a good measure." [122].

We believe that RL is given credit for much of the work that is actually attributable to engineers through their intelligent design of reward functions and model architectures. More fundamentally, we argue that a framework for general intelligence is better founded if it does not assume that rewards (or more generally, objectives) can ever be fully known, even by the stakeholders. While some things can be optimized to a fixed point, a generally intelligent agent intended to replace human labor would not benefit from such essentially stationary notions. For example, domestic robots that are expected to work well around growing children must be able to accommodate ever-changing constraints and preferences from the family as its youngest members grow up. In fact, it becomes increasingly evident that, in the context of general intelligence, the very concept of 'reward function' ends up creating more problems than it solves. This notion is explored further in Sect. 6.1.

5.2 Sampling: Safety and Efficiency

In Sect. 4.3, we emphasized that *reflection* is a key requirement for knowledge representation. As with our preceding discussion of deep supervised learning, we have identified that feedback mechanisms are a key bottleneck in RL. In the former, we focused on the limitations of iteratively updating neural networks through gradient descent. Here, we discuss an analogous issue which is fundamental to RL, irrespective of reliance on function approximation using deep architectures. In essence, in the MDP formalism the agent receives a reward at each timestep as a result of its state and chosen action, and hence, goals (taken to be regions of the state space) are only expressed via sampling. Note that RL based on *model-predictive control* (MPC) often takes a reward function in closed form, visible to the agent, though this still leaves open the problem of finding its optima.

For arguably the majority of human activities we wish to automate, we take advantage of the fact that we can interpret our desired goals prior to the need to sample any information from the environment. Granted, this requires sufficient world knowledge to make sense of a goal description, but in such a case, this approach confers multiple benefits. We explore ways to make use of this perspective in Sect. 6.1. Here, we highlight the undesirable side effects of a sample-based approach to goal descriptions, of which RL reward feedback is an example. The key point is given in the following claim:

Claim 6

The reliance on sampled rewards means that it is inherently unsafe to train RL systems in the real world. Even if we approached this with very high fidelity simulations, RL would need to exhibit good sample efficiency, which it currently does not.

For humans, a reward function can in principle be made *intensional*: a ‘white-box’ function of state and action. In contrast, the RL procedure incrementally constructs an *extensional* description of the reward, and starts out entirely blind in its sampling. This raises serious concerns for the safety of systems developed in this way. Recent work has emerged which tries to specifically address safe exploration in deep RL [282], [24], [83], [181], but the fundamental issues of using deep neural networks for estimation and having to sample during training remain. Of course, one natural response would be that we can use resettable simulations to safely train agents until we certify them ready to act in the real world. The use of simulation may appear to be a panacea, but this strategy simply delays confronting the issue, analogous to our observations on domain randomization in Sect. 4.2. The primary challenge is the fidelity of the simulation. As a minimum, there is the issue of verisimilitude in the behavior of physical objects, with well-known precision issues [326]. More daunting is the credible simulation of other agents.

Moreover, the *sample efficiency* of deep RL agents is a pressing concern. Sample efficiency is a performance measure which is inversely proportional to the number of samples that the system needs to obtain from the environment in order to achieve success at some objective, e.g., loss, accuracy. Weak sample efficiency is countered in the deep RL community by an increasing reliance on extremely large computational resources to train on vast amounts of data. The associated engineering remedies are hence superficial and reduce visibility of the underlying obstacles.

Deep RL’s most high-profile achievements are some of the best examples to showcase the growing challenge of sample efficiency. OpenAI Five [22] was trained with roughly 40,000 years of compressed real-time experience over the course of ten months. AlphaStar [351] made use of more sophisticated inductive biases and replaced self-play [17] with population-based training [156] but still required roughly 200 years of real-time gameplay per agent, which was parallelized over massive resources in order to achieve a training time of 14 days. In a task of dextrous object manipulation, OpenAI’s Rubik’s Cube agent [247] required 13,000 years of compressed real-time simulation to learn a single task. Note how the single task required a similar amount of experience to that required for a highly complex multi-agent video game for OpenAI Five, clearly showing that the former is nontrivially more challenging for RL to handle.

In the following chapter, we proceed to address the issue of a priori rewards by proposing a framework for *Work on Command*.

Chapter 6
Work on Command: The Case for Generality

Workin' 9 to 5,
What a way to make a living,
Barely gettin' by,
It's all takin' and no giving,
They just use your mind and they never give you credit,
It's enough to drive you crazy if you let it.

Dolly Parton , '9 to 5' [251]

Let us recall that, from a pragmatic perspective, AI is nothing more or less than a *tool* for implementing a new leap in *automation*. This pragmatic perspective on AI is the one that matters in the current world economy, and therefore will necessarily receive primacy for development. While we do acknowledge AI-related notions such as 'artificial life', a significant business case for 'AI as organism' has yet to be demonstrated, and therefore we consider AI that does not directly seek to deliver automation to be out of scope.

As discussed in Sect. 3.2, applications in automation with a known and well-defined target or utility function can be addressed using reinforcement learning—RL allows one to optimize a controller to perform exactly that task with guarantees on speed and accuracy. However, training policies is difficult and time consuming, and this hampers an orthogonal class of applications that require the minimization of the cost/latency incurred by engineer-then-deploy cycles. There is indeed a recurrent need in industry to streamline automation engineering and, in particular, to capitalize on learning processes, recognizing that knowledge acquired from automating one process can be useful for automating the next. Moreover, techno-economic conditions for business sustainability are shifting rapidly, as exemplified by e.g. the Industry 4.0 initiative in Europe [74, 141]: production lines need to absorb increasing levels of flexibility, meaning that processes are irreversibly moving away from the stationarity

J. Swan et al., *The Road to General Intelligence*,
Studies in Computational Intelligence 1049,
https://doi.org/10.1007/978-3-031-08020-3_6

that was once the norm. For example, a system may be tasked with controlling the assembly process of product X, but after a few months X is replaced by new-and-improved product Y, similar to X in some ways. Now we would like to tell the system:

> Stop assembling X immediately, here's a specification of Y, and here are most of your old and a few new effectors. Now start assembling Y, avoiding such and such kinds of defects and wastage.

We use the notion of 'work on command' to refer to the ability of a system to respond, at any time, to changes in task specifications, both positive (goals to be achieved) and negative (constraints to be respected). To be of any use, we include in this notion the ability to leverage all relevant knowledge from prior tasks with little effort. Such leveraging should be non-destructive, i.e., it must be possible to command the system to resume an earlier task, on which it should in general be able to perform at least as well as before.

We posit that performing work on command requires *general* intelligence. Keeping the pragmatic perspective focused on automation engineering, a system's generality can be measured as *the inverse of the cost of its deployment and maintenance in a given range of real-world task/environment spaces*.[1] It will be clear that, the more general an AI system, the better (and cheaper) it will be at performing work on command.

We have seen in Sect. 3.1 that the function of RL is to compile a policy consisting of a direct mapping from environment states to actions. In this paradigm, behavior is the (fixed) computation of a response to a stimulus, best attuned to the assumed specifications of the task and the environment. This notion of 'behavior as a curried planner' offers the benefits of speed and accuracy: once learned, a policy needs little computation to deliver optimal results. However, this comes at the cost of brittleness: adaptation is impossible should the task or environment escape the initial assumptions after deployment. But it does not have to be like this. In cybernetics, system theory, and psychology, behavior is better described as "a control process where actions are performed in order to affect perceptions" as noted by Cisek in his critique of computationalism [49]—see also von Uexküll [191]. Since it is a process, a behavior can be more easily adapted to the variability of its goal, environment, and contingencies. For this reason, we consider processes, *not algorithms*, as the concept most constructive for the synthesis of intelligent systems, as opposed to purely reactive systems—see also "The Irrelevance of Turing Machines to AI" by Sloman [316].

From this perspective, agents have to learn a world model which allows for dynamic planning through simulated trajectories.[2] The value proposition is that this

[1] One might argue that generality is better defined as inversely proportional to computational effort than to monetary cost. However, both involve empirically-determined (but in general, arbitrary) resource expenditure. From the pragmatic, real-world perspective taken here, we find money the more suitable unit of measurement.

[2] As noted before, even if model-based RL leverages world models while learning policies, the final outcome remains a fixed behavior.

model is *task-agnostic*, making it a general-purpose corpus of knowledge that can be reused for achieving a broad variety of goals and shared with other agents to hasten their learning.

The general requirement to perform work on command can be broken down in three main components:

- To handle explicit specifications of goals/constraints and make provision for their possible change.
- To plan dynamically using world models which may change *during* planning.
- To deliver plans and learn anytime, i.e., asynchronously with regards to the system activity.

These requirements add to the challenges for RL discussed previously and require to alter, significantly, its conceptual basis: in the next two sections, we describe how RL can be viewed as an 'artificially constrained' version of an extended framework which we term WoC-RL (WoC for *work on command*). The purpose of WoC-RL is purely pedagogical: it serves to introduce aspects of the subsequently described 'Semantically Closed Learning' from the familiar perspective of RL. In the last section, we depart from the ML algorithmic world view and propose a process-centric perspective on system agency to address the requirement of anytime operation.

6.1 Goals and Constraints

The pedagogical purpose of the WoC-RL exercise is to imagine a version of RL controllers which would be robust to change and capable of adapting their behavior *on the job*. For this reason, WoC-RL makes the idealistic assumption that a controller (hereafter, 'the agent') performs RL endogenously[3] *after deployment*.

The RL procedure operates in accordance with the formulation given in Sect. 3.1. The basic approach can be extended to accommodate additional complexity such as partial observability, stochasticity, and multiple agents. The objective for RL algorithms is to maximize returns, defined as the sum of (discounted) rewards:

$$\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)$$

WoC-RL generalizes the objective of RL by instead seeking to achieve *goals*.[4] This is a more prescriptive notion than *goal-conditioned RL*, an extension of RL in which decisions and value estimates are conditioned on a goal state or embedding thereof (sometimes via universal value function approximation [305]). Goal-conditioned RL is generally used to separate out subtasks and treat them as distinct learning

[3] We leave aside the issue of intrinsic motivation.

[4] Note that WoC-RL is still subject to the limitations described in Claim 4 of Chap. 5.

objectives to hasten the learning of a single complex task [6, 161, 237] or alternatively parameterize a set of related tasks for multitask learning [322, 371]. Regardless, there is no alteration to the reward structure: it is defined a priori and is sampled from every state because while the 'goal' is known, the reward is not.

The goals in WoC-RL effectively replace the notion of reward to become the sole motivation for agent action. Whilst state space regions defined by these goals can indeed be equipped with some associated quantity indicating desirability (which could therefore be said to constitute a 'reward' when reached), the set of state space regions from which this reward can be earned is explicitly specified, thus obviating the need for pointwise sampling of rewards. As such, a WoC-RL agent has access, at any real-valued time t, to the following goal structure $\mathcal{G}$:

$$\mathcal{G}_t = \{\ (S^1, T^1, R^1),\ (S^2, T^2, R^2),\ \ldots\ \} \tag{6.1}$$

where each S^i is (a partial specification of)[5] a state, each T^i is a time interval, and each R^i is a positive or negative real. Having access to $\mathcal{G}_t$, the agent knows which (future) states are promised to give a reward or punishment when visited during a specific time interval. If the current state S_t matches a rewarding state (i.e., there exists $(S^i, T^i, R^i) \in \mathcal{G}_t$ such that $S_t \supseteq S^i$ and $t \in T^i$ and $R^i > 0$), a *goal* is said to be achieved; if the current state matches a punishing state ($R^i < 0$), a *constraint* is said to be violated. The relative values of R^i are only useful to an agent in that they aid prioritization between different goals.

The idea is that the tuples in $\mathcal{G}$ will typically persist over long time spans. Nonetheless, in accordance with the requirement to perform work on command, a user can *at any time* modify $\mathcal{G}$ by adding or deleting state space regions: whether goal states in order to 'command' the agent to achieve things, or forbidden regions in order to 'warn' the agent. Assuming that the user is not perfectly wise in specifying the work, it may happen that $\mathcal{G}$ will contain just one or few goals at any time, but also ever more numerous or detailed constraints, as the user's insight increases.

The expected value of the reward of a WoC-RL agent for taking action a in state s can be defined as in traditional RL [331]:

$$r(s, a) \doteq \mathbb{E}[\ R_t \mid S_{t-1} = s,\ A_{t-1} = a\ ] \tag{6.2}$$

but WoC-RL additionally defines the reward signal R as being completely specified by $\mathcal{G}$:

$$R_t \mid \mathcal{G} \doteq \textstyle\sum[\ R \mid (S, T, R) \in \mathcal{G},\ S_t \supseteq S,\ t \in T\ ] \tag{6.3}$$

where $R_t \mid \mathcal{G}$ reads as 'the reward given the goals'. The long-term *return* as well as the *value function* can again be defined as is traditional in RL [331]; however, neither of these notions serve any purpose in WoC-RL.

The goal structure $\mathcal{G}$ can also be viewed as an explicit description of those states for which a (sparse) reward function would output non-zero values—the only difference

[5] We write $s \supseteq s'$ to mean that s matches s', where s' may be partially specified.

being that this description is explicitly given to the WoC-RL agent without needing to be sampled. Here we see that RL is a restricted case of WoC-RL, namely where a (hypothetical) user would only provide instantaneous goals. That is, RL is obtained if $\mathcal{G}$ is restricted as follows.

$$\forall s, t : \quad \text{either} \quad \mathcal{G}_t = \varnothing \quad \text{or} \quad \exists r \in \mathbb{R} : \mathcal{G}_t = \{(s_t, \{t\}, r)\} \tag{6.4}$$

This stipulates that rewards are immediate and their distribution is not explicitly known a priori. Restricted in this way, WoC-RL turns into traditional RL, as the agent lacks prior information about goals and constraints, never being told ahead of time when or where a reward might be received.

6.2 Planning

WoC-RL changes the optimization problem of RL to that of *planning*: the agent must plan to reach the most valuable state space regions—while avoiding the most punishing ones—as per the goal structure $\mathcal{G}$, while remaining open to changes in $\mathcal{G}$ that can occur at any time.

This framework is inherently *task-agnostic* since the WoC-RL controller must be designed to handle anytime changes in $\mathcal{G}$, whereas RL algorithms typically expect rewards to be stationary. As such, the concept of a policy is not particularly helpful to WoC-RL as it relates to reward. What WoC-RL needs to do is propagate goals through a world model $\mathcal{M}$ in order to generate plans. Such a world model is effectively a learned transition model, ideally of a causal nature, and even more ideally, allowing bidirectional processing. A bidirectional world model allows sensory information to propagate forward as predictions (which, when falsified, can trigger learning), and goals to propagate backward as abductive planning. Nonetheless, even a purely forward world model can be used for deductive planning (disregarding tractability issues), which is easier to express formally, and will be the focus of this section. Abduction will be treated in more detail in Sect. 9.4.

Plan generation can be illustrated by defining an anytime planner:

$$\mathit{work} : \mathit{Time} \rightarrow A$$

that finds the action with the best return according to its current world model $\mathcal{M}$ and goal structure $\mathcal{G}$ (which are retrieved from an implicit storage). Here it is crucial to note that *work is fixed*, i.e., it is not a learnable policy. As prescribed before, $\mathcal{G}$ is changeable by the user but not by the agent. That leaves $\mathcal{M}$—the world model—as *the only adaptable ingredient* of WoC-RL. For pedagogical reasons let us formally specify *work* in terms familiar to the RL community:

$$\mathit{work}(t) \doteq \arg\max_a \; q_*(S_t, a, t \mid \mathcal{G}_t, \mathcal{M}_t) \tag{6.5}$$

where q_* is analogous to the *optimal action-value function* [331], except here it takes not only state s and action a, but also explicitly propagates time t, goals $\mathcal{G}$, and model $\mathcal{M}$:

$$q_*(s, a, t \mid \mathcal{G}, \mathcal{M}) \doteq \\ \sum_{s',t'} p_{\mathcal{M}}(s', t' \mid s, a, t)[(R_{t'} \mid \mathcal{G}) + \max_{a'} q_*(s', a', t' \mid \mathcal{G}, \mathcal{M})]$$

where it is assumed the state transition probability p is included in, or can be derived from, model $\mathcal{M}$.[6] It is important to note that the goal structure $\mathcal{G}$ is propagated *unchanged* by the agent even though it can be changed at any time by the user: if that happens at time t'', $work(t'')$ will start using the new goal structure $\mathcal{G}_{t''}$ containing the user's new goal and constraint specifications. This is in line with the *work-on-command* ideal of an agent performing task(s) exactly until the user tells it to do otherwise.

The purpose of the above formula is to demonstrate that WoC-RL is not concerned with sampling rewards or learning a policy. State quality q_* is well-defined in terms of $\mathcal{G}$ and $\mathcal{M}$ which are given to q_*. Action sequences (plans) output by *work* change if and only if $\mathcal{G}$ or $\mathcal{M}$ changes. Thus, here we see how WoC-RL *avoids conflating knowledge and motivation*, which is crucial for the ability to work on command: whenever motivations are (externally) adapted, the latest knowledge should immediately be brought to bear to act toward those new motivations. In contrast, the classical notion of a policy is as a body of (behavioral) knowledge with respect to one target. This offers no provisions for re-applying the same knowledge for new and changeable targets, such as when automating a succession of related business cases.

Still for pedagogical reasons, we sketch below the algorithmic context in which *work* can be embedded. The simplest setup is a loop which also updates the world model $\mathcal{M}$ based on the results of its predictions (gathered in 'pred' below). Note that changes to the goal structure are supposed to occur externally: $work(t)$ always retrieves the actual $\mathcal{G}_t$.

```
1 pred := ∅;
2 while true do
3     $t$ := WallClock.now();
4     expired := { $s_{t'}$ ∈ pred | $t' \le t$ };
5     update $\mathcal{M}_t$ if 'expired' contains surprises;
6     pred := (pred \ expired) ∪ {$S_t$};
7     pred := pred ∪ ⋃{ forward($\mathcal{M}_t$, $s_{t'}$) | $s_{t'}$ ∈ pred };
8     $a$ := work($t$);
9     execute($a$);
```

[6] Although in RL it is typically the case that $t' = t + 1$, WoC-RL does not require such discrete time-stepping, assuming instead simply that $t' > t$.

The function *forward* yields zero or more predictions. Without going into the detail of *forward*, we remark that its predictions could carry a likelihood, which may be multiplied for each *forward* step—and *forward* may omit predictions below a certain likelihood threshold. In that case, line 7 could be performed in a loop until 'pred' no longer changes.

We see in this algorithm that $\mathcal{M}_t$ is used for making predictions, causing $\mathcal{M}_t$ to be updated whenever they fail. Now we can also see the naivety of *work* as specified in formula (6.5): if $\mathcal{M}_t$ is used forward for prediction making, then it should also be used backward for abductive planning. Such abductive planning can proceed iteratively, in a similar manner to the prediction making as specified in the above algorithm.

Notice also that the actual *work on command*, as embodied in lines 8 and 9, can be parallelized with the prediction making, since these two lines do not depend on the others, only on the objective time. Thus, we see here how we can finally abolish the 'cognitive cycle', namely the perceive–act–update loop in RL, which is inherited from the sense–think–act loop in GOFAI. However, RL still relies on time-stepping in lockstep with the environment, dangerously assuming that the environment does not change while the agent chooses its next action. The way out is to realize that changes in knowledge and motivation do not have to occur at every time-step or at set times and that plans span over larger time horizons.[7] For as long as predictions succeed, and for as long as the goal structure is not modified by the user, the plans that were or are being formed can be assumed to remain valid—to be best of the agent's knowledge. Thus, a WoC-RL agent could be made to learn, predict, and plan *continually* and *asynchronously* from the environment. An asynchronous open-ended continual inference mechanism will be described later in Sect. 7.3.

Finally, this sketch of WoC-RL illustrates the need for *bootstrapping*: if $\mathcal{M}$ and/or $\mathcal{G}$ would start out empty, no work will be performed. In the early learning phase, the user will need to 'seed' $\mathcal{M}$ with minimal world knowledge, and populate $\mathcal{G}$ with simple goals that can be achieved with that little knowledge. The user may have to act as a 'teacher' or 'mentor', populating $\mathcal{G}$ with progressively more difficult goals and constraints.

Here we have described how a pragmatic perspective on general intelligence calls for the engineering of systems that can perform work on command. WoC-RL illustrates what such a system might look like, and how it would differ from 'vanilla' RL. A crucial aspect is the decoupling of knowledge ($\mathcal{M}$) and motivation ($\mathcal{G}$). Although we have detailed what $\mathcal{G}$ may look like technically, we have so far not delved into the learnable $\mathcal{M}$. To attain general machine intelligence, both $\mathcal{G}$ and $\mathcal{M}$ must be framed in a different manner than has been customary throughout the history of AI, from GOFAI to reinforcement learning. The proposed approach is described in the following chapters.

[7] This would require changing the signature of the anytime planner *work* to return a set of timed actions instead of a single one.

6.3 Anytime Operation

We are concerned with the requirements of deliberative intelligence interacting in a society of asynchronous actors (whether organic or synthetic) to achieve multiple goals—temporally overlapping, potentially non-stationary—at arbitrarily long time horizons (Fig. 6.1). We have determined that such agents must plan over world models, and this raises the following issue. If the worst-case execution time (WCET) of consulting a world model exceeds a certain threshold,[8] the system becomes too unresponsive to be effective. This is exacerbated by the fact that, in the setting of open-ended and lifelong learning, world models inevitably grow in size and complexity.

A general solution to this issue is to enforce *granularity* as a property of the world model. Rather than treating the world state as monolithic, a system should be able to devote computation time to consult only the parts of the world model that are relevant to its goals. However, we argue that granularity alone is not enough with regards to world modeling. Even if the WCET of planning is adequate, there is one final necessary conceptual change. From the perspective of RL, inference is paced by the wall clock of the environment. Mathematically, this may not make a difference when compared to the alternative of an internal, asynchronous clock, but again, the context of real-world situated intelligence changes this. Whereas the WCET of RL inference is negligible compared to the time scale at which the environment evolves, the inference WCET of a deliberative agent is potentially subject to high variability, precluding pacing inferences by the wall clock. Instead, inferences must be computed in *anticipation* of the unfolding of world events, and this requires an internal asynchronous clock: to comply to 'anytime' requirements means to coincide asynchronous internal deliberations with world-synchronous goals. Just as there is

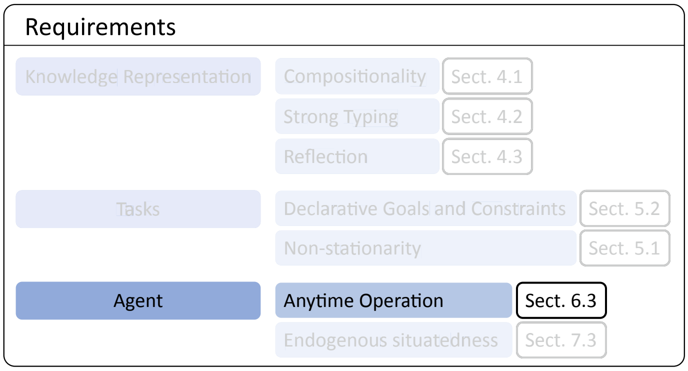


Fig. 6.1 Anytime operation is required to deliver relevant correct action plans on time and asynchronously (relative to system activity). It enables the economically desirable capability of *work on command*

[8] Given a goal, at time t, with a deadline d, the WCET of predicting and planning must be less than $d - t$.

the study of *bounded* rationality [362], there is work on the corresponding concept of *anytime bounded* rationality [27, 150, 241], and we see that this combination is an apt way to unify many of the desirable properties of general intelligence. Unfortunately, common practice in RL is fundamentally incompatible with anytime rationality since it relies both on synchronous coupling with the environment and the concept of 'behavior as a curried planner'. As alluded to previously, anytime operation has not been considered in most demonstrations of RL to date, but we believe that it will become a stubborn hindrance for situated control and multi-agent scenarios.

Part II
Semantically Closed Learning

Chapter 7
Philosophy

The process of induction is the process of assuming the simplest law that can be made to harmonize with our experience. This process, however, has no logical foundation, but only a psychological one.

Wittgenstein, 'Tractatus Logico-Philosophicus', 6.363 [367]

In many respects, machine learning's current concerns are reminiscent of those which heralded the rejection of GOFAI. Since it was evidently not possible to construct a suitably malleable model of the world a priori in terms of rigid logical facts, the solution was surely to induce the required representations, ideally from raw data? Given the challenges previously discussed, the requirement to create robust models is just as pressing today as it was in the early 1990s, when GOFAI was nominally supplanted by the 'Physical Grounding Hypothesis' [35]. In that sense, AI still needs learning algorithms that can do more than 'skim off the surface' of the world they attempt to represent. By this, we mean that knowledge representation should enjoy both robustness and malleability. By 'robustness' we mean Gestalt compositional interpretation in the presence of noise, so that turtles are not considered to be rifles [9] even if their images have some similarity at a local scale. By 'malleability' we mean the ability to envision a range of alternative hypotheses which are compatible with some context. In order to achieve this, we believe that machine learning needs to undergo the same fundamental shift that took place in the philosophy of science in the mid 20th century.

J. Swan et al., *The Road to General Intelligence*,
Studies in Computational Intelligence 1049,
https://doi.org/10.1007/978-3-031-08020-3_7

7.1 The Problem of Machine Induction

The discussion of Chaps. 4 and 5 illustrates that a major concern for DL and RL is the ability to obtain robust generalizations with high sample efficiency.[1] However, the ubiquity of domains with long-tailed distributions is antithetical to the very notion that learning can be predominantly driven via sampling. For example, a widely-respected pioneer of autonomous vehicles [339] has stated:

> *To build a self-driving car that solves 99% of the problems it encounters in everyday driving might take a month. Then there's 1% left. So 1% would still mean that you have a fatal accident every week. One hundredth of one percent would still be completely unacceptable in terms of safety.*

Human drivers, though naturally fallible, are considerably more robust to the combinatorially vast range of situations they might encounter. What change in perspective might be required in order to imbue a system with analogous capabilities?

The essential practice of supervised learning is to tabulate samples of input/output observations and fit a regression model based on a numerical loss function. Likewise, the RL framework optimizes an objective function sampled iteratively from the environment, and deep RL uses deep learning tools for function approximation. As previously observed [212, 344], in terms of the philosophy of science, this is very much in the *empiricist* tradition, in which observations are the primary entities. In DL, treating observations as primary has led to the notion of model induction as a curve-fitting process, independent of the domain of discourse. However, the incorporation of sufficient analytic information can obviate the need for sampling in both cases. As observed by Medawar, 'theories destroy facts' [221]: in order to predict e.g. future planetary motion, we do not need to tabulate the position of all the molecules that compose a celestial body, but rather apply Newton's Laws to a macroscale approximation of its center of gravity [326]. Similarly, under certain conditions a description of the behavior of the simple pendulum can be obtained in closed form [160], demoting empirical sampling of orbits to the role of fitting a distribution to any remaining noise. Indeed, science itself can arguably be characterized as the progressive transformation of 'noise' into 'signal': replacing, insofar as human comprehension permits, uncertainty and nondeterminism with coherent (i.e. relationally-consistent) structure. The resulting structures yield a much stronger notion of 'compression' than expressed by the corresponding use of the term in RL.

Although the empiricist perspective has prevailed since the Renaissance, it was inextricably bound to a deep philosophical problem: the 'Problem of Induction'. The problem asks what firm basis we have to believe that past inferences will continue to be valid, e.g., that the Sun will continue to rise in the morning. Epistemologically, we cannot hypothesize that past distributions of observations will resemble future distributions, since this begs the question. The problem resisted all solution attempts

[1] Addressing this concern is necessary but not sufficient: even if we could obtain robust generalizations via current DL methods, they would be neither granular nor compositional, both of which we subsequently argue for.

until Karl Popper provided one in the mid 20th century [268]. Popper's solution was to show that conclusions could be well-founded without requiring that 'laws' or distributions be somehow Platonically propagated through time. Instead, he argued that although our hypotheses may *be inspired by* observation, they are altogether of a higher-order, being predominantly characterized by their *explanatory power*. As subsequently further developed by Deutsch [63], the key objects of discourse for science are therefore not observations but the *explanatory characteristics* of the hypotheses which they motivate. Hence, consistent with the definition of Sect. 2.2, we can consider a hypothesis as an inter-related system of statements intended to account for empirical observations. The tentative, self-correcting nature of the scientific method means that:

- At any given instant, the collection of statements are not necessarily *entirely* self-consistent (cf. the longstanding inability to reconcile quantum mechanics and general relativity).
- Falsifiability via observation is not the *primary* driver. Although Popper emphasized falsification via observations, a subsequent refinement [186] emphasized that the prevailing hypothesis may not even agree with all observations, provided that it is a rich enough source of alternative hypotheses which potentially could do so. It seems reasonable to consider this to be the spirit underlying the opening Feynman quote.

A famous demonstration that such heuristics are part of common scientific practice is the discrepancy between the predictions of general relativity and the observed rotational velocities of galaxies [246], relativity being a hypothesis which has repeatedly been vindicated in other wide-ranging experiments. Hence, while there is still some global inconsistency between different local models (which will continue to motivate further, hopefully ultimately unifying, hypotheses), this still allows the useful application of local models at the appropriate scale. Over the years, the philosophy of science has conjectured various heuristics for confronting rival hypotheses:

- Parsimony: this heuristic is exemplified by 'Occam's Razor'. However, it must be stressed that this is not merely a domain-independent measure such as is advocated by Algorithmic Information Theory [43], but something that is achieved via reflective interpretation of the hypothesis in order to reconcile causal inconsistencies.
- 'Hard to Vary': This notion was introduced by Deutsch [63]. Hypotheses which are so unconstrained as to permit the generation of many roughly-equivalent alternatives are unlikely to capture the pertinent causal aspects of a situation. Conversely, when a hypothesis which has been preferred to many others generates few or no alternatives, that is an indication that it is a good hypothesis. An initial investigation of the role played by 'hard to vary' heuristics in AI is given by Elton [79].

It is also important to note that, by virtue of compositionality, the notion of hypothesis here is stronger than 'distribution over outcomes' [64]. For example, suppose that six in a hundred patients who are flu sufferers were to hold a crystal and experience a subsequent improvement. Despite statistical significance, an experimenter would not (in the absence of some other deeply compelling reason) subscribe immediately to the notion that the crystal was the cause, because of the end-to-end consistency of existing explanations about how viral infections and crystals actually operate. Such inferences therefore operate at a different level than purely statistical notions, in which claims of causality must anyway be justified in terms of priors known to the domain-aware researcher when they frame the experiment. Hence the researcher here has two privileges that traditional ML lacks: firstly, prior semantic knowledge about the *type* of variables (the displacement of the pendulum bob is measured in radians, the color of the pendulum bob is a property of the material with which it is coated, etc.). Secondly, in the case that prior knowledge (or hypothetical interventions such as Pearl's 'do operator' [254]) does not adequately make the case for causality, the researcher has the potential to clarify further via alternative experiments.

Scientific explanations have proved remarkably effective in describing the world [363]; for example, our understanding of force and motion at the 'human scale' (i.e., between quantum and relativistic) has remained robust since Newton. Most significantly, such understanding is emphatically not in general a *quantitative* function of the causal chain (e.g., some loss or objective function), but is instead dependent on the overall consistency of explanation. 'Consistency' here means not only consistency with respect to empirical observation, but the 'internal consistency' of the entire causal chain described by the hypothesis.

The solution to the 'Problem of Machine Induction' should therefore precisely mirror Popper's solution to the 'Problem of Induction', i.e., to reject empiricism in favor of explanatory power and attempt to afford suitably curious machine learners the same privileges in determining causality as are presently enjoyed only by human experimenters. In the remainder of this chapter, we describe 'Semantically Closed Learning', a framework proposed to support this.

7.2 Semantically Closed Learning (SCL)

Just as the logical expressions of GOFAI could be said to be too 'rigid' with respect to their ability to model complex environments, so the parameter space of DL architectures is too 'loose'. Hence, while it is relatively computationally inexpensive to fit a deep learning model to *almost any* naturally-occurring observations [203], generalization is certainly not assured [374]. It would appear that something intermediate is required, in which there is no requirement for a priori provision of either an arbitrarily complex objective function or an exponentially large collection of rules. To that end, we describe below a set of operations intended to support principled and scalable scientific reasoning. In particular, the 'scientific' aspect of reasoning can be characterized by the gradual progression from an *extensional* representation (i.e., pointwise tabulation of the effects of operators) to an *intensional* one (i.e., representable as an expression tree with a knowable semantic interpretation), as with the analytic description of the pendulum described above.[2] These operations are invoked by a granular inference architecture, a reference version of which is described in the next chapter. We tie these together under the heading of 'Semantically Closed Learning' (SCL), the name having been chosen with reference to the property of 'semantic closure'. In the context of open-ended evolution, the term semantic closure was coined by Pattee [252, 253], who described it as:

> An autonomous closure between (A) the dynamics (physical laws) of the material aspects of an organization and (B) the constraints (syntactic rules) of the symbolic aspects of said organization.

An implementation of such open-ended evolution is given in Clark et al. [51]. They describe a 'Universal Constructor Architecture', in which genomes both contain and are decoded via an *expressor*. The decoding process is stateful (being analogous to gene transcription), and may experience degradation via contact with the environment. Notably, they state:

> The cleanest possible example of demonstrating semantic closure: the genome originally encoded the seed expressor, now it encodes a different expressor, but the genome string itself is not altered at any time, only the *meaning* of the genome string has been altered.

Abstracting from the concrete implementation of Clark et al., we consider a *semantically closed system* to be one equipped with a stateful interpreter, such that:

- The next step in the state trajectory is determined via the application of the interpreter.
- The interpreter is jointly a function of the state trajectory of the system and its interaction with the environment.

This suffices to describe systems capable of open-ended *evolution*, but is closer to the notion of 'AI as organism' than the required one of 'AI as tool'. Hence, we must additionally cater for the achievement of *goals* (self-imposed or otherwise). We therefore define a semantically closed *learner* as a semantically closed system whose interpreter state is adapted, via interaction with the environment, so as to reduce the discrepancy between expected and actual states.

When the discrepancy between actual and expected states is determined via interaction with a 'sufficiently complex' environment (see **Prop-3** of Sect. 7.3, below for more details), the above notion of semantic closure affords situatedness. The interpreter maps from system state to predictions and actions, being subject to repair when predictions are not met. Learning thus takes place as a function of the discrepancy between the actual and desired state of affairs. The important aspect for general intelligence purposes is support for *open-ended learning*. Such learning is considered here as an operationalization of the scientific method, which requires that an agent can generate hypotheses which it can process as first-class entities.

[2] As previously stated in Chap. 4, we of course acknowledge that not all phenomena can be effectively compressed analytically. In certain cases, deep learning is indeed a viable approach, but the anticipation is that this will occur predominantly at the 'leaves' of the expression tree, which are operating directly on raw sensor data.

In one sense, all human beings are scientists (cf. Sloman on ‘Toddler theorems’ [317]). For example, even at a nascent level of cognition, concept formation [262] can be seen as abstracting from an iterated process of hypothesis generation/validation. One may consider higher levels of cognition to be hierarchical, in the sense that they make use of lower-level hypotheses (such as object permanence) as elements. A certain amount of introspection into one’s own problem-solving activity will reveal that higher levels of human reasoning are an ‘end-to-end white-box’ activity: arbitrary (and even novel) features of both a problem and its proposed solutions can be confronted with one another. These features are of course ultimately grounded in experience of the real world [187]. As such, the hypotheses evoked by any confrontation of features are so strongly biased towards the real world that events from the long-tail of impossible/vanishingly unlikely worlds are never even entertained. However, to talk purely in terms of bias as a means of efficient inference is missing a key aspect of human cognition in general, and the scientific method in particular: a hierarchy of compositional representations offers the potential to reason at a much coarser granularity than the micro-inferences from which it was constructed. Therefore, reasoning can be considered to occur in a Domain-Specific Language (DSL) which has been abstracted from the environment by virtue of the ubiquity and robustness of its terms [87, 187]. This is in contrast to the prevailing approach in ML, in which inference is entirely mediated through numerical representations, as biased via some loss or reward function. There, some level of generality is achieved by reducing the notion of feedback to the ‘lowest common denominator’ across problem domains.

It is therefore instructive to consider compositional learning in the context of the historical development of intelligent systems. The early cyberneticists understood that ‘purposeful’ behavior must be mediated via feedback from the goal [299]. Since they were predominantly concerned with analog systems, the feedback and the input signal with which it was combined were commensurate: they could be said to be ‘in the same language’. In the move to digital systems, this essential grounding property is typically lost: feedback is often mediated via a numeric vector that is typically a surjective mapping from some more richly-structured source of information. Useful information is therefore lost at the boundary between the learner and the feedback mechanism. In Sect. 9.4, we describe a compositional mechanism for hybrid symbolic-numeric feedback at arbitrary hierarchical levels that does not intrinsically require any such information loss.

7.3 Baseline Properties of SCL

As detailed subsequently, while the specific choice of expression language for the DSL is rightfully an open-ended research problem, a number of elementary properties are required:

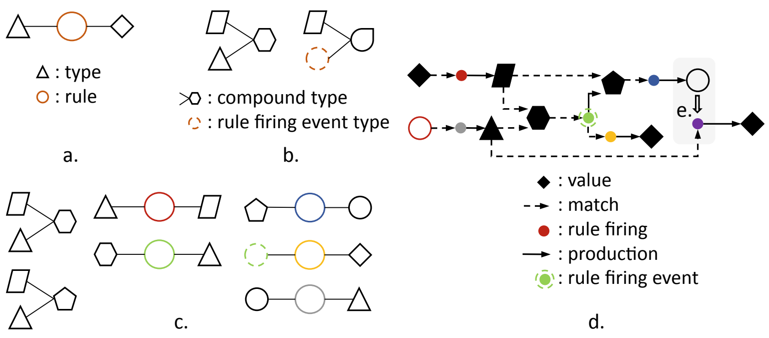


Fig. 7.1 Concepts involved in the reasoning process. **a** A rule (colored circle) implements a relation between typed values (shapes on either side). For forward inference, rules are read left-to-right: an object of one type is transformed into an object of another type via a transfer function. **b** A type may be structured in terms of other types. **c** A repertoire of rules and types. Rules are values and may be composed, such as in the blue and gray rules. Rule firing is also a value (here depicted on the left side of the yellow rule), and so the reasoning process (i.e., the production of inferences) can be reasoned about. **d** A possible unfolding of forward inferences produced by the repertoire. **e** Inferences can produce new rules—they can also produce new types (not depicted)

Support for Strong Typing (Prop-1)

At the most elementary level of representation, labels for state space dimensions can be used to index into code (e.g. stored procedures) and/or data (e.g. ontologies). Building upon this explicit delineation and naming of state space dimensions, a defining property of SCL is the use of a strongly-typed ‘expression language’ [265] which can be used to represent both constrained subregions of the state space and the ‘transfer functions’ that map between such regions (see Fig. 7.1). Types therefore form the basis for a representation language which, at a minimum, constrains inference to compose only commensurate (i.e. type-compatible) objects. Unlike testing, which can only prove the *presence* of errors, the *absence* of errors (and indeed, more general safety guarantees) can be witnessed via strong typing. An elementary such example is the construction of causal models which are physically consistent with respect to dimensional analysis.

In software engineering, such modeling has well-understood safety implications; for example, the bug that led to the loss of NASA’s ‘Mars Climate Orbiter’ in 1999 was due to an invalid conversion between two different dimensional representations of impulse [215]. However, this example only scratches the surface of what can be expressed [263, 265]: the rich body of work in type theory is an ongoing investigation into which aspects of system behavior can be expressed statically, i.e., without requiring actual program execution. For example, certain invariants of the application of transfer functions to subregions of the state space can be modeled via *refinement types*, which use predicates to define which subset of the possible values representable via a tuple actually correspond to valid instances of the type; as pedagogical examples, one can explicitly define the type of all primes or the type of pairs of even

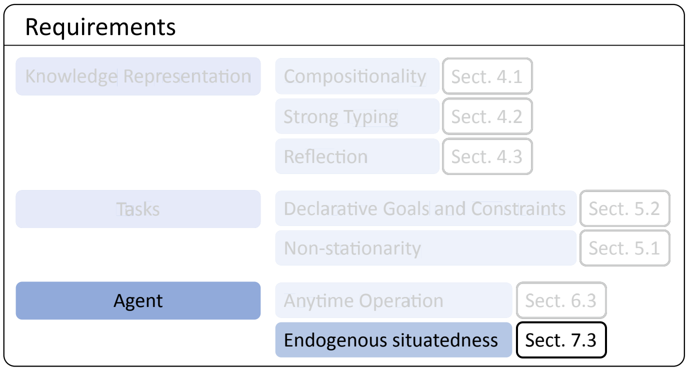


Fig. 7.2 The property of 'endogenous situatedness' imbues an agent with knowledge of its own causal abilities, which includes various proxies for the capabilities of its own reasoning process. This of course also requires a reflective representation and declarative goals and constraints

integers. As well as constructing arbitrary new dimensions of singleton type, it is possible to create new dimensions via other type-theoretic constructions, e.g. tupling or disjoint union of existing types [176]. Since there are intrinsic trade-offs between expressiveness, decidability, and learnability, the specific choice of type system is intentionally left open, being rightfully a matter for continuing research (a concrete example is nonetheless provided in Sect. 10.1). Fortunately, the constructions for compositional inference given in Chap. 9 can be defined in a manner that is agnostic with respect to the underlying type system.

Reflective State Space Representation (Prop-2)

As a minimum, the state space includes the actionables and observables of the environment and/or the system. As discussed in Chap. 6, it must be possible to explicitly declare objectives ('goals') as delineated regions within the state space. While the base dimensions of the state space (corresponding to sensors and actuators for a situated agent) are specified a priori, the representation may also permit the construction of synthetic dimensions (e.g. to denote hidden variables or abstractions, as described below), similar to Drescher's 'synthetic items' [71]. As discussed under 'Work on Command', the property of reflection obviates the need for sampling of rewards, and allows for dynamic changes to goal specification, since state space constraints are available to the agent. A reflective state space is also key for enabling the creation of new types through abstraction.

Endogenous Situatedness (Prop-3)

A system has 'grounded meaning' if its symbols are a context-sensitive function of the system's experience, a property typically lacked by GOFAI systems. In his seminal work, Harnad considers a system to be grounded [134] if reasoning is driven by sensory inputs (or invariants induced thereof) from the real world. Wang [355] argues that this notion of 'real world' can be relaxed to apply to 'real-time operation in any complex, uncertain dynamic environment', provided that (1) symbol interpretation is contextually driven by information obtained from the environment and (2) information is updated via a feedback loop that includes the action of the system. The reflective state space representation of end-to-end hypothesis chains of **Prop-2** thus suffices for the system to be *situated* in its environment. That is, the mapping between grounded sensors and effectors proceeds via a world model in which feedback from effectors is reflectively inspectable.

However, there is a yet stronger notion of 'situated' which more closely captures a system's causal capabilities: being *endogenously* situated. This arises from the observation that *"an organism's own patterns [. . .] are also stimuli"* [97]. A system is therefore endogenously situated when (at least some of) its *internal* representations[3] should also be considered part of the environment in which the system operates, and these endogenous stimuli are given meaning via their ultimate participation in causal sensor-effector chains.

Open-Ended Continual Granular Inference (Prop-4)

As discussed, our pragmatic definition of general intelligence emphasizes the need for *flexibility of response*. This requires that an intelligent system avoids the 'perceive–act–update' cycle of traditional RL and GOFAI, in which it is effectively assumed that the system and the world progress in lockstep. Since system deliberation time will necessarily increase with environment and task complexity, the lockstep approach will not scale. As per previous work on time-bounded inference [242, 243] the alternative is to perform many simultaneous inferences of smaller scope, each inference having a WCET (worst-case execution time) that is both small and bounded—hence 'granular'.

In some of our previous work [242], scheduling based on dynamic priorities is used to explore a vast number of lines of reasoning in parallel, while retaining flexibility and responsiveness. As described in more detail in the next Chapter, by virtue of scheduling over granular inferences, *attention* is then an emergent process: the analog of attention-weights [16] are the priorities, which are dynamically updated as a function of the expected value of inference chains.

[3] For example, projections from SCL's internal 'train of thought' state trajectory, monitoring of resource usage, etc., as described in Chap. 8.

7.4 High-Level Inference Mechanisms of SCL

Building on the support for strong typing (**Prop-1**) and reflective state space representation (**Prop-2**), SCL makes use of four methods of compound inference: hypothesis generation, abduction, abstraction, and analogy. All inference steps in SCL can be considered to be the application of some rule $r : A \rightarrow B$, for types A and B. If no such rule exists, then it is necessary to synthesize it, as described in more detail in Chap. 10. This synthesis process may involve any combination of the following:

Abstraction

For purposes of SCL, abstraction is considered to be the process of factorizing commonality from two or more hypotheses. One can view this factorization as a parametric or 'partially instantiated' hypothesis. A suitable choice of parameters may allow the motivating hypotheses to be (at least approximately) recovered. In a numerical domain, approaches such as PCA/SVD [25] compress a set of empirical observations into a basis set from which observations can be reconstructed via a specific weighting of the basis vectors. In SCL, the methods used for decomposition and reconstruction of the state space must be applicable at the symbolic level of expression trees, as well as for any numerical expressions at their leaves. In Sect. 9.3, we describe one possible compositional mechanism for abstraction.

Hypothesis Generation

This is the means by which salient hypotheses are generated. Hypothesis generation interprets an existing hypothesis to yield a new one intended to have fewer relational inconsistencies. It is a broader notion than the counterfactual reasoning conducted using structural causal models (SCM), since rather than merely taking different actions, it considers the overall consistency of alternative models. Informally, this can be seen as the imposition of semantic/pragmatic constraints on expressions in a generative grammar. As an example from the domain of natural language, the famous phrase "*Colorless green ideas sleep furiously*" is syntactically valid, but not semantically consistent: neither color nor the ability to sleep are properties typically associated with ideas. The semantic inconsistency here is immediately obvious to the human reader, but in general an artificial learning system must use feedback from the environment to discover any semantic inconsistencies in its interpretation. Hypothesis generation is described in detail in Sect. 9.2.

Analogical Reasoning

Analogy has been argued to be the *"core of cognition"* [146]. It can be considered as a generative mechanism that factors out a common 'blend' [84] between two situations. There is considerable literature on cognitive and computational models of analogy: in-depth surveys can be found in Genter and Forbus [106] and Prade and Richard [271]. Analogy is generally considered to be either *predictive* or *proportional*. Predictive analogy is concerned with inferring properties of a target object as a function of its similarity to a source object (e.g. the well-known association between the orbits of planets in the solar system and electron shells in the Rutherford model of the atom). Study of proportional analogy extends at least as far back as Aristotle [52]. A proportional analogy problem, denoted:

```
A : B  ::  C : D
```

is concerned with finding D such that D is to C as B is to A. For example, "gills are to fish as what is to mammals" is notated as:

```
fish : gills  ::  mammals :  ???
```

In Sect. 9.3, we describe one possible computational approach to proportional analogy.

Abduction

By virtue of the reflective representation, it is possible to perform inverse inference. A hypothesis can thereby be updated *directly* by working backwards from its effects, rather than via the indirection of a sampled objective function, which was the primary objection raised in both Sects. 4.3 and 5.2.

With a bidirectional reasoning process (illustrated in Fig. 7.3) it is possible to 'backpropagate' actions directly along the hypothesis chain from effects ('failure of

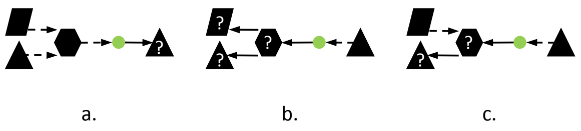


Fig. 7.3 Bidirectional rules. Rules support both induction and abduction; depending on their denotational semantics, their inputs and outputs (marked '?') are ascribed particular meanings. **a** Induction: the output can be a prediction or a timeless entailment (e.g., an instance of a subtyping relation). The inputs may be (counter)factual (e.g., sensory inputs or absence thereof), induced or abducted. **b** Abduction: the input can be a goal, an assumption, or a (counter)fact. The outputs can be subgoals, subassumptions, or timeless premises; they are not necessarily unique. **c** The choice of outputs is constrained by an input

an upside-down table to provide support') to counterfactuals ('*if* the table were turned the other way up ...'). In DL and RL, the 'representation language' is untyped and noncompositional, so this kind of direct modification of hypotheses is not possible. In Sect. 9.4, we describe a compositional mechanism for abduction.

7.5 Intrinsic Motivation and Unsupervised Learning

In contrast to the a priori problem formulation required for supervised learning, the scientific method is an *iterative process* of problem formulation and solving. Such an iterative approach performs both supervised and unsupervised learning, the former corresponding to the meeting of objectives supplied a priori, the latter being the search for more compelling hypotheses, potentially via new experiments. In this wider framework, hypotheses have the non-monotonic property of the scientific method itself, i.e., they are potentially falsifiable by subsequent observation or experiment.

The aspiring robot scientist must therefore decide how to interleave the processes of observation and hypothesis generation. Prior art on this in the (simply-stated but open-ended) domain of number sequence extrapolation is Hofstadter et al.'s *Seek-Whence* [144], which decides when to take further samples as a function of the consistency of its hypothesis. In a more general setting, the self-modifying Power-Play framework searches a combined task and solver space, until it finds a solver that can solve all previously learned tasks [307, 323]. In more recent work, Lara-Dammer et al. [190] induce invariants in a 'molecular dynamics' microdomain in a psychologically-credible manner.

In particular, our chosen definition of general intelligence acknowledges that resources (compute, expected solution time, relevant inputs, etc.) are finitely bounded. At the topmost level, the corresponding resource-bounded framework for the scientific method is simple: within supplied constraints, devote resources to finding good hypotheses, balancing the predicted merits of hypothesis refinement against the ability of the refinement to further distinguish signal from noise. The presence of such bounds is an intrinsic guard against the kind of 'pathologically mechanical' behaviors that one might expect from algorithms which do not incorporate finite concerns about their own operation, as detailed further in the next chapter.

Chapter 8
Architecture

ACHILLES: The profession of Anteater would seem to be synonymous with being an expert on ant colonies.
ANTEATER: I beg your pardon. 'Anteater' is not my profession; it is my species. By profession, I am a colony surgeon. I specialize in correcting nervous disorders of the colony by the technique of surgical removal.

Hofstadter, 'Gödel, Escher, Bach' [147]

Machine learning excels at inducing mappings from data, but struggles to induce causal hierarchies. In contrast, symbolic reasoning (in particular, when considered as an *expression language*) can represent any form of domain knowledge and can index into code or data via pattern matching.[1] Evidently, reasoning and learning must be robust to both the variability of inputs and the reliability of prior knowledge, learned or imparted. In that regard, Marcus has argued extensively for neuro-symbolic hybrids [210, 213, 214], advocating the complementarity of distributed representations ('neuro') and qualitative localist causal knowledge (symbolic); see also d'Avila Garcez [59]. We explain in this chapter how SCL defines a framework with equivalent goals: although not explicitly 'neural' in operation, the dynamic attention mechanism can be considered to play an analogous role to that of neural connection weights, although perhaps a better guiding metaphor than homogeneous neurons is the 'structural stigmergy' [179] of ant colonies [70, 146]. We then proceed to present a reference system architecture for SCL: its purpose is of course to show how SCL can be realized *in silico* and, by design, to provide sufficient guarantees for its claims (open-ended learning, anytime operation, grounded reasoning, etc.) at the operational level.

[1] Of course, heuristic synthesis of symbolic expressions (e.g. as in Genetic Programming [180]) has been practiced for decades, but has never really been considered as 'mainstream' ML.

J. Swan et al., *The Road to General Intelligence*,
Studies in Computational Intelligence 1049,
https://doi.org/10.1007/978-3-031-08020-3_8

8.1 SCL as a Distributed/Localist Hybrid

Numerous authors have indeed attempted the integration of distributed and localist representations for more than twenty years. However, depending on the original focus of inquiry (reasoning, control theory, ML, or other), 'integration' can serve a rather broad variety of purposes. We now list a few exemplars of these to give some perspective to the purpose of hybridization in SCL; a broader survey can be found in Bharadhwaj et al. [23].

The AKIRA hybrid architecture addresses control rather than learning [257] and is designed around concurrent code fragments. These fragments are organized in a weighted dynamic network of dependencies, competing for budget-constrained computing power on the basis of their expected outcome (goal satisfaction). Focusing on learning instead of control, the DUAL/AMBR architecture [177, 178] controlled symbolic reasoning via spreading activation based on numeric truth values across a network of causal and structural relations, in a manner reminiscent of Copycat [146]. Clarion, another hybrid approach [327, 328], layered symbolic reasoning on top of neural networks, with the aim of enabling top-down and bottom-up learning processes. These were ultimately based on RL, thus imposing, architecture-wide, the fundamental limitations described in previous chapters. In pursuance of another objective, the Sigma architecture attempts to define intelligence in a principled manner [295]: the authors claim 'grand unification', and 'generic cognition' based on graphical models. It is indeed not impossible to think that such a computational substrate, pending further significant work, might become the lingua franca of representation processes able to transcend levels of abstraction. However, the authors have so far limited themselves to the reimplementation of established concepts such as episodic memory [297], RL [296], and a 'standard' model of cognition based on the 'sense–think–act' loop [298].

More recently, the majority of ongoing research on hybrids is unsurprisingly ML-centric, attempting to remedy some of the inadequacies of deep neural networks for reasoning applications. For example, some recurrent neural networks such as LSTM, although designed for sequential/temporal inference, face difficulties with learning long-term dependencies, mainly because their memory consists essentially of compressed (and degradable) input sequences. The Differentiable Neural Computer (DNC) [124] alleviates this problem by coupling the network to an external symbolic memory (other work uses different types of memory augmentation [7, 166, 260]), but does so at the cost of magnifying other shortcomings of DL: the DNC is notoriously harder to train than LSTM (sample inefficiency), and its applicability to arbitrary tasks appears to depend even more than before on the architecture employed and its initial choice of parameters (brittleness). Several improvements have since been proposed [55, 92, 248] and have addressed some of the performance issues but less so the issue of applicability. As of today, the DNC performs reasonably well on latching, copying, associative recall, and simple state machines; the general framework of differentiable computing is preserved but the capabilities of the DNC remain very remote compared to the requirements for general reasoning (abstracting, analogy making, planning, etc.). Other approaches have made the opposite trade-off, namely that of sacrificing end-to-end differentiability to accommodate more powerful reasoning substrates. They proceed by inserting hand-crafted concepts directly into the RL framework in various forms, e.g. conceptual graphs [152], algorithmic primitives [138, 193], Drescher-style schemata [71, 164], or ontologies and associated auto-encoders [105]. In all of these cases, the agent learns a concept's extension and operational semantics, thus enabling planning via various forms of probabilistic inference. However, as recent research shows, hand-crafting of explicit knowledge structures can be entirely avoided in some cases (so far, mostly problems that can be addressed by direct visual attention). Notably, the PrediNet neural architecture [311], when subjected to an appropriate curriculum, is capable of acquiring some representations with explicit relational and propositional structure. With PrediNet, the neural substrate is used only for learning and does not commit to any specific form of knowledge exploitation: this can be carried out either symbolically (e.g. using predicate calculus, temporal logic, etc.) or, more speculatively, via differentiable 'reasoning models' [69, 225, 294].

These attempts at 'getting the best of symbolic AI and ML' proceed quite literally, essentially reimplementing reasoning (and the necessary supporting representations) in the terms and components of machine learning. We proceed differently. Three aspects of SCL are of particular relevance to what is expected from the reconciliation of ML and symbolic AI:

- Strong typing.
- Fine-grained, open-ended, continual, and compositional inference.
- Emergent resource-aware and goal-directed attention.

We claim that these principles combine the strengths of both approaches: whilst SCL can be provided with prior domain knowledge in any desired form, it is not subject to the problems which plagued GOFAI, since the ability to reflectively reason at the type level allows the *sustained* and *progressive* learning of invariants from the environment. In SCL, learning and planning do not need to be reconciled. By design, both learning and planning are side-effects of the same reasoning process which unfolds in response to the pressure of external goals and constraints over limited knowledge and resources (e.g. inputs, time, computational power and memory, energy, physical agency, etc.). The research cited above pursues the objective of end-to-end representability and actionability of structured knowledge; to this, we add the (orthogonal) requirement of end-to-end controllability, i.e., self-referential goal-directed resource allocation. In the SCL framework, the duality between distributed and local does not concern the representation of functional knowledge (world models, goals, etc. are already unified), but rather the representation of cognition itself. For reasons explained below, we use distributed representations for controlling the reasoning process, itself explicitly represented in the corpus of world knowledge to describe the state of the 'system-in-the-world' (the 'endogenous situatedness' of **Prop-3** in Sect. 7.3); Wang's Non-Axiomatic Reasoning System (NARS) [358] follows a similar approach.

For open-ended learning, inputs may originate from outside the distribution from which knowledge was initially learned. Yet *progress* must nonetheless be sustained, both in terms of continual learning and action. ‘Falling outside the system’s comfort zone’ is not sufficient reason for invalidating acquired knowledge. This would amount to ‘stop, erase, and re-learn’, similar to the RL ‘learn-then-deploy’ procedure that, as we have seen, runs counter the desired property of anytime operation. In fact, keeping going might just be the right thing to do: in case foreign inputs happen to be in a semantic continuum with prior distributions, extrapolation would be correct, the system’s activity would carry on while vindicating its knowledge over increasingly broad scopes. Of course, in case of a semantic discontinuity, prior knowledge would produce faulty extrapolations and the system might fail to meet its goals. In that respect, the system has to perform two activities, continually and on a case-by-case basis (possibly concurrently). The first consists of extending an initial distribution with foreign inputs and vindicating the current related knowledge. The second is to learn new knowledge from a seemingly novel distribution—the system must also surmise explicit constraints on the relevance of the initial one to guard it against further unwarranted use. This can be achieved by assessing the *degree* to which the learned patterns match the new inputs and propagating the discrepancies across the possible consequences at higher levels of abstraction in the knowledge hierarchy—for these are ‘battle hardened’ oracles: by construction, they are broader, more reliable and, critically, change less frequently than the lower layers of the hierarchy.

SCL accommodates control heuristics for which truth values are not axiomatic but instead are assessed up to a certain degree (certainty is asymptotic, a tenet we share with Wang [356]). In this approach, truth values are not static, they unfold over time: they are multiplied at each step of inferencing and are also updated whenever new (counter-)evidences become known [241]. At the conceptual level, dynamic numeric truth values thus reflect, quantitatively, inference composition. Interpreted, at the operational level, as the reliability of inference chains, truth values allow to compute the worthiness of resource expenditure (i.e., for further ramifying inference chains) with regards to the goals at hand. This forms the substrate from which goal-directed attention emerges to control the reasoning process, as we will see in the next section.

8.2 Reference Architecture

The ultimate purpose of SCL is to achieve end-to-end consistency of the feedback loops.[2] This is enabled by *endogenous situatedness* as per **Prop-3** of Sect. 7.3 which (1) preserves, anytime, both relevance and correctness, despite the cognitive load possibly overwhelming the inevitably limited computational resources, while (2)

[2] Feedback loops are manifested both at the macro-scale, coupling system and environment, and at the micro-scale (i.e. intra-system), coupling inference chains—formally, the *composition of lenses*, defined in Sect. 9.4.

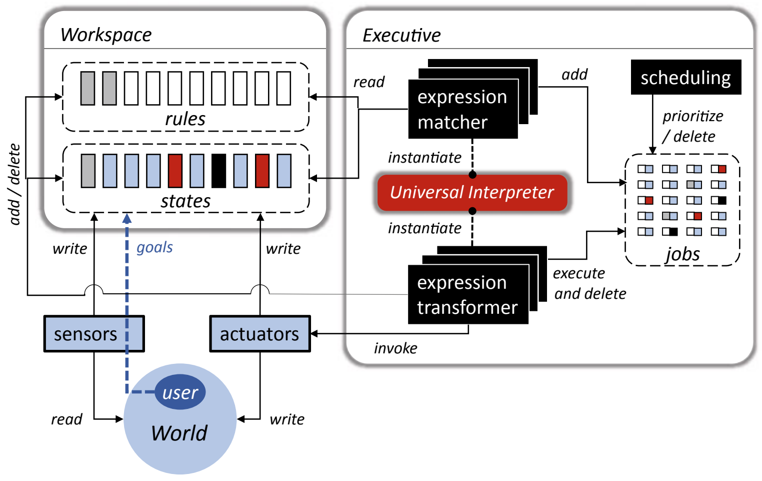


Fig. 8.1 SCL reference system architecture for end-to-end semantic closure. The executive orchestrates a number of asynchronous processes (in black) and provides interfaces to external sensing/ actuation data streams (also asynchronous). Scheduling is a resource-aware goal-directed *emergent process*. Rules and states in grey denote axioms. The picture does not distinguish between state modalities—past, present, assumed, predicted, or desired. States in blue pertain to the world, states in red reflect the reasoning process, and states of the executive (memory usage, energy consumption, etc.) are represented in black. The write operation performed by actuators on the workspace carry efference copies; see text for details

deriving such adaptive behavior from the goals at hand, in light of (3) explicit representations of the physical and cognitive state of the system.

Figure 8.1 gives an overview of a reference system architecture. The architecture consists of two main subsystems: a workspace and an executive. The *workspace* contains learned relational knowledge (implemented by rules) and a representation of both world and system states—be they actual (past or present), hypothetical (assumed or predicted), or desired (goals and constraints). Regarding ‘relational knowledge’, we move away from the vocabulary of causality, which is framed in terms of elementary changes to the ‘wiring diagram’ of hypothesis chains, as per Pearl’s ‘do operator’. Instead, in common with the emerging discipline of behavioural control [365] (see Sect. 11.2.3), we adopt the *relational* perspective, which is concerned with the invariants propagated by a relation, as per the notion of *contract* [245]. This offers a more prescriptive framework that is therefore better suited to describing state spaces, particularly those where rules are partially instantiated (e.g. input terms contain wildcards or constraints)—see Sect. 10.1.

The consistency of the workspace is maintained by the *executive* which, besides providing interfaces to sensors and actuators, consists essentially of a scheduler and multiple instances of a universal interpreter. In broad terms, the architecture can be considered a massive fine-grained *production system* [288, 300], in which huge numbers of rules fire concurrently and asynchronously. Some small fraction of the workspace rules may be provided a priori while the vast majority are produced and maintained dynamically. In addition to rules, the workspace contains states—axioms are also accommodated. Each inferred state has an associated time horizon and an estimate of its likelihood. States qualify both the world, the deliberations of the system ('trains of thought'), and the system itself *embodied* in the world (manifested as memory expenditure, performance profile, physical agency and integrity, etc.).

Sensors and actuators constitute the physical interface of the system to the world, and as such, must be amenable to modeling. For this reason, actuators write *efference copies* in the workspace. An efference copy is an explicit state which encodes the action that was actually executed (by the 'body') in response of the request (by the 'mind') for a desired action: the error between the two allows the system to model the contingencies of its embodiment, i.e., capabilities, limitations and costs (including response latency, energy expenditure, wear and tear, etc.) in a feedback loop generalizing the biologically plausible model of motor control proposed by Wolpert [136]. Errors are useful to learn 'body models' at the edge—'inside-out'—, but not only: there is an inherent dual (and equally important) way to make sense of errors. When body models are well established (i.e., vindicated by significant experience), errors change their meaning: they then signify hidden world states. To take the example used by Wolpert, assuming a carton of milk is full, one will exert a rather strong torque on one's limbs to lift it, only to overshoot the intended end location in case the carton turns out to be empty. In other words, a misaligned efference copy defeats an invalid assumption (a case of abduction). The symmetric feedback loop (i.e. the one pertaining to sensors) is similar, the main distinction being that the inherent duality of errors is operationally *mirrored*. Sensing errors are signals for learning world models at the edge ('outside-in') in response to the invalidation of predictions (a case of deduction). Conversely, when the stationarity of world models is well established (i.e. the reliability of world models is predicted reliably, vindicated by significant experience) sensing errors also change their meaning: they then signify the failure of the sensing apparatus.

Each rule acts as *match-and-transform* schema for typed expressions, encoding patterns that denote subregions S_1, S_2 of a state space S, together with a transfer function $\tau : S_1 \rightarrow S_2$. The state space is dynamic: additional dimensions may be synthesized (e.g. via abstraction) at any time by some rules and subsequently also be available for matching and transformation; least recently significant dimensions may conversely be deleted. *Matching* binds values (states, rules or transformation events) to rules, whereas *transforming* combines several such bindings to produce new values (along with a transformation event as a side effect). The allocation of resources for transforming available bindings is prioritized by the scheduler; the most promising bindings will receive computational attention first.

Matching and transformation of SCL expressions is performed via instantiations of a *universal interpreter*. Paths through the state space are composed via the application of interpreter instances and compound SCL inference methods (as introduced in Chap. 7 and discussed in further detail in Sects. 9.2–9.4) to produce inferences (deductive, abductive, or analogical) and state-space dimensions. This interpretation is explicitly compositional and imposes a denotational semantics on expressions.

The unfolding operation of universal interpreter application bears witness to the system's internal state of deliberation: such 'trains of thought' are considered first-class citizens and are duly recorded in the workspace.

The learned ruleset constitutes a fine-grained representation of the entire transition function of the world model. In contrast, axiomatic rules have a different purpose: to implement heuristics for rule and type construction and maintenance. For example, some rules seek to "carve Nature at its joints" [127], by identifying 'surprising' world states such as the failure of a prediction or the unpredicted achievement of a goal. In either case, corrective rule synthesis is applied to the ruleset: this can be considered to impose learned pragmatics on the default denotational semantics. Conversely, the *absence* of surprise vindicates parts of the world model and the rules that implement them: these rules will be deemed more reliable than others that see repeated failures. For completeness, other axiomatic rules include seeking out candidate patterns as a basis for constructing abstractions and analogies, as well as identifying opportunities for generating improved hypotheses. Rule synthesis is addressed extensively in Chap. 10.

Computing resources are inevitably limited and a system must allocate these wisely to cater to arbitrary influxes of inputs and goals. The reference architecture decouples matching from transformation and treats them accordingly as two distinct classes of processes. Matching processes produces *requests* for transformation, termed 'jobs', each of these being associated with a quantitative estimate of its utility with respect to all goals at hand, system-wide. This estimate is based on three main factors: (1) the likelihood of the matched value, which is a function of the reliability of the (chain of) rules that produced it; (2) the reliability of the matching rule; and (3) the relative importance of the goals that may be served (or opposed) by the result of the job. This is revised continually as premises, rules, and goals vary not only in quality (the geometry of the subregions they occupy in the state space) but also quantitatively as per their reliability and relevance. Estimates of utility are primarily used for prioritizing jobs for execution, i.e. in fine, to schedule the execution of transformations competing for resources.

The role of the scheduler in the SCL architecture is merely to re-order jobs continually according to their priorities. For example, previously unattended jobs may jump ahead of new ones as their prospective benefits become apparent, whereas hopelessly unpromising jobs are eventually deleted. This particular scheduling design confers three essential benefits. First of all, it *avoids the scalability issues* inherent in standard job-scheduling theory. The scheduler does not perform explicit allocation but instead slows down relatively less beneficial inference chains, exploiting the fine granularity thereof to its full extent. In the standard acceptance of the term, this is hardly a scheduler at all: SCL scheduling is indeed more a distributed emergent *process* than it is an algorithm. Second, this particular design enforces a fundamental property of the system architecture, namely *endogenous situatedness*. It achieves this by imposing a semantically-grounded control over the execution of the expression transformers: control is based on up-to-date predictions of functional costs and benefits, and balances the cognitive load for the best use of time (deadlines) and available (possibly varying) computing power. Last but not least, semantically-grounded control implements an *implicit attentional process*: centralized attention algorithms are notorious not only for hindering scalability but also for opposing anytime responsiveness. Instead, an SCL system keeps the cost of attention constant by amortizing the computation of scheduling priorities over the continual transformations of expressions.

The embodiment of the system also imposes limits on the size of its workspace. Although the reference architecture does not impose any specific heuristics to keep the growth of the workspace in check, we mention here a few possibilities. A rule can be evicted on the basis of its reliability: a general trend downwards indicates irrelevance and warrants deletion. There are also other reasons why an otherwise reliable rule can become irrelevant, for example, a temporary change of the stationary regime of the environment or a change in the agent's mission. In such cases, deletion should be avoided (the system may need to revert to previous conditions) and the incriminated rule shall instead be swapped out to some larger auxiliary storage at the cost of incurring extra latency upon its recall (swap in). Whenever a chain of abduction halts before reaching its goal a swap-in request is issued, taking the missing rule signature as its argument. In case no rule is returned, then the system would have found a 'gap' in the world model structure and trigger learning. Inferred states, as we have seen, are qualified by their likelihood; they can be furthermore qualified by their utility: the utility of a prediction increases when its target state matches that of a goal and vice-versa. Low values for both the likelihood and utility can meaningfully trigger the eviction of an inference. Finally, possible heuristics for evicting observed states from the workspace include LRU (least recently used), age, and number of references.

Chapter 9
A Compositional Framework

A couple in love walking along the banks of the Seine are, in real fact, a couple in love walking along the banks of the Seine, not mere particles in motion.

Stewart A. Kauffman [168]

The system architecture presented in Chap. 8 controls (i.e., sustains and constrains) the invocation of the inference methods introduced in Chap. 7. In this chapter, we describe the methods of higher level inference in more detail. There is increasing conviction that methods of *category theory* are well-suited for providing generic descriptions for cognitive science, general intelligence [258], and control [42], the latter being newly christened as 'categorical cybernetics'. In this chapter, we describe how to leverage the power of selected category-theoretic constructions to realize SCL operations in a compositional manner.

9.1 Categorical Cybernetics

Category theory has served as a formidable unifying mechanism in mathematics. It was devised in the 1940s by Eilenberg and MacLane [209] in order to provide a higher-order vocabulary for algebraic topology and defines a principled setting for the study of structurally-informed transformations. Amongst applied category theorists there is increasing interest in machine learning applications, and it is becoming apparent (e.g. [78, 90, 318]) that much can be done to unify and generalize existing methods. Following a brief overview of category theory, we describe how SCL operations may be implemented in terms of specific category-theoretic constructions. Below, we describe the essential concepts; more detail is available in various excellent texts (e.g. [196, 264, 290]). Mathematical approaches to formalizing compositionality can be broadly divided into 'syntactic' and 'semantic'. While syntactic

J. Swan et al., *The Road to General Intelligence*,
Studies in Computational Intelligence 1049,
https://doi.org/10.1007/978-3-031-08020-3_9

approaches such as those in linguistics [361] and process algebra [224] might be better known, the semantic approach via category theory has become increasingly popular in diverse fields due to its flexibility and mathematical elegance [89].

A category is a two-sorted algebraic structure rather like a graph, consisting of ‘objects’ and ‘morphisms’: every morphism has a source object and a target object, written $f : X \to Y$. In addition to the graph-like structure is a way to compose morphisms: given any morphisms $f : X \to Y$ and $g : Y \to Z$ with the target of one agreeing with the source of the other, there is a composite morphism $fg : X \to Z$ (typically written $g \circ f$). This must satisfy the ‘associativity property’ familiar from algebraic structures such as groups, i.e., given three composable morphisms f, g, h the two different ways of composing them must agree: $(fg)h = f(gh)$. Every object must also have an assigned ‘identity morphism’ $1_X : X \to X$, and they must act as the identity element for composition: $1_X f = f = f 1_Y$ for all $f : X \to Y$.

There are many examples of categories, and we will name only a handful:

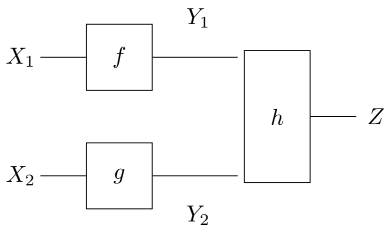
1. The category **Set** of sets, whose objects are sets and morphisms are functions.
2. The category **FinVec** of finite-dimensional vector spaces, whose objects are finite dimensional vector spaces and morphisms are linear maps.
3. The category **Rel** of relations, whose objects are sets and morphisms are binary relations.
4. For any graph G, there is the ‘free category on G’, whose objects are nodes of G and morphisms are paths in G. Composition is concatenation of paths, and identity morphisms are paths of length zero.
5. Any monoid M can be viewed as a category with a single object $*$, where every element $m \in M$ is viewed as a morphism $m : * \to *$.

Of particular relevance is the fact that typed programming languages (e.g. the simply-typed lambda-calculus [47] or intuitionistic type theory [217]) give rise to corresponding categories: it is possible[1] to consider a category with types as objects and functions as morphisms [54].

When studying compositionality, it is typical to work in the context of ‘monoidal categories’, which have additional structure: there is a monoid-like structure on objects, with a binary operation $\otimes$ and a unit I, in a way that is also compatible with morphisms, so if $f : X_1 \to Y_1$ and $g : X_2 \to Y_2$ are morphisms, then so is $f \otimes g : X_1 \otimes X_2 \to Y_1 \otimes Y_2$, satisfying various laws. A category typically has several different monoidal structures. For example, in the category of sets we could take $\otimes$ to be Cartesian product of sets (whose unit is a 1-element set) or disjoint union of sets (whose unit is the empty set). In the category of finite-dimensional vector spaces we could take $\otimes$ to be the direct product (whose unit is the 0-dimensional space) or the tensor product (whose unit is the 1-dimensional space).

Morphisms in a monoidal category are often represented using the graphical notation of *string diagrams* [216]. For example, if we have morphisms $f : X_1 \to Y_1$, $g : X_2 \to Y_2$ and $h : Y_1 \otimes Y_2 \to Z$, then the composite morphism $(f \otimes g)h : X_1 \otimes X_2 \to Z$ is represented by the diagram:

[1] Modulo certain technical considerations.



The other basic concept required is a ‘functor’, which is a structure-preserving map between categories. If $\mathcal{C}$ and $\mathcal{D}$ are categories than a functor $F : \mathcal{C} \to \mathcal{D}$ is an assignment sending every object X in $\mathcal{C}$ to an object $F(X)$ in $\mathcal{D}$, and every morphism $f : X \to Y$ in $\mathcal{C}$ to a morphism $F(f) : F(X) \to F(Y)$ in $\mathcal{D}$, in a way that preserves identities and composition. If our categories are monoidal then we consider ‘monoidal functors’, which also preserve the monoidal structure.

9.2 Hypothesis Generation

One of the most important requirements for general intelligence is that the hypotheses which are generated are *salient*, i.e., pertinent to the task at hand. In the case of the simple pendulum, the closed form expression was obtained by virtue of observation that e.g. the color of the pendulum bob was not relevant, but the angle of displacement was, and so on. However, this does not imply that all features of the pendulum were given equal attention. Humans have sufficiently strong priors about force and motion that it is hard to imagine an experimenter ever consciously entertaining color as a factor. It is therefore evident that scientific hypothesis generation enjoys a degree of subtlety which is absent from traditional ML approaches.

Previous such work on non-quantitative generation of alternative hypotheses can be found in Mitchell and Hofstadter’s ‘Copycat’ [146]. Copycat is proposed as a cognitively-plausible generator of proportional analogies between letter-strings and operates without requiring a ‘top-down, a priori’ objective function. In the abstract, Copycat can be considered as an interpreter for expressions that describe (potentially partially constructed) analogies, in which top-down and bottom-up perspectives interact. At any point in construction, structures can be interpreted to yield what Hofstadter describes as their ‘counterfactual halo’, i.e., to suggest alternative expressions that tend to be more self-consistent. Copycat avoids explicit combinatorial search via a combination of attention heuristics (which share a great deal of commonality with the scheduling mechanism described in Sect. 8.2) and interacting hierarchical constraints. Salient actions are indexed ‘on demand’ by local updates to hypotheses and no form of global truth maintenance is required. These local updates act to greatly prune the space of possible alternatives, being biased by design in the general direction of ‘more consistent, hence more compelling’ hierarchies.

More generally, a number of previous works in cognitive architectures argue that the frame problem is an artifact of an overly rigid perspective on hypothesis representation [97, 146]. Specifically, the claim is that hypotheses should be both causally grounded (via participation in a sensor-effector mapping that receives feedback from the environment) and ‘reified on demand’ via the data-driven context of task features. It is claimed that the arguments of a priori representationalism that lead to the frame problem are then no longer applicable. Such context-specific hypothesis-chaining is an intrinsic aspect of SCL: salience is facilitated via the joint action of fine-grained scheduling and resource bounds on both time and space [325]. In the following section, we describe a general mechanism for interpreting the structure of hypotheses, in which alternative hypotheses are generated via a specific form of interpretation which yields a modified hypothesis as a result.

Compositional Interpretation

Compositional interpretation of SCL-expressions is achieved via *denotational semantics*: the meaning of a composite expression is interpreted as as a function of the meaning of its component parts. In a ‘closed-world’ setting, such an interpretation is fixed at the point of deployment. In an open world setting, it remains forever possible that there are surprising latent interactions between components and the interpretation process must therefore incorporate online learning.

Marcus [210, 214] has previously contrasted deep learning with human capabilities (as inferred via observations from neuroscience), and notes that the former typically lacks generic mechanisms for recursion and variable binding. He proposes a hierarchical structured representation (‘treelets’) for addressing this, but leaves open the question of how they should be manipulated for efficient inference. As regards efficiency, Marcus gives desiderata which are evocative of the ‘active symbols’ of Mitchell and Hofstadter’s ‘Fluid Analogies’ architecture [146], the direct analog of which in SCL is provided via the matching and transformation processes described in the beginning of this chapter.

It has previously been proposed [259] that the category theoretic mechanism of *initial F-algebras* provides an appropriate and parsimonious means of modeling these aspects of human cognition and we describe below a concrete application that supports such recursion and variable binding. F-algebras provide a universal mechanism for the generic interpretation of expressions. By ‘generic’, we mean that a wide class of typed expression languages can be compositionally interpreted. As we describe below, ‘universal’ has a specific technical meaning in category theory, but can for practical purposes be taken to mean that the interpreter is parameterized by the target datatype and can approximate (e.g. via learning [333, 334]) any primitive recursive semantic interpretation.[2] Moreover, the interpreter can both accommodate recursive expression languages and be stateful (and hence perform variable-binding, so meeting both of Marcus’s requirements, above).

[2] Strictly, the expressiveness is more general than primitive recursive in that it includes e.g. Ackermann’s function [154].

$$\begin{tikzcd}
FA \arrow[r, "Fh"] \arrow[d, "f"'] & FB \arrow[d, "g"] \\
A \arrow[r, "h"] & B
\end{tikzcd}$$

Fig. 9.1 *h* as a homomorphism

$$\begin{tikzcd}
F(\mu F) \arrow[r, "F(\mathrm{Cata}\, f)"] \arrow[d, "in"'] & FA \arrow[d, "f"] \\
\mu F \arrow[r, "\mathrm{Cata}\, f"] & A
\end{tikzcd}$$

Fig. 9.2 Cata as a unique homomorphism

Technically, the universiality property arises as follows: for category C and functor $F : C \to C$, an *algebra*[3] consists of a pair (A, f) consisting of an object A and a morphism $f : FA \to A$. A *homomorphism* $h : (A, f) \to (B, g)$ between algebras is a morphism $h : A \to B$ such that the square in Fig. 9.1 commutes.

In the category that has algebras as objects and homomorphisms as morphisms, an *initial algebra* is an algebra that is unique (up to isomorphism) and *initial*, i.e., it can be transformed into any other algebra in the category. We write $(\mu F, in)$ for an initial algebra, and Cata f for the unique homomorphism $h : (\mu F, in) \to (A, f)$ from the initial algebra to any other algebra (A, f). That is, Cata f is defined as the unique arrow that makes the diagram of Fig. 9.2 commute.

The universal interpreter property of Cata then arises by virtue of this initiality [154]. Cata is an abbreviation of 'catamorphism' (from Greek: $\kappa\alpha\tau\alpha$ 'downwards' and $\mu o\rho\varphi\eta$ 'shape'); informally, a generic method of transforming some typed source expression into a target expression (of whatever desired type). Hence, in a category of expressions Ex in which the objects are types and the morphisms are functions, Cata thus provides an *algorithm template* for the interpretation of expressions. Hence predicates on Ex are represented as a transformation of some source expression which has target type bool and alternative hypotheses as having target type Ex.

As previously mentioned, in the wider context of general intelligence, such a 'Closed World Assumption' is insufficient: reality may always intrude to defeat prior assumptions.[4] Such exceptions arise as a function of the difference between expected and observed states—whether because an action fails to yield the anticipated state, or else because some state of interest arises in an unanticipated manner. In the context of SCL, then, the term hypothesis recovers its traditional meaning as a tentative proposition about reality that remains forever potentially subject to revision. When

[3] This definition subsumes the familiar usage of the term.

[4] As discovered by Bertrand Russell's unfortunate fictional chicken.

the distributed transition function of the world model is determined to be in error in this manner, a corresponding 'repair' must be applied to the current denotational semantics. Those repairs can either widen the constraints or suitably constrain the transfer function.

In SCL, constraints are expressed via the predicate of a refinement type. As types are aggregated into composites (via sum and product), so the hierarchical structure of predicates becomes more complex. In the particular case of hypothesis generation, the catamorphic traversal of expressions accumulates proposed alternatives and performs 'conflict resolution' on them in order to propose a hypothesis that better meets its constraints. Any inference mechanism could conceivably be applied in the process, depending on the intermediate mappings between types that are required. Fortunately, the repair process is considerably facilitated by the granular nature of inference: it is typical that repairs require only local changes to inference rules.

It is also interesting to note the overlap here with the contemporaneous work of Goertzel [116], which proposes *chronomorphisms* as a potential unifying mechanism for the OpenCog framework. While we certainly share the belief that the zoo of recursion schemes (including anamorphisms, futuromorphisms, etc.) have a role to play in open-ended inference, we are not of the opinion that their role lies in solving optimization problems, as discussed in Sect. 5.1 on a priori rewards.

9.3 Abstraction and Analogy

Consensus on the value of abstraction in AI dates back to the inception of the field [220]. Various forms of analogical reasoning have similarly been widely argued, by cognitive and computer scientists alike [84, 144], to play a vital role. There are a wide variety of proposed definitions for both. For example, Gentner [107] defines abstraction as 'the process of decreasing the specificity of a concept'. Cremonini et al. [53] define abstraction as 'the process of mapping between problem representations, so as to simplify reasoning while preserving the essence of the problem'. Definitions of abstraction and analogy can overlap considerably. For example, the formal and general abstraction framework of Giunchiglia and Walsh [111] describes abstraction in terms of properties that are provably preserved under the source to target mapping. This definition could also be said to be applicable to *predictive analogy* [271], which is concerned with inferring properties of a target object as a function of its similarity to a source object, the oft-cited example of which is the similarity between the solar system and the Rutherford model of the atom. Given the perceived richness and complexity of abstraction and analogy, this overlap is unsurprising. Indeed, it seems possible that the processes are recursively interleaved in a nontrivial and data-driven manner. Hence, whilst in this Section we propose a concrete mechanism for abstraction that can then be used as a basis for analogy, this should be considered as a pedagogical device rather than any attempt at a definitive statement.

The interpretation of SCL expressions via catamorphisms already provides an elementary abstraction mechanism: the algorithm skeleton for the interpreter abstracts over the generation of alternative types and values from a given expression. However,

$$\begin{tikzcd}
& u' \arrow[dl, "\sigma_1'"'] \arrow[d, dashed, "\sigma'"] \arrow[dr, "\sigma_2'"] & \\
e_1 & u \arrow[l, "\sigma_1"] \arrow[r, "\sigma_2"'] & e_2
\end{tikzcd}$$

Fig. 9.3 Anti-unification as a categorical product. The anti-unifier of expressions e_1 and e_2 is an expression u, together with two substitutions σ_1 and σ_2. For $s \rightarrow_\sigma t$, we say that t is a *specialization* of s, i.e. it has been obtained from s via the instantiation of one or more variables. The *least general* anti-unifier (u, σ_1, σ_2) is the *unique* u such that, for any other candidate $(u', \sigma_1', \sigma_2')$, u is a specialization of u' via some substitution σ'

since expressions in SCL are first-class objects, we may also perform abstraction via other means, such as *anti-unification*. Anti-unification has a variety of uses in program analysis, including invariant generation and clone detection [39]. The various forms of unification [266] can be described categorically [119] via a category in which the objects are terms (i.e. SCL expressions) and the morphisms are *substitutions*, i.e., the binding of one or more variables to subexpressions. Anti-unification is the categorical dual of unification [162], representing the factorization of substructure common to two expressions. The discovery of such 'abstract patterns' is analogous to the induction of subroutines, which can be instantiated across a range of parameter values. More generally, abstraction is applicable across different dimensions of the state space—see Sect. 11.2.2 for a discussion of wider prospects in this regard.

Figure 9.3 depicts anti-unification as a *categorical product*, a construction that generalizes the familiar notion of Cartesian product. The diagram denotes that anti-unifier (u, σ_1, σ_2) is more specialized than all other candidates $(u', \sigma_1', \sigma_2')$ because the latter can be recovered from the former via σ'.

Analogical reasoning in humans appears to afford unbounded creative potential [95]. We share the belief that analogy is a dominant mechanism in human cognition [148, 228] and envisage that computational models of analogy will be a key research area for general intelligence. It should be clear that such research is completely open-ended. The categorical approach we describe below is therefore for pedagogical purposes; the further incorporation of heuristics is a more realistic prospect for practical use.

We give a categorical construction for proportional analogies which builds upon the method of abstraction defined above.

As previously described in Sect. 7.4, the example application domain we consider here is that of letter-string analogy (LSA) problems (e.g., `abc : abd` :: `ijk : ???`). Although the domain may appear simple, it has been remarked that it can require considerable sophistication to obtain solutions that appear credible to humans

$$\begin{tikzcd}
\texttt{abc} \arrow[r, "h"] \arrow[d, "v"'] & \texttt{abd} \arrow[d, "v'"] \\
\texttt{ijk} \arrow[r, "h'"] & \texttt{???}
\end{tikzcd}$$

Fig. 9.4 Proportional analogy as a commutative diagram: `abc : abd` :: `ijk : ???`

[95], not least because the domain is not readily modeled as an optimization problem. It is therefore reasonable to assume that mechanisms for solving LSA problems have high relevance and are applicable for implementing cognitive processes at many levels. Notable approaches to LSA problems include Hofstadter and Mitchell's Copycat [146] and the 'E-generalization' approach of Weller and Schmid [360], although the latter is not cognitively plausible for reasons of scale.

As can be seen in Fig. 9.4, proportional analogy problems can also be considered to form a commutative diagram. The 'abstraction via anti-unification' approach described above can be used as a building block for constructing analogies, for example as is done in 'Heuristic Driven Theory Projection' [308]. In particular, abstraction can be combined with the powerful category theoretic constructions of *pushouts* and *pullbacks* to construct, express, and understand such analogies in a computationally automatable manner across a wide range of expression languages. Specifically, we can use these constructions to determine the possible relationships between our objects $A =$ `abc`, $B =$ `abd` and $C =$ `ijk` such that $D =$ `ijl` is uniquely determined through commutative diagrams.[5]

Pushouts

A pushout may be understood as an 'abstract gluing' of a pair of morphisms $b : A \to B$ and $c : A \to C$. The construction of a pushout involves the construction of some fourth object D alongside morphisms $c' : B \to D$ and $b' : C \to D$ such that the resulting diagram commutes, i.e., $c'(b(A)) = b'(c(A)) \cong D$, where $\cong$ denotes equality up to isomorphism. Further, the resultant commuting diagram must satisfy the 'pushout condition', that for all D' with $e : B \to D'$ and $f : C \to D'$ and $e(b(A)) = f(c(A)) \cong D'$, there exists a unique morphism $d : D \to D'$ such that $d(c'(B)) = d(b'(C)) = e(B) = f(C) \cong D'$.

The meaning of these conditions will become immediately clear through an example of a pushout in the category of sets (typically denoted **Set**). In **Set**, objects are sets and morphisms are functions. Below we given an example of a pushout in **Set**, with objects $A = \{a, b, c\}$, $B = \{1, 2, 3, 4\}$ and $C = \{W, X, Y, Z\}$, and morphisms $h = \{a \to 1, b \to 2, c \to 3\}$ and $v = \{a \to X, b \to Y, c \to Z\}$. For each object in B, C and D, we denote its pre-image in A in parenthesis.

[5] The possibility $D =$ `ijd` is not treated here; a more fully-featured approach to analogy would include nondeterminism and preference heuristics, e.g. as in Goguen's 'sesqui-categories' [117].

$$\begin{tikzcd}
A = \{a, b, c\} \arrow[r, "h"] \arrow[d, "v"'] & B = \{1(a), 2(b), 3(c), 4\} \arrow[d, "v'"] \\
C = \{W, X(a), Y(b), Z(c)\} \arrow[r, "h'"] & D = \{W, 1X(a), 2Y(b), 3Z(c), 4\}
\end{tikzcd}$$

The resultant object $D = \{W, 1X, 2Y, 3Z, 4\}$ with morphisms $v' = \{1 \rightarrow 1X, 2 \rightarrow 2Y, 3 \rightarrow 3Z, 4 \rightarrow 4\}$ and $h' = \{X \rightarrow 1X, Y \rightarrow 2Y, Z \rightarrow 3Z, W \rightarrow W\}$ are the unique pushout of morphisms h and v.

The uniqueness is a result of the pushout condition; if we instead had $D' = \{1X, 2Y, 3Z, 4W\}$ and $v'(4) = 4W$ and $h'(W) = 4W$ then we would have elements in B and C, with no common pre-image in A, being mapped to the same element in D'. As a result, there would be no morphism $d : D' \rightarrow D$, such that the resultant diagram commutes as the offending element $4W$ would need to be mapped to two separate elements in D, 4 and W. Thus we can see that the pushout condition prevents 'confusion' such that, if an element in D has pre-images in both B and C, then those pre-images must themselves have a common pre-image in A.

Similarly, if we have $D' = \{W, 1X, 2Y, 3Z, 4, Q\}$, such that the element Q has no pre-image in B or C, then it may be mapped to any element in D by some morphism $d' : D' \rightarrow D$, such that d' is no longer unique. Thus we can see that the pushout condition prevents 'junk' such that all elements in D must have a pre-image in at least one of B or C. A common phrase describing this property is that v' and h' are *jointly surjective*.

Another relevant concept related to pushouts is that of pushout complements. A pushout complement is the completion of a pair of arrows $h : A \rightarrow B$ and $v' : B \rightarrow D$ to a pushout, and is given by an object C with morphisms $v : A \rightarrow C$ and $h' : C \rightarrow D$ such that the resulting diagram of h, h', v and v' is a pushout.

Pullbacks

If pushouts are 'abstract gluings' of morphisms, then 'pullbacks' may be understood as 'abstract intersections' of morphisms. A pullback is given over a pair of morphisms $h : B \rightarrow D$ and $v : C \rightarrow D$. The construction of the pullback of h and v yields a fourth object A with morphisms $v' : A \rightarrow B$ and $h' : A \rightarrow C$ such that the resulting diagram commutes and satisfies the 'pullback' condition. This condition requires that for all A' with $e : A' \rightarrow B$ and $f : B' \rightarrow C$ and $e(h(A')) = f(v(A')) \cong D$, there exists a unique morphism $a : A' \rightarrow A$ such that $v'(a(A')) = B$ and $h'(a(A')) = C$.

Analogy via Pushouts, Pullbacks, and Pushout Complements

We now demonstrate that these concepts give mechanisms for automatic derivation of analogies according to abstractions. For a more complete description of pushouts, pullbacks, and pushout complements in this context, see Taentzer et al. [76]. In Fig. 9.5, we give a sketch of a pushout-based solution to classic letter-string analogy problems. Letter-strings are represented as lists of natural numbers, with natural numbers themselves represented via Peano arithmetic. We are working in the category

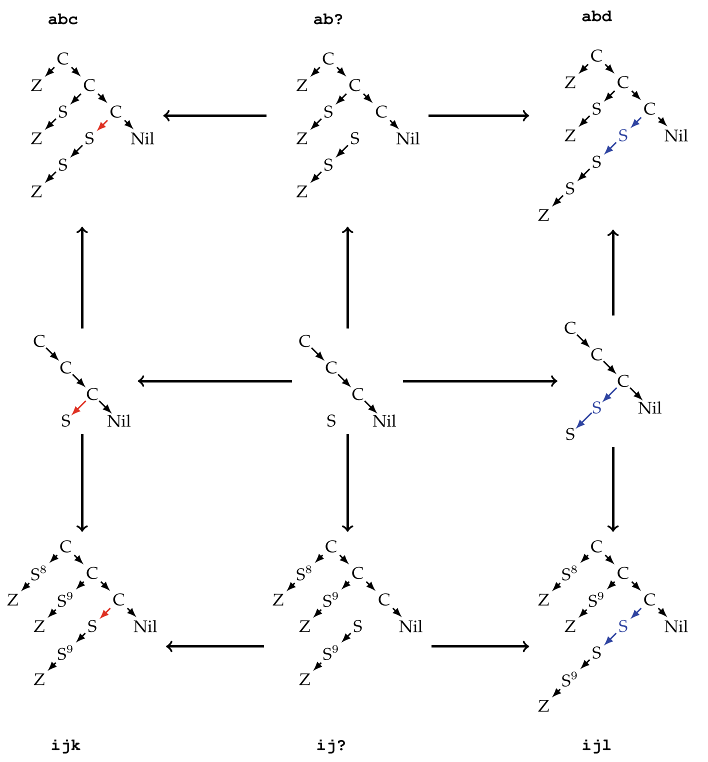


Fig. 9.5 A solution to the letter-string analogy question `abc : abd` :: `ijk : ???` via pushouts. Each letter string is represented by its corresponding tree in Peano arithmetic, with the letter 'a' corresponding to the number '0'. The shorthand 'S^x' is used to concisely represent a sequence of x consecutive successor nodes. Elements in red are 'deleted' by the transformation, whereas elements in blue are 'created'. The middle rule can be viewed as an abstract rule for transforming letter-strings, and interpreted in language as 'increment the final (non-`a`) character in the letter-string'

of labeled graphs (an 'adhesive category' [185]), along injective morphisms which are both label- and structure-preserving. Note that in this scenario, relabeling may be achieved in rewriting systems over labeled objects through the use of partially labeled objects as intermediaries [126].

The example works as follows. We are provided with the expression trees of the letter strings `abc`, `abd`, and `ijk`. We know that `abc` relates to `abd` in some manner and wish to induce the equivalent relation for `ijk` and some unknown letter string. Thus the task may be phrased "`abc` is to `abd` as `ijk` is to what?". The first step is to induce common substructures between the pairs `abc` and `abd` as well as `abc` and `ijk`. Two particularly promising substructures are shown in the top middle and left middle of the diagram. This first step is most critical; the two common substructures will deterministically define the rest of the transformation. If either common substructure is too specific, it may form too rigid a restriction for the rest of the process to be successful. Alternatively, if the common substructures are too general, ambiguity may occur causing there to be several possible analogous letter-strings.

Given the two common substructures, a pullback may be constructed, yielding the graph in the center of the diagram. This may be understood as the common elements of both common substructures such that the top-left square commutes. With the central element, we may then construct the pushout complements in the middle-right and bottom-middle spaces, and finally the pushout of those elements with the central element to give the final analogous graph: `ijl`. A remarkable property of this process is that it actually yields an abstract rule $L \leftarrow K \rightarrow R$ (the elements of the middle row), which may then be applied to any other letter-string expression T according to the following process:

1. Find a graph morphism $f : L \rightarrow T$.
2. Construct the pushout complement of f and $L \leftarrow K$ to give intermediary D with morphisms $t : D \rightarrow T$ and $g : K \rightarrow D$.
3. Construct the pushout of g and $K \rightarrow R$, giving result expression S.

This process is only applicable if there exists a graph morphism $f : L \rightarrow T$, and this is only the case when the letter-string expression T is at least 3 letters long and its final letter is not `a`. Hence the rule $L \leftarrow K \rightarrow R$ may be interpreted in language as; 'increment the final (non-`a`) character in the letter-string'. When applied to the letter-strings `ddd` and `mmjjkk` it gives the unique results `dde` and `mmjjkl` respectively.

Although LSA is given as an example domain, the treatment is a general one for adhesive categories [185]. This means that the principles described above can be leveraged to induce analogies over labeled graphs [76] and their derivatives (e.g. forests and trees), hierarchical graphs [72], and port graphs [85, 267].

9.4 Abduction

There has recently been much interest in the applied category theory community in 'lenses', which provide a simple but powerful abstraction for hierarchical feedback. Originally appearing in database theory and functional programming for describing deeply nested destructive updates [91], they have later turned out to be of central interest in categorical approaches to both game theory [108] and machine learning [88, 90]. One perspective on lenses is that they are a general theory of 'things that compose by a chain rule'.

Lenses

Lenses are the morphisms of a category **Lens** whose objects are pairs of sets, where we think of the first as 'forwards' and the second as 'backwards'. A lens

$$\lambda : (X^+, X^-) \rightarrow (Y^+, Y^-)$$

is a pair consisting of a ‘forwards’ function:

$$\lambda^+ : X^+ \to Y^+$$

and a ‘backwards’ function:

$$\lambda^- : X^+ \times Y^- \to X^-$$

If we have another lens $\mu : (Y^+, Y^-) \to (Z^+, Z^-)$ then the composite lens has forwards function:

$$(\lambda\mu)^+(x) = \mu^+(\lambda^+(x))$$

while the backwards function is given by the characteristic law

$$(\lambda\mu)^-(x, z) = \lambda^-(x, \mu^-(\lambda^+(x), z))$$

The category of lenses has a monoidal structure, which is given on objects by:

$$(X^+, X^-) \otimes (Y^+, Y^-) = (X^+ \times Y^+, X^- \times Y^-)$$

and on morphisms by:

$$(\lambda \otimes \mu)(x_1, x_2) = (\lambda^+(x_1), \mu^+(x_2))$$
$$(\lambda \otimes \mu)^-((x_1, x_2), (y_1, y_2)) = (\lambda^-(x_1, y_1), \mu^-(x_2, y_2))$$

While this definition is written in terms of sets and functions, it turns out that lenses can be more generally defined over (essentially) any monoidal category, although the correct general definition is not obvious [291].

Examples in Machine Learning

Backpropagation. An important class of lenses consist of a function paired with its first derivative, which is usually known as reverse-mode automatic differentiation. Specifically, let **Smth** be the category whose objects are Euclidean spaces and whose morphisms are differentiable functions. There is a functor D : **Smth** $\to$ **Lens** which is given on objects by $D(\mathbb{R}^n) = (\mathbb{R}^n, \mathbb{R}^n)$ where we think of the second $\mathbb{R}^n$ as the cotangent space at a point in the first. On morphisms the functor is given by:

$$D(f) = (f, f^{\#}), \text{ where } f^{\#}(x, v) = J_f(x)^\top v$$

where J_f is the Jacobian of f. In order for this to be a functor it must be the case that the derivate of a composite function fg can be determined from the derivatives of f and g using the lens composition law. This turns out to be essentially the chain rule: Given composable smooth functions f, g, we have

$$\begin{aligned}
(D(f)D(g))^{-}(x,v) &= f^{\#}(x, g^{\#}(f(x), v)) && \text{(lens composition law)}\\
&= J_f(x)^{\top} J_g(f(x))^{\top} v \\
&= [J_g(f(x)) J_f(x)]^{\top} v && \text{(law of matrix transpose)}\\
&= J_{fg}(x)^{\top} v && \text{(multivariate chain rule)}\\
&= (fg)^{\#}(x, v)
\end{aligned}$$

From this it follows that $D(f)D(g) = D(fg)$, i.e., that D is a functor. This connection between lenses and the chain rule was explicitly observed by Zucker [376] and is also implicit in Fong et al. [90].

Variational inference. There is a category whose morphisms $X \to Y$ are probability distributions on Y conditional on X. There are several different ways to make this precise, for example to say that X, Y are measurable spaces and the conditional distribution is modeled as a measurable function $X \to \mathcal{G}(Y)$, where $\mathcal{G}(Y)$ is the measurable space of all probability measures on Y [98, 110]. Equivalently, one might say that objects are finite sets and morphisms are stochastic matrices. Composition of morphisms is by 'integrating out' the middle variable (sometimes called the Chapman-Kolmogorov equation [250]), which is simply matrix multiplication in the finite case. Call this category **Stoch**. There is a morphism **Stoch** $\to$ **Lens** that pairs a conditional distribution with the function that performs Bayesian inversion on it, namely $f \mapsto (f, f^{\#})$ where $f^{\#} : \mathcal{G}(X) \times Y \to \mathcal{G}(X)$ returns the posterior distribution $f^{\#}(\pi, y)$ given a prior π and an observation y. Bayesian inversion satisfies a 'chain rule' with respect to composition, meaning that the Bayesian inverse of a composite conditional distribution can be computed in terms of the Bayesian inverses of the components, and this fact precisely says that **Stoch** $\to$ **Lens** is a functor [318].

Dynamic programming. Consider a Markov chain with state space S, action space A, and (stochastic) transition function $P : S \times A \to S$. Suppose further that actions are controlled by an agent, who obtains utility $U : S \times A \to \mathbb{R}$ on each transition. For each policy $\pi : S \to A$ we obtain a function $f : S \to S$ given by $f(s) = P(s, \pi(s))$, and a function $f^{\#} : S \times \mathbb{R} \to \mathbb{R}$ given by $f^{\#}(s, c) = U(s, \pi(s)) + \gamma c$, where $0 < \gamma < 1$ is a fixed discount factor. The second input to $f^{\#}$ is known as the continuation payoff. These two functions constitute a lens $\lambda_\pi : (S, \mathbb{R}) \to (S, \mathbb{R})$, indexed by the policy. On the other hand, a lens $V : (S, \mathbb{R}) \to (1, 1)$ turns out to be just a function $V : S \to \mathbb{R}$, which we take to be the value function. If V is an initial value function and π is the appropriately optimal policy for it, the lens composition $\lambda_\pi V : (S, \mathbb{R}) \to (1, 1)$ performs a single stage of value function iteration. Thus value function iteration amounts to approximating the limit:

$$\cdots \longrightarrow (S, \mathbb{R}) \xrightarrow{\lambda_{\pi_2}} (S, \mathbb{R}) \xrightarrow{\lambda_{\pi_1}} (S, \mathbb{R}) \xrightarrow{V_0} (1, 1)$$

where each π_i is the optimal policy for the current value function at each stage.[6]

[6] This connection between dynamic programming and lenses is due to Viktor Winschel.

Hierarchical Symbolic-Numeric Lenses

All three examples of the 'lens pattern' we have described above for machine learning are notably 'low-level' and numerical. However, now that the common pattern has been identified, it is possible in principle to design systems which are structurally the same but which are semantically 'higher-level'. This allows the best of both worlds: logical languages embodying GOFAI principles such as abduction and generalization can be combined with the hierarchical feedback which has been enormously successful in numerical and connectionist approaches. One option is to construct a monoidal functor $\mathcal{C} \to \textbf{Lens}$ where $\mathcal{C}$ is a suitable category for higher-level reasoning; another is to build additional structure into **Lens** itself using its more general definition.

The specific approach proposed here is to construct lenses in which the forwards map performs deductive reasoning, and the backwards map performs abductive reasoning. The idea is that the forwards map λ^+ will, given a hypothesis x, generate a deductive conclusion $\lambda^+(x)$, while the backwards map will, given an *initial* hypothesis x and an observation y, abductively generate an updated hypothesis $\lambda^-(x, y)$ in order to explain the observation in a way that is in some sense 'as close as possible' to the starting hypothesis.

Suppose now that from an initial hypothesis x we make a 2-step deduction $\mu^+(\lambda^+(x))$. If we then observe z, we can perform a 2-step abduction using the lens composition law to determine a new hypothesis. First, using the deduced hypothesis $\lambda^+(x)$ and the observation z, we use μ^- to abductively determine the new 'middle' hypothesis $\mu^-(\lambda^+(x), z)$. We then treat this as though it is an observation, which together with the initial hypothesis x abductively determines the final result:

$$\lambda^-(x, \mu^-(\lambda^+(x), z))$$

Another possibility is that forwards maps perform abstraction between different levels of representation of a state, and backwards maps are control *commands* (or *desires*). We will illustrate this with a simple worked example. Consider a factory robot moving in the region:

$$R = \{(x, y) \in \mathbb{R}^2 \mid 0 \leq x \leq 10, 0 \leq y \leq 10\}$$

Since we will only be describing the robot's movement with pseudocode it suffices to informally describe the map. The factory floor is divided into two *zones*, with a path running through both. In each zone there is a *goal* adjacent to the path, representing a place where the robot can pick up or deliver objects. An abstracted description of the robot's position is given by elements of $\texttt{Zone} \times \texttt{Chunk}$, where

$$\texttt{Zone} = \{\texttt{zone1}, \texttt{zone2}\}$$

is the set of zones and

$$\texttt{Chunk} = \{\texttt{path}, \texttt{goal}, \texttt{nothing}\}$$

There is a function $\lambda^+_{\text{pos}} : R \to \texttt{Zone} \times \texttt{Chunk}$ that abstracts the robot's position. Going the other way, a command to move to a certain state at the more abstract level can be 'translated down' into a lower-level command to move in the space R. For this we also need to know the current position in the concrete space R, making the type of the backwards function:

$$\lambda^-_{\text{pos}} : R \times (\texttt{Chunk} \times \texttt{Zone}) \to R$$

λ^-_{pos} need not move the robot directly to a position that satisfies the goal, that is, the equation $\lambda^+_{\text{pos}}(\lambda^-_{\text{pos}}(x, g)) = g$ (known as the 'put-get law' of lenses) need not always hold. Rather, λ^-_{pos} can direct the robot through a series of 'waypoints' by treating the position variable as a state variable. What should be guaranteed is that holding the goal fixed and iterating $\lambda^-_{\text{pos}}(-, g) : R \to R$ from any starting position will after finitely many steps reach a position $x \in R$ satisfying $\lambda^+_{\text{pos}}(x) = g$ (provided the goal is physically reachable for the robot). For example, our $\lambda^-_{\text{pos}}(x, g)$ could be given by the following pseudocode:

- If the current position satisfies the goal ($\lambda^+_{\text{pos}}(x) = g$) then do nothing ($\lambda^-_{\text{pos}}(x, g) = x$).
- Otherwise, if the current position is within a fixed short distance of the goal, then move onto the center of the goal.
- Otherwise, if the robot is on the path, move along the path towards the goal.
- Otherwise, move directly onto the path.

Thus if we iterate $\lambda^-_{\text{pos}}(-, (\texttt{zone1}, \texttt{goal}))$ from a typical starting position then the robot will first move onto the path, then move along the path towards zone 1, and then move onto the goal. Together the functions λ^+_{pos} and λ^-_{pos} constitute a lens:

$$\lambda_{\text{pos}} : R \to \texttt{Chunk} \times \texttt{Zone}$$

We will now demonstrate how this lens can be a part of a hierarchy in which the next level is task-centric. Suppose the robot can carry an object, from the set:

$$\texttt{Object} = \{\texttt{widget}, \texttt{gizmo}, \texttt{nothing}\}$$

and sense its carry weight. A widget weighs 2, a gizmo weighs 7 and no object weighs 0, defining a function $W : \texttt{Object} \to [0, \infty)$.

We define a lens:

$$\lambda_{\text{wt}} : [0, \infty) \to \texttt{Object}$$

where

$$\lambda^+_{\text{wt}} : [0, \infty) \to \texttt{Object}$$

classifies any weight less than 1 as `nothing`, any weight between 1 and 5 as a `widget` and any weight greater than 5 as a `gizmo`. The backward function:

$$\lambda^-_{\mathrm{wt}} : [0,\infty) \times \texttt{Object} \to [0,\infty)$$

ignores the current weight, and takes the desired object to its resulting desired weight, namely, $\lambda^-_{\mathrm{wt}}(w, o) = W(o)$.

We can run these two lenses in parallel, yielding:

$$\lambda_{\mathrm{pos}} \otimes \lambda_{\mathrm{wt}} : R \times [0,\infty) \to \texttt{Chunk} \times \texttt{Zone} \times \texttt{Object}$$

The parallel composition of lenses

$$\lambda_1 : X_1 \to Y_1 \text{ and } \lambda_2 : X_2 \to Y_2$$

is the lens:

$$\lambda_1 \otimes \lambda_2 : X_1 \times X_2 \to Y_1 \times Y_2$$

given by:

$$\begin{aligned} v(x_1, x_2) &= (v_1(x_1), v_2(x_2)) \text{ and} \\ u((x_1, x_2), (y_1, y_2)) &= (u_1(x_1, y_1), u_2(x_2, y_2)) \end{aligned}$$

Unlike the sequential composition defined previously, this is a non-interacting composition.

The second level of the hierarchy will be described as a lens:

$$\lambda_{\mathrm{task}} : \texttt{Chunk} \times \texttt{Zone} \times \texttt{Object} \to \texttt{Task}$$

where

$$\texttt{Task} = \{\texttt{task1}, \texttt{task2}, \texttt{nothing}\}$$

The backwards function takes the current state and the desired task, and returns the desired next state required to complete the task. Task 1 entails collecting a `widget` from the goal in zone 1 and delivering it to the goal in zone 2; task 2 entails collecting a `gizmo` from the goal in zone 2 and delivering it to the goal in zone 1. $\lambda^-_{\mathrm{task}}(c, z, o, t)$, which returns the robot's desired next state given the current state and the task, is given by the following pseudocode:

- If the task is 1 and the held object is a `widget`, then proceed to the goal of zone 2 to deliver it:

$$\lambda^-_{\mathrm{task}}(c, z, \texttt{widget}, \texttt{task1}) = (\texttt{goal}, \texttt{zone2}, \texttt{nothing})$$

- If the task is 1 and the held object is a `gizmo`, then proceed to the goal of zone 2 to return it:

$$\lambda^{-}_{\text{task}}(c, z, \texttt{gizmo}, \texttt{task1}) = (\texttt{goal}, \texttt{zone2}, \texttt{nothing})$$

- If the task is 1 and no object is held, then proceed to the goal of zone 1 to pick up a `widget`:

$$\lambda^{-}_{\text{task}}(c, z, \texttt{nothing}, \texttt{task1}) = (\texttt{goal}, \texttt{zone1}, \texttt{widget})$$

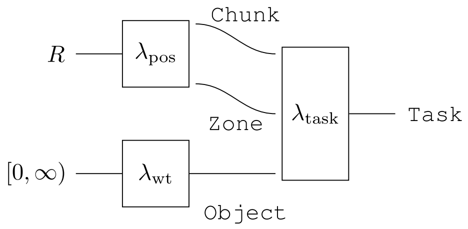
- If the task is 2, then there are three cases similar to the above.
- If there is no task, then remain in the current state:

$$\lambda^{-}_{\text{task}}(c, z, o, \texttt{nothing}) = (c, z, o)$$

We can now compose together the entire control system; it is the lens:

$$(\lambda_{\text{pos}} \otimes \lambda_{\text{wt}})\lambda_{\text{task}} : R \times [0, \infty) \to \texttt{Task}$$

This composite is commonly represented schematically by a 'string diagram' as follows:



The update function of this composite treats the output of $\lambda^{-}_{\text{task}}$, which is the next state desired by the high-level planner, as the input to the first level, which 'translates' it into the lower level of coordinates.

Around this composite system we must place an 'environment'. On the right-hand side sits the human controller or other top-level planner, which decides the top-level tasks given the top-level observations. On the left sits the 'physical environment', consisting of the real world (or a simulation thereof), together with actuators that implement the robot's bottom-level desires and sensors that produce the bottom-level observations. Crucially, this physical environment will typically have an internal state that cannot be observed by the robot.

In our example, for simplicity we take the top-level planner to be a constant task. The physical environment will store the robot's position and currently held object. The current position is continually updated with the desired position provided it is reachable in a straight line. The desired weight is ignored since the robot has no corresponding actuator. When the robot's position is in one of the goal areas, the carried object will change as an object is either picked up or delivered.

If we iterate the composite backward function:

$$((\lambda_{\text{pos}} \otimes \lambda_{\text{wt}})\lambda_{\text{task}})^{-}(-, \texttt{task1})$$

from any starting position, the robot will repeatedly navigate between the goals of `zone1` and `zone2`, picking up and delivering `widgets`. (If it is initially carrying a `gizmo` it will first return the `gizmo` before picking up its first `widget`.)

This setup has the feature that time 'naturally' moves slower the higher one goes up the hierarchy. Suppose the robot's initial position is in `zone2` and it is holding no object. If the task is `task1` then $\lambda^{-}_{\text{task}}$ will output (`zone1`, `goal`, `widget`). This will be used as the desired input to λ^{-}_{pos}. The robot will navigate through several stages towards the goal of `zone1`, during which time the output of $\lambda^{-}_{\text{task}}$ will not change. After the robot reaches the goal, the environment will update its held object to a `widget`, which will cause λ^{+}_{wt} to change its output to `widget`. This in turn will finally cause $\lambda^{-}_{\text{task}}$ to change its output to (`zone2`, `goal`, `nothing`), signaling a change in desire to move to the other goal to deliver the `widget`. This will again stay constant while the lower level λ_{pos} navigates the robot towards the new goal.

Here we have proposed to found abductive inference on the category-theoretic machinery of lenses. Besides abduction, we have also shown how lenses generalize backpropagation, variational inference, and dynamic programming. We then introduced novel 'symbolic-numeric' lenses, which allows hybrid structures, consisting of both symbols and these pre-existing lenses, to be hierarchically composed. This is important for implementing *scalable planning*: the general planning problem suffers from both branching and time horizon, which can be ameliorated by lower dimensionality as well as longer time jumps. This can be achieved by progressively building a hierarchy of 'concepts' and their affordances (cf. Sect. 11.2.2), and operationalizing planning as abductive reasoning at the highest available level, which, thanks to the hierarchical composition, will still be firmly anchored in the sensorimotor level. In the next chapter, we will see how control loops, which are considered from a lens perspective by emerging research in categorical cybernetics, provide a compositional vocabulary to identify and regulate control systems.

Chapter 10
2nd Order Automation Engineering

A scientific theory is intelligible for scientists if they can recognize qualitatively characteristic consequences without performing exact calculations.

H.W. de Regt [61]

In this chapter, semantic closure meets system engineering: we describe how SCL systems can be constructed and controlled in practice, casting a developmental perspective on automation which we call '2nd order automation engineering'. Let us first give context to our objective, starting with a quote from Bundy and McNeil [40], who described in 2006 what they considered to be 'a major goal of artificial intelligence research over the next 50 years':

> For autonomous agents able to solve multiple and evolving goals, the representation must be a *fluent*, i.e., it must evolve under machine control. This proposal goes beyond conventional machine learning or belief revision, because these both deal with content changes within a fixed representation. The representation itself needs to be manipulated automatically.

The *extrinsic* motivation of an SCL system is utilitarian: it is to accept and perform work on command, where work is specified in terms of states to be reached (goals) or avoided (constraints). Thus, the system shall proceed toward constructing a world model such that, ideally, the entire state space is closed under inferencing: starting from as many states as might present themselves, any goal state which concludes the work is predictable (deduction) and conversely, starting from any goal state, as many other states as might occur are reachable (abduction).

By definition, work is a contingent measure for reducing entropy, which is always performed *against* an environment: laws of physics prevail, other agents follow their own agenda and, willfully or not, resist the 'mission' of the system in one way or another. Even the embodiment/agency of the system itself inevitably constrains its prospects of success. In that sense, the *intrinsic* motivation of the system is necessarily structural: it is to broaden the scope and depth ('horizontal' and 'vertical' structure) of

J. Swan et al., *The Road to General Intelligence*,
Studies in Computational Intelligence 1049,
https://doi.org/10.1007/978-3-031-08020-3_10

a world model while absorbing the 'dents' caused by its contingencies. The *function* of an SCL system is therefore to accord both motivations, as per the definition of semantic closure (Sect. 7.2). In other words, this means to sustain a structural goal-directed homeostasis 'at the edge of chaos' as Kauffman puts it [167]—beyond that, ignorance unleashes oscillating behaviors, ending up in futility at best, in disaster at worst. Although Kauffman considers a multi-agent epigenetic landscape over evolutionary time scales, we are concerned with the growth of a single mind over the human time scale of work on command. As we will now see, 'adaptation at the edge of chaos' is as relevant for automation engineering as it is for the development of organisms, when re-cast as the hard-coded drive to control the multitude of feedback loops which unfold over the dynamic landscape of fine-grained models.

Kauffman developed his concept of adaptation from a systemic perspective to explain how open systems evolve in complexity to fulfill their intrinsic determination (spontaneous order) *despite* exogenous constraints (evolutionary pressure). But how do *forms*—the recognizable and composable manifestations of complexity—arise in the first place? When in 1917 D'Arcy Thompson published the first edition of 'On growth and form' [337] he probably did not anticipate that, four decades later, his inquiry into the genesis of stable structures would eventually be met by a formal theory that would extend its reach well beyond embryogenesis. In Thom's 'General Theory of Models' [336], forms arise from a substrate, determined by a *kinematics* (the laws which govern the arrangement/interaction of its constituents) whereas the temporal evolution of forms is regulated by a separate *dynamics*. Indeed, forms generally remain invariant under some selected pseudo-group G of interest (stability), yet sometimes break up completely (catastrophe) and new forms arise (morphogenesis). Thom's theory of models is a mathematical framework for eliciting the dynamics necessary to explain (and when possible, to predict) change, *despite* the inevitable under-specification of the underlying kinematics, and *parameterized* by G.

We confront the related inverse problem: to control the morphogenesis of controllers arising from a known kinematics. A general solution to this problem from the perspective of system engineering—'2nd order automation engineering'—ultimately deserves its own book. In the present work, we limit ourselves to presenting a minimal viable design within the scope of the SCL framework.

10.1 Behavioral System Engineering

As we have amply discussed in Part I, autonomous control systems are not systems that routinely switch between compiled behavior routines. Rather, the function of an autonomous control system is to engineer its own behavioral processes in compliance with requirements of scope, correctness, and timeliness. Behaviors result from control feedback loops instantiated over the substrate of a world model. Hence, to sustain the homeostasis introduced above, is to construct and maintain a world model such that the instantiation of desired control loops is predictable within the prescribed operational envelope. To an SCL system, control loops are the essential observables by which it can assess the adequacy of its world model to the prospects of fulfilling its mission; as such, they constitute the *forms* of interest to be reasoned about.

Relational Control Loops

Functionally, a control loop is the coupling of a subset of rules (the output types of some rules are the inputs types of another) which guarantees the reachability of a goal state, starting from a *terminal* state (for now, the state of an actuator—a definition we will generalize later) constrained by *parameters*; parameters are non-terminal states which are not under the direct control of the loop. The guarantee of reachability is given by the composition of *contracts*, locally defined by rules. A rule is defined by $(A, P_A, f) \rightarrow (B, P_B)$ where A and B are types (generally composite), P_A a predicate on A, P_B a predicate on B, and f a transfer function that maps A to B. The pair (A, P_A) forms a *refinement type* [157], where the predicate P_A defines which subset of the possible representable values actually correspond to valid instances of the type A for the purpose of the rule. A contract is the guarantee that if one predicate holds, so will the other. This perspective of programming-by-contract [245], in which rules are relations (not functions), is close in spirit to the Behavioral System Theory developed by Willems [365] and confers a number of advantages: (1) unlike functions, all relations have a converse, (2) they allow ready modeling of nondeterminism, and (3) they allow identification and composition of invariants.

Figure 10.1 illustrates the relational perspective on control. Further possibilities arising from an alternative categorical presentation via string diagrams due to Baez and Erbele are discussed in Sect. 11.2.4. The generic relational controller (b) is a lens implemented by a single rule

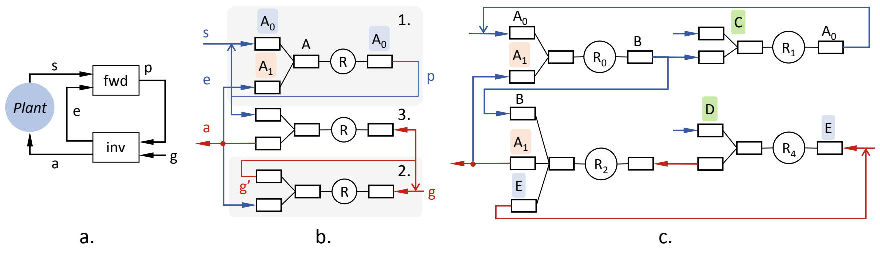


Fig. 10.1 Relational control loops. **a** A generic model-based controller in *functional* form (coupling of inputs/outputs *values*). A forward model *fwd* makes predictions p from sensor readouts s and efference copies e; based on such predictions and set-point g, an inverse model *inv* computes the action a. **b** The same controller in *relational* form (coupling of *constraints* on types) is a lens, implemented by a rule R. The controlled state is A_0, the terminal state A_1, the parameters $\varnothing$. There are two concurrent inference loops: (1) deduction loop and (2) abduction loop. The mixed flow (3) computes possible actions. **c** Complex control loops are composed of several rules/loops. Here, the controlled state is E, the terminal state A_1, the parameters $C \cup D$; see text for details

$$R : (A, P_A, f) \to (A_0, P_{A_0}), A = A_0 \cup A_1$$

through which concurrent inference flows are determined by contract satisfaction. In the first case (b.1), contracts are propagated forward to refine the state of interest (A_0) over an *increasing* time horizon. In the second case (b.3), there exists a solution a such as $f(P_A(a, p))$ satisfies $P_{A_0}(g)$. This is not so in the third case (b.2) where, instead of a, a subgoal g' is abducted such that $P_A(g', e)$ satisfies $P_{A_0}(g)$. The constraint refined by g' is furthermore refined over a *decreasing* time horizon as the b.2 loop iterates until the circumstances (here, p) are eventually such that b.3 holds—then the loop unwinds. This contract-constrained coupling of inferences constitutes the general kinematics of the ruleset and drives the instantiation of control loops at any scale in terms of rules involved.

The SCL framework is parameterized by expression language. For concreteness and ease of explanation, we adopt for the rest of this chapter, the language of *first-order linear arithmetic*[1] (FLA) [33], in which types are subspaces[2] of $\mathbb{R}^n$, rules are linear, and predicates are constructed inductively through Boolean combinations, i.e.,

$$\neg\varphi,\ \varphi \wedge \psi,\ \varphi \vee \psi,\ \varphi \to \psi,\ \varphi \leftrightarrow \psi,\ \forall x_i.\varphi,\ \exists x_i.\varphi$$

where φ and ψ are linear arithmetic formulae. Note that FLA negations express that subregions of the state space are 'to be avoided' to the degree of their likelihood—'forbidden' when the likelihood approaches one—a trait we will leverage in Sect. 10.4. Predicates in FLA are closed under the application of transfer functions, hence the predicate constraining the result of a rule can be automatically derived from the predicate constraining the inputs, namely by applying the transfer function to all of the constants, variables, and linear functions in the input predicate. Contracts can therefore be composed along any arbitrary chain of rules coupled via nonempty intersection between their input/output refinement types. This, in turn, allows representing control loops as (macro) rules and composing them as such; this plays an important part in system identification as we will discuss later.

Hierarchical Behavior Selection

Behaviors result from planning, which results itself from the instantiation of control loops, subjected to the kinematics discussed above. In SCL, abductions are always simulated: this allows exploring, concurrently, several possible courses of action (plans) to achieve a given goal. Of course, at some point in time, the system has to *commit* to one of these plans and start acting.

Each time an efference copy is a premise of a prediction, the corresponding (terminal) goal is added to an auxiliary list associated with the prediction. Such lists are

[1] We discuss other possibilities in Sect. 11.2.1.

[2] Recall that, as befits the open-ended setting, n varies dynamically, according to the addition of synthetic dimensions.

concatenated as predictions become premises of others, provided that the trajectories they define over the state space are consistent. As a result, a plan is a prediction of a goal state, associated with a list of terminal goals ordered by deadline. We call ‘end-plan’ a plan whose goal state M is the one imposed by the mission. All end-plans pursue the same goal and, at any point in time, there is at most one end-plan committed to. Anytime the deadline of a goal g (part of an end-plan) is reached, a procedure `commit` is invoked—g is said to be *signaled*. Note that the procedure is invoked concurrently by goals whose deadlines overlap within its WCET. At time t a goal g invokes `commit`:

1. If there is a current plan and its likelihood is below some threshold of acceptance T, then cancel the plan.
2. If there is no current plan, then find the best one (at minima, the one with the highest likelihood above T); if none is found, return.
3. Execute all goals in the current plan, whose deadline falls in $[t - WCET/2, t + WCET/2)$ and which have not been executed yet and are signaled. Due to the composite nature of states, non-terminal goals are the conjunction of sub-goals. If a terminal goal k is in such a conjunction C, then the execution of k is effective if and only if the executions of all other goals in C are effective.
4. If the execution of g is not effective, then cancel g.

For the commitment procedure to be effective requires that end-plans can be produced before the deadline of its earliest goal, which is of course rarely the case in practice. To keep the WCET of computing end-plans within acceptable bounds requires hierarchical planning. Rules are coupled via their types, which in turn are hierarchized by virtue of abstraction; in that sense, type abstraction imposes a hierarchy upon the ruleset. Consider for example, a mobile robot with an arm equipped with a gripper. When the gripper G is actuated, effectively seizing an object O, then when the robot moves, observations that O moves along the same trajectory as G will ensue: a rule R will be synthesized (see next section) to capture these, along with a positive precondition P on R predicting the success of R based on the prior context of A having been close enough to G and G having been actuated. The successful firing of P actually signifies 'G grabbed O', denoted by a new abstract type T (also detailed in the next section). In terms of abduction, T is considered a terminal state in the sense that the modalities of reaching T are irrelevant to the activity of planning the movement of O to some target location.

All type abstractions (super-types, latent states, etc.) actually decouple control loops. Accordingly, we extend the definition of terminal states to ‘states transduced to lower levels of abstraction’—sensors and actuators being at level zero. ‘Transduced’ means that there exists a rule (a *transducer*) whose right-hand state is at a level of abstraction higher than that of its left-hand state. Hence, a terminal state of a control loop at level N may be the controllable state of another loop at $N - 1$, mediated by some transducer, as illustrated in Fig. 10.2. Finally, we can extend our definition of an end-plan to ‘a plan whose goal state is a terminal state’, considering the top-level mission goal state M a terminal state. The `commit` procedure will be invoked for

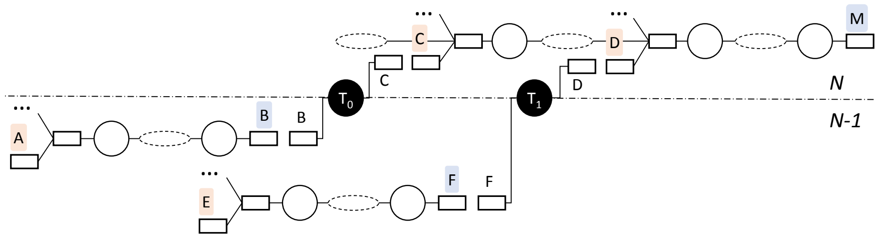


Fig. 10.2 Hierarchical behaviors (an ellipse denotes a path across the ruleset). An end-plan with end-goal M at abstraction level N is composed of terminal goals, some of which (C and D) are transduced (rules T_0 and T_1) into lowers levels: there these global goals become end-goals for local end-plans

all end-plans at some level N, each plan being computed in parallel to assemble the subgoals of plans at level $N + 1$.

System Identification

In automation engineering, system identification refers to the activity (human labor) of modeling an existing system at some appropriate level of description, chosen to confer some expected operationality: this can be, for example, that of predicting, controlling, upgrading, or all of these (e.g. to build a digital twin [280]). Confronted by the variability of both its environment and mission, an SCL system continually adapts its world model to ensure the existence of the necessary control loops, the set of which constitutes a model of its capabilities (self model) while—conversely—delineating the frontiers of its ignorance. Self-identification is the internal process producing and maintaining such a self model for a purpose: deficiencies therein are bound to trigger adaptation of the world model via rules and types *synthesis*. Conversely, adaptation is also triggered externally, any time the actual world is 'surprisingly' at odds with its model. Accordingly, the SCL dynamics is implemented by two complementary and concurrent synthesis processes: one proactive, the other reactive—Fig. 10.3 gives an overview.

In SCL, both the self and world models are written in the same language expressing the same substrate, on which operates the same categorical interpreter—the self model consists of rules which abstract the rules modeling the world. At this operational level of description, a control loop is a mapping between a system trajectory toward a goal state and a subset of the world model at some level of abstraction (its local substrate). To identify a control loop is to find a substrate matching the pattern illustrated in Fig. 10.4. Specifically, the substrate of a loop controlling the state S is a set of rules verifying the following structural properties:

(P_1) There exists an inference loop for predicting and subgoaling S.

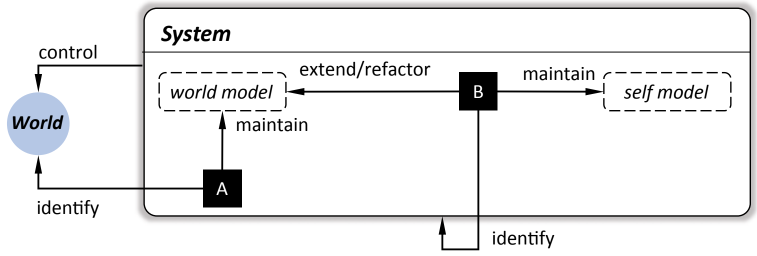


Fig. 10.3 Reflective system adaptation. System identification is performed *on* and *by* the system itself as its world model grows. The dynamics of the system is controlled by two processes by means of rule and type synthesis: A *reacts* to variations of surface phenomena (Sect. 10.2), whereas B *proactively* invokes structural heuristics (intensification/diversification, Sect. 10.3). B is domain-independent, decoupled from A, and operates deeper in the structure of the world model and at longer time horizons. Constraints on resources and responsiveness are less stringent for B than for A

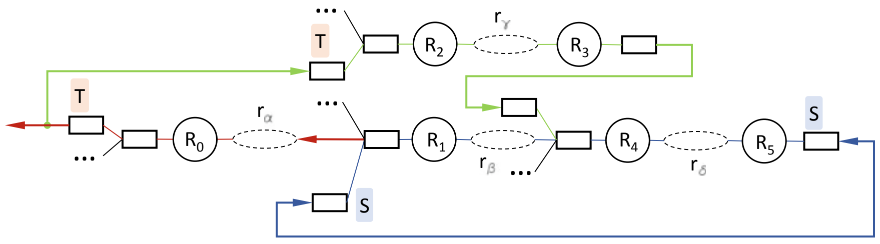


Fig. 10.4 General control loop template. T is the terminal state, S the controlled state. The properties P_1, P_{2a} and P_{2b} are illustrated in blue, red, and green, respectively; see text for details

(P_2) There exists a terminal state T such that (a) T is abducted from S and (b) S is predicted from efferent copies of T.

The self model is both hierarchical and compositional, both properties being inherited from rules and contracts. Technically, two rules

$$X : A \rightarrow B \text{ and } Y : C \rightarrow D \text{ with } B \cap C \neq \varnothing$$

are composed serially into a third $X + Y = Z : F \rightarrow G$, where

$$F = (A \cup C) \setminus (B \cap C) \text{ and } G = (B \cup D) \setminus (B \cap C)$$

We extend the $+$ operator to express joint-serial composition: a set of n rules $\{X_i : A_i \rightarrow B_i\}$ is composed with a rule

$$Y : C \rightarrow D \text{ with } B_i \cap C \neq \varnothing$$

into a third $X + Y = Z : F \rightarrow G$ where

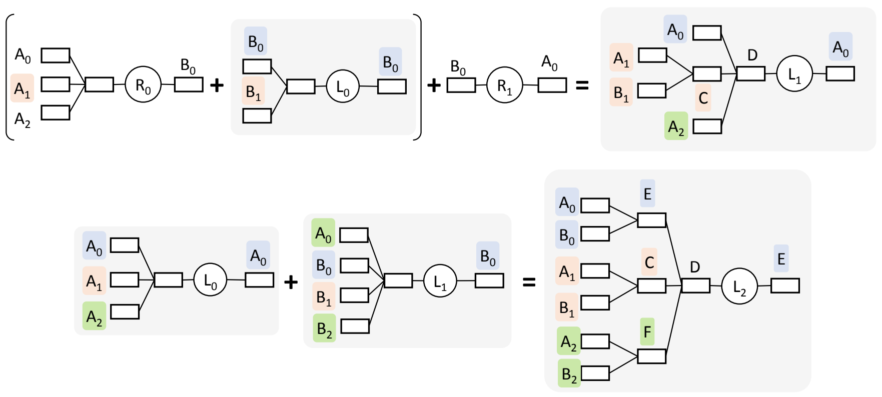


Fig. 10.5 Loop composition. The result of a + operation is a loop describing its arguments at one level of hierarchy higher. Controlled states are marked in blue, terminal states in orange, parameters in green, and control loops in grey; see text for details

$$F = \bigcup_{i=1}^{n}(A_i \cup C) \setminus \bigcup_{i=1}^{n}(B_i \cap C)$$

and

$$G = \bigcup_{i=1}^{n}(B_i \cup D) \setminus \bigcup_{i=1}^{n}(B_i \cap C)$$

Control loops can therefore be expressed by the same formalism used for rules (see Fig. 10.5) and, in particular, the general loop template with terminal state T and controlled state S is

$$L(T, S) = \{R_0 + r_\alpha + R_1 + r_\beta, R_2 + r_\gamma + R_3\} + R_4 + r_\delta + R_5$$

where the r_i are free parameters and the R_i are subject to the following constraints:

$$\begin{aligned} &R_0 : A_0 \to B_0, T \in A_0 && R_1 : A_1 \to B_0, S \in A_1 \\ &R_2 : A_2 \to B_2, T \in A_2 && R_3 : A_3 \to B_3 \\ &R_4 : A_4 \to B_4, B_3 \cap A_4 \neq \varnothing && R_5 : A_5 \to B_5, S \in B_5 \end{aligned}$$

L is then the search pattern for identifying control loops during system identification at arbitrary levels of abstraction.

10.2 Reactive Synthesis

The reactive synthesis of a rule is triggered by one of the following two events: the unpredicted success of a goal, or the failure of a prediction—a missing path in the model structure in the first case, a faulty path in the second. We begin with the first case.

Local Scale: Rules

We are concerned with synthesizing a rule of the form $A \rightarrow B$, where B is the observed goal state which the system could not predict, and A the history of all states observed before B. Of course, a heuristic is needed to limit the scope of A. For example, A can be restricted to the tuple formed by the last N most salient states only—the saliency of a state S being the attention it receives relatively to others, i.e., the relative number of references to S in the jobs of the highest priority (see Sect. 8.2). Of most relevance, we introduce at the end of this section the notion of *postcondition* which summarizes histories of states as hierarchical patterns. Saliency can then be strengthened considering the 'height' of a state in a hierarchy, reminiscent of the 'chunks' theorized to compose human short-term memory and learning—see [114] for a computational model.

As illustrated in Fig. 10.6, the synthesis of a rule $A \rightarrow B$ requires $dim(A) + 1$ successive non-colinear sample pairs (a_i, b_i) of vectors in A and B. Each newly observed vector in A (e.g. a_3) is transformed into its rejection from the hyperplane formed by the previous samples (a_0 and a_1); its associated vector in B (b_2) is transformed accordingly (into b_3). It then suffices to transform the resulting orthogonal basis into an axis-aligned unit basis and to transform the b_i accordingly in order to

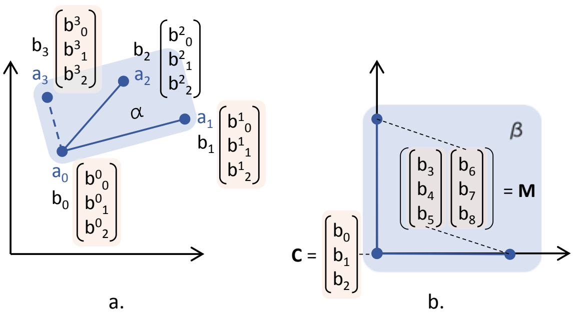


Fig. 10.6 Synthesis of a rule $(A, P_A, f) \rightarrow (B, P_B)$ with $A = \mathbb{R}^2$ and $B = \mathbb{R}^3$. **a** Samples, each a pair (a_i, b_i) of vectors in A and B, are observed in a succession from which an orthogonal basis (α) of A is constructed. **b** α is then transformed into an axis-aligned unit basis (β); f is $b : B = C + M \times a : A$. P_A is the polytope formed by the a_i whereas P_B is that formed by the b_i; see text for details

define the transfer function f as $b : B = C + M \times a : A$. P_A is the polytope enclosing the a_i whereas P_B is that enclosing the b_i. Null columns in M indicate that the inputs located in the tuple A at these column indices are irrelevant to the output; both A and P_A are pruned accordingly.

Polytopes grow as new positive evidences (samples) are observed whereas negative evidences trigger the synthesis of new local restrictions of f. In general, f is not linear globally—over the entire state space—hence applying f outside of P_A may yield an error beyond acceptable tolerance. As a consequence, a new rule R' is to be synthesized as above, whose predicate P'_A is disjoint from P_A. By this construction, P_A is kept convex: convexity warrants interpolation should an input fall within a predicate; outside the predicate, outputs are extrapolated, albeit with a confidence decreasing with the distance of the input to the predicate. Note that it is also possible to subject the likelihood of interpolated outputs to experience. For this, it suffices to maintain the covariance matrix and centroid of all past inputs: the likelihood of interpolated outputs is then inversely proportional to the Mahalanobis distance (times the reliability of the rule and the likelihood of the inputs).

Global non-linear mappings are approximated as the union of local linear ones, each defining its own convex local restriction of a more global domain. The mutual exclusion of the local domains P_A and P'_A is maintained, as they grow, using *preconditions* as follows:

1. If $P'_A \subset P_A$, then a negative precondition on R is synthesized with P'_A as its left-hand predicate.
2. Otherwise, if $P'_A \cap P_A \neq \varnothing$ then one negative precondition is synthesized for each rule, with $P'_A \cap P_A$ as its predicate.
3. Otherwise, no precondition on either rule is synthesized.

Global Scale: Pre- and Postconditions

A second possible trigger for the reactive synthesis of a rule is the failure of a prediction. We proceed as in the case of the unexpected success of a goal, to synthesize a rule N: $(C, P_C, f) \rightarrow (D, P_D)$ where (D, P_D) is the refinement type denoting the failure of some rule R. N is a negative precondition for R: when N fires, the reliability of the inferences produced by R is lowered, proportionally to that of the inference $d : D$ produced by N; during abduction, a goal matching the right side of R triggers the production of a subgoal to prevent N from firing, hence, other subgoals are derived in order to avoid instances of C. Figure 10.7 illustrates the flow of inferences.

Whereas predicates in left-hand refinement types impose *local* constraints on forward input bindings, negative preconditions impose *global* constraints on the target rule, i.e., they define the context in which the rule is likely to fail, regardless of the validity of the bindings of its left-hand refinement type. Just as there exist contexts for failure, there exist contexts for success: these are captured in the left-hand refinement types of positive preconditions—see Fig. 10.8. The synthesis of the positive precondition P is triggered by the success of R if not already predicted

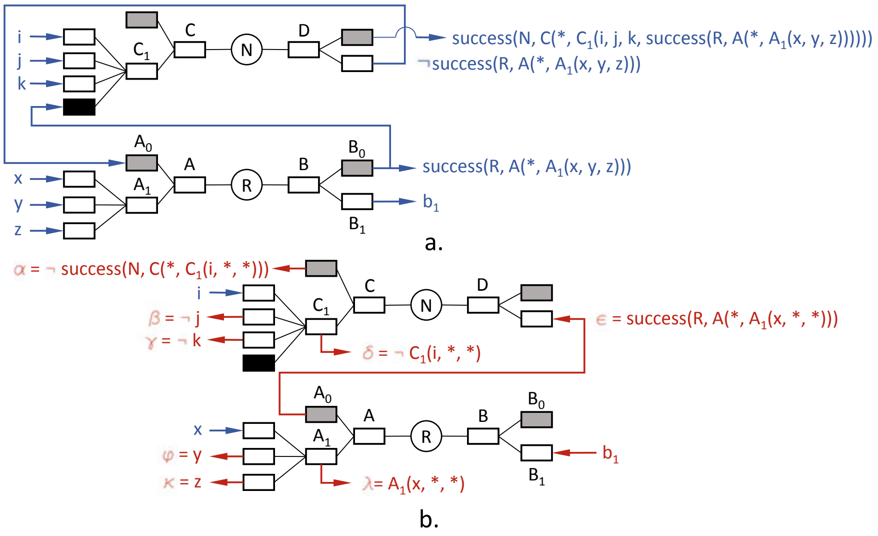

i
j
k
C
C1
N
D
success(N, C(*, C1(i, j, k, success(R, A(*, A1(x, y, z))))))
¬success(R, A(*, A1(x, y, z)))
A0
A
A1
R
B
B0
B1
x
y
z
success(R, A(*, A1(x, y, z)))
b1
a.
α = ¬ success(N, C(*, C1(i, *, *)))
β = ¬ j
γ = ¬ k
δ = ¬ C1(i, *, *)
∊ = success(R, A(*, A1(x, *, *)))
φ = y
κ = z
λ= A1(x, *, *)
b.

Fig. 10.7 A negative precondition N for a rule R is represented. The first type (in grey) embedded in the type of either the left- or right-hand term of R always denotes the success of R; inputs matching A_0 or B_0 do not trigger inferencing. **a** Forward inference (in blue): any time an instance $a : A$ triggers a forward inference, a prediction $success(R, a)$ is produced by R, which in turn may trigger the production by N of its negation: this lowers the likelihood of b_1. **b** Backward inference (in red): $b_1 \rightarrow \lambda \wedge \epsilon, \lambda \rightarrow \varphi \wedge \kappa, \epsilon \rightarrow \delta \vee \alpha, \delta \rightarrow \beta \vee \gamma$

by another positive precondition. Note that preconditions are rules and as such, can themselves be subjected to other preconditions.

Any time a precondition $P : A \rightarrow S$ (where S is the success/failure of some rule R) is synthesized, an abstract type X and a transducer T are also surmised: X is defined as a new dimension of the state space, and T as $A \rightarrow X$. The transfer function of T is learned as any other function, with the minor difference that since X is abstract (therefore not observable), evidences for the outcome of R are taken as proxies for evidences of X.

Mirroring preconditions, whereas predicates in right-hand refinement types define local guarantees on forward output bindings, *postconditions* define global consequences of rule firings. Recall that forward rule firing statements (e.g., $success(R, ...)$ or $\neg success(R, ...)$) are first-class states: hence they can be surmised as input types embedded in the composite left-hand type of a rule. Postconditions are rules taking the (forward) firing of other rules as input types. Such input types summarize (parts of) the state history: if a rule R admits A_i as input types composing its left-hand type, then the single input type $success(R, ...)$ in a postcondition S is equivalent to S admitting all the A_i (and the conjunction of the negations of all negative preconditions for R) as its own input types. Conversely, an input type $\neg success(R, ...)$ in S is equivalent to the negation of at least one of the A_i or the assertion of at least

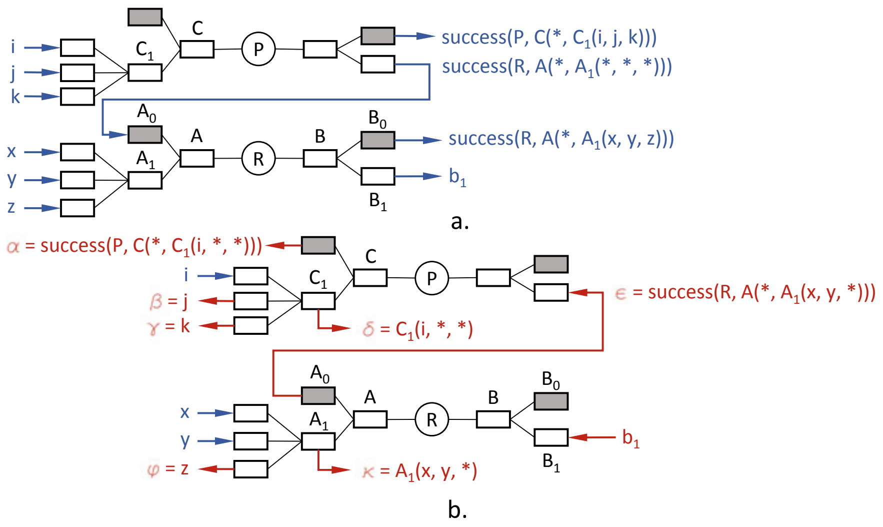

i
j
k
C
C1
P
success(P, C(*, C1(i, j, k)))
success(R, A(*, A1(*, *, *)))
A0
A
A1
R
B
B0
B1
x
y
z
success(R, A(*, A1(x, y, z)))
b1
a.
α = success(P, C(*, C1(i, *, *)))
β = j
γ = k
δ = C1(i, *, *)
ϵ = success(R, A(*, A1(x, y, *)))
φ = z
κ = A1(x, y, *)
b.

Fig. 10.8 A positive precondition P for a rule R is represented. **a** Forward inference: any time an instance $c : C$ triggers a forward inference, a prediction $success(R, *)$ is produced by P: this increases the likelihood of b_1. **b** Backward inference: $b_1 \rightarrow \kappa \wedge \epsilon, \kappa \rightarrow \varphi, \epsilon \rightarrow \delta \wedge \alpha, \delta \rightarrow \beta \wedge \gamma$

one of the negative preconditions for R. Note that some of the A_i may themselves be the firing of postconditions: this enables postconditions to capture state histories as temporal hierarchical patterns.

10.3 Proactive Synthesis

The purpose of proactive synthesis is to mitigate deficiencies observed in the set of macro-rules (control loops) describing the system, resulting from the process of system identification, as discussed in Sect. 10.1. We now proceed to describe an elementary strategy for proactive synthesis which frames intrinsic motivation in terms of control—possible alternative approaches are clearly an open-ended research issue. Within the discipline of metaheuristic search, it is often useful to characterize the state trajectory of a system as *diversifying* or *intensifying*.[3] According to Glover and Laguna [113]:

> The main difference between intensification and diversification is that during an intensification stage the search focuses on examining neighbours of elite solutions. The diversification stage on the other hand encourages the search process to examine unvisited regions and to generate solutions that differ in various significant ways from those seen before.

[3] The related notions of *exploration* and *exploitation* also appear in the literature on evolutionary computation and intrinsic motivation.

As observed by Blum and Roli [26], the notions of intensification and diversification are not mutually exclusive. For example, while random search is strongly diversifying and gradient descent is strongly intensifying, simulated annealing [172] lies somewhere in-between, progressing from diversification at high temperatures to intensification at low temperatures.

The control mechanism we describe here is derived from that of the *Reactive Tabu Search* (RTS) [18, 19] an extension of Glover's original tabu search [113]. Tabu search is a local search metaheuristic [208] in the same family of 'single-point search' techniques as stochastic gradient descent and simulated annealing. The basic mechanism of tabu search (termed *recency-based memory*) maintains a restricted local neighbourhood by prohibiting the choice of a neighbouring state if (some attribute of) that neighbour has been encountered recently. The simplest (or *fixed-tabu*) implementation implements the recency structure as a sequence of the last k states encountered, where k is the *tabu-tenure*. In addition to the recency-based memory structure (which could be said to model 'short-term' memory), many implementations also maintain a 'long-term' or *frequency-based* memory, which is essentially a frequency-histogram of attributes with a tenure much larger than k.

The essential idea of RTS is to inform control via dynamical system metrics. We therefore briefly recap dynamical systems terminology. Informally, an attractor of a dynamical system is a set of points in state space such that all 'nearby' points in the state space eventually move close to it. The simplest dynamical system is one in which there is a single fixed point which acts as an attractor for all states. The next simplest attractor is a *limit cycle*, in which trajectories converge to a closed loop. The *cycle length* of a dynamical system in state s is the number of iterations since s was last encountered (or ∞ if no encounter with s is recorded). It is also possible for the trajectory to be confined within some region of phase space but exhibit no obvious periodicity, due to the presence of a so-called *strange attractor*. RTS thus instruments the search space with recency and frequency information, in order to drive control mechanisms that:

1. are self-adaptive (tabu-tenure is a function of the moving average of the detected cycle-length);
2. maintain a good balance between intensification (i.e. exploration of promising regions) and diversification;
3. recognize when the search lies in the attractor of an unpromising region.

The presence of previously-encountered attributes in the recency list triggers the 'fast-reaction' mechanism which leads on successive iterations to a geometric increase in recency tabu-tenure. This tends to force the search to choose neighbours with unexplored attributes and will eventually break any limit cycle. Conversely, a 'slow-reaction' mechanism acts to counter redundant prohibition by reducing the tabu-tenure when the fast-reaction mechanism has been idle for some time. In order to detect the presence of strange attractors, the number of repeated attributes in the frequency structure is examined. When the number of such attributes exceeds a threshold level an 'escape mechanism' is activated, which (as per Battiti's original implementation) consists of a random walk of length proportional to the moving average of the current cycle length.

Measures which characterize intensification and diversification are obtained from the recency and frequency structures and used to select of an inference method. Describing any particular selection mechanism in detail would be overly specific: capturing entire families of indexing strategies via representations such as fuzzy logic [372] is clearly possible. Hence, we intentionally give here a qualitative description:

1. When intensifying, the system tends to:
   - apply or synthesize rules which tend to move towards goal states;
   - compress known regions of the state space via union of more primitive regions;
   - invent synthetic types via the application of abstraction.
2. When diversifying, the system tends to:
   - invent some target type which does not overlap with any explored region;
   - place low priority on goal states (e.g. potentially bypassing a theoretically reachable goal state in favor of traversing new areas of the state space);
   - synthesize new rules and types via analogy (i.e. pick two rules with a common domain).

The heuristics above are merely intended to give a flavor of the relationship between metrics and strategies. Cross-domain heuristics for proactive synthesis are clearly of great potential value, and are rightfully the subject of an extended empirical investigation.

In order to make a choice of rules and target types in the above, the selection mechanism must ultimately be grounded in a specific state space. There are actually two choices here: (1) the 'first-order' state space described by the current ruleset and (2) the infinite 'second-order' state space described by the prospective addition of rules. By means of defunctionalization [287], second-order rules of the form $A \rightarrow (B \rightarrow C)$ can be 'uncurried' into first-order rules $(A, B) \rightarrow C$, thereby allowing all selection decisions to be grounded in a first order state space as follows:

- The state of a type is given by its predicate.
- The state of a rule is the swept-surface defined by interpolating the application of transfer function between the domain and codomain values associated with predicates.
- The measure of *static-intensification* of some newly proposed type or rule with respect to a ruleset is a function of the degree of overlap with (i.e. polytope intersection) that ruleset.
- The measure of *static-diversification* of some newly proposed type or rule with respect to a ruleset is a function of the distance from the closest point in that ruleset.
- The *dynamic* analog of the above measures is considered with respect to the recent past/near future trajectory of the state of the ruleset, optionally with some time-discounted weighting factor.

In summary, the above allows higher-order cognition to be framed in terms of reactive control, as applied to the representation of the reasoning process itself. One key issue is scalability (i.e. how to avoid 'long-tailed' fragmentation of the representation). The proposed means of addressing this is to favour the frequent application of abstraction as a means of imposing equivalence classes on the search space.

10.4 Safety

Safety is rightfully a dominant concern for autonomous systems. Indeed, taken to the extreme, 'AI safety' has spawned an academic subdiscipline devoted to the most dire of futurist predictions [30]. However, we are concerned here with the mismatch between requirements and the capabilities of contemporary approaches. We can broadly consider safety to be 'confidence that the system will not exceed its operational envelope'. There is clearly a continuum of formality with respect to the degree of confidence and bounds on/communicability of the envelope. Communicability is a bidirectional notion: how accurately has desired behaviour been specified to the system, and how interpretable is the system's representation of it?

Regarding contemporary approaches: at one extreme, we have formal methods, which permit the explicit delineation of behaviours, together with guarantees that forbidden regions of the state space will not be entered. In the manner of traditional symbolic systems and (idealized) theorem provers such as the Gödel machine [306], it is possible to enshrine 'ground truths' within the system by equipping the seed with axioms and sound rules of inference. At the other end are approaches layered on deep (reinforcement) learning, in which behaviour is indirectly mediated via reward, and obtaining confidence in the behavioural envelope is an active research area.

Considered in isolation (i.e. independent of some more autonomous invoking architecture), these approaches are anyway only applicable in 'closed world' environments, the dimensions of which must be prescribed a priori by human ingenuity and which are assumed to be unchanging thereafter.

Experiential Safety

In the open-world setting of genuinely autonomous systems, there may always be possibilities not foreseeable as a function of current inference. This could be because e.g. the sensors are not equipped to detect the associated latent interactions a priori; the butterfly effect; emergent properties of interactions, etc. As befits the scientific method, knowledge obtained via the system's own hypothesizing is therefore provisional, and 'facts' are simply hypotheses that have proved useful in the context of recent tasks. However, overriding constraints remain in place: *to the best of its knowledge* the system will never enter a forbidden region of the state space. Hence a robot tasked with remaining upright could fail to do so, given ignorance of high winds, for example. In such an 'experiential safety' setting, safety guarantees are at both global and local levels: the local level is concerned with guarantees at the scale of single inference steps, the global scale with the longer-term behaviour of the system. Regarding local guarantees, the framework provided by F-algebras is of particular importance for general intelligence, in that it allows safety to be reconciled with open-endedness. Specifically:

- **Open-endedness**: the *algorithm template* for the F-algebra of a type can be programmatically derived [334], even if the type has been synthesized online by the system. The template orchestrates the invocation of learned rules, as described in Sect. 10.3.
- **Safety**: The interpreter defined by the algorithm template can nonetheless be constrained to well-defined behavior, i.e., mapping only between prescribed input and output types with required constraints.

Global safety properties of general interest are *reachability*, i.e. find the set of states reachable from a given initial state X_0 and *controllability*, i.e. find the set of states controllable to a given final state X_t. Modulo efficiency considerations, the bidirectional nature of rules means that reachability and control can be considered equivalent—they are anyway the same in linear systems, for example (further details of a categorical treatment in this setting can be found in Section 11.2.4). Depending on the properties of the expression language used, variants (e.g. point-to-point reachability) may be more computationally efficient. More generally, such guarantees can be divided into two categories:

Formal Reachability

In this approach, we temporarily assume the soundness of current hypotheses—based, for example, on the reliability of rules and likelihood of states. In linear continuous-time dynamical systems, reachability is decidable [131]. In this setting, reachability can be updated periodically, e.g., whenever a rule/type is added/deleted, or perhaps less frequently as an empirically-determined tradeoff between task urgency and available resources.

Time-Bounded Reachability

For expression languages in which reachability is not statically decidable, an alternative is to assume the world remains unchanged while a simulation procedure runs in the background: effectively iterating (perhaps some approximation of) the current transition relation of the system to determine which states are reached. Hence, this approach is likely to be of greatest value for determining the behavioural envelope in the near-term.

Chapter 11
Prospects

'Understanding as Representation Manipulability' reduces understanding — a notion that picks out a particularly complex cognitive state — to representation, inference, and object manipulation.

D. A. Wilkenfeld [364]

11.1 Summary

Intelligence as Managing Limited Resources

The failure of GOFAI created a vacancy for a new guiding philosophy for AI. As it happened, a new perspective was already waiting in the wings: having previously been repeatedly rejected by peer review, Rodney Brooks's behavior-based approach to robotics [37] had privately gained traction, with seminal works [35, 36] setting out the philosophy and practice of the 'Physical Grounding Hypothesis' [35]. Soon, the notion that 'intelligence requires a body' was common parlance in the AI community. In retrospect, this ostensible embrace of embodiment was rather more superficial than one would have hoped: Brooks subsequently observed [34] that in certain areas of AI research, *"hardly a whiff of the new approaches can be smelled"*. Relatively soon thereafter, nascent enthusiasm for deep learning turned attention away from the intrinsically 'in vivo' Brooksian approach towards the typically more forgiving 'in silico' environments, in which interaction with real-world objects was replaced with the simpler (but seemingly hard-to-generalize) task of classifying pictures of them.

As happened with the move from analog cybernetic feedback to digital loss functions (Sect. 7.2), we claim that an essential guiding property was thereby lost, to the detriment of AI philosophy and practice. We have argued throughout that notions of 'universal intelligence' [194] and associated AI architectures such as AIXI [153] have no economic utility, since they are not in any way grounded in the finiteness

J. Swan et al., *The Road to General Intelligence*,
Studies in Computational Intelligence 1049,
https://doi.org/10.1007/978-3-031-08020-3_11

of resources. A system which mechanistically searches vast spaces,[1] oblivious to the ticking clock of its environment and the mortal concerns of its users is clearly the antithesis of intelligent. To the contrary, we have emphasized that the primary drivers for an intelligent system must, *by design*, be the asynchronous dictates of the user, continual changes in environment, and inevitable variations of resources; an intelligent system must respond gracefully to either in a timely manner.

'Work on Command' Manifests General Intelligence

Although Brooks emphasized that embodiment in real-world environments was essential, related work was predominantly characterized by being reactive. This therefore omits the kind of high-order cognition that we have argued is vital for sample efficiency. In contrast, the SCL approach reconciles deliberative planning with asynchronous and open-ended environments, whilst still preserving the anytime requirement of 'work on command'. In common with many cognitive scientists, we consider the ability to leverage and extend knowledge across domains to be synonymous with general intelligence [84, 148, 189]. We therefore claim that *any* meaningful notions of generality must also be grounded in a purpose-centric perspective. Just as Brooks has insisted that 'in vivo' experimentation in noisy and complex real-world settings is the proper framing for experimentation, we claim that the essential pragmatically meaningful setting for domain generalization is that of 'work on command'. In this setting, efficient knowledge transfer is then a prerequisite for the ability to complete related tasks in a timely manner. In Sect. 11.2.2 below, we consider the prospect that it may be possible to learn some 'universal' building blocks of compositional knowledge representation.

Understanding as Representation Manipulability

The SCL formulation effectively considers 'understanding' to be synonymous with 'representation manipulability' [364]. Formally, one can say that an agent A understands phenomenon F in context C iff A possesses a representation R of F, to which A could make local modifications, within constraints specified by C, thereby producing a representation R' of F, that enables inferences or manipulations of F towards goals specified by C.

What is therefore required is that representations R be 'sufficiently malleable' to act as a basis for construction, common usage, and novelty (in order of depth of understanding required). This malleability is a Gestalt property: metaphorically, it must preserve the 'essential nature' [148] of the representation, only allowing it to be perturbed in a way that describes a possible world.[2] As per the discussion in Sect. 9.2, the production of R' in this manner is thus synonymous with our notion of the functorial transformation of hypotheses. In this sense, SCL provides an exemplar for the research program proposed by Bundy and McNeill in 2006 [40]:

> *Reasoning systems must be able to develop, evolve, and repair their underlying representations as well as reason with them. The world changes too fast and too radically to rely on humans to patch the representations.*

[1] Whether for proofs of optimality or for solutions to a regression problem.

[2] As generalized from a priori seed constraints and the empirical experience of the system.

It is also illustrative to contrast this notion of ‘understanding’ with the weaker notion of ‘robustness’ in machine learning. Robustness is typically interpreted as (e.g. translation) invariance and/or noise tolerance, with these together proposed to guard against adversarial inputs in the general form of ‘single pixel noise’ [9]. Robustness interpreted as translation invariance would have a classifier say a table is a table even when it is upside down, although we understand that the essence of ‘tableness’ is that it ‘affords support’. Robustness in the sense of noise tolerance would have a classifier say a car without wheels is still a car, although humans immediately understand that does not afford forward motion (q.v. [338]).

Representational issues have been long debated in AI, but—as with many related issues such as the symbol grounding and frame problems—many ostensible problems can instead be considered to be artifacts of this kind of ‘overactive reification’. This is exacerbated in current practice by the supervised learning preoccupation with ‘noun-centric’ classification, in which objects are simply assigned a nominal category that is disconnected from the context of end usage. As per Wilkenfield [364], we therefore consider ‘understanding’ to be a *process*: contingent on the situated relationship between system and environment, cashed-out via demonstrations of ability on ‘analogous tasks’.

11.2 Research Topics

11.2.1 Choice of Expression Language

The power of reflective reasoning in SCL is determined by the choice of expression language. By virtue of inductive construction, the interpretation of recursive expressions described in Sect. 9.2 is guaranteed to terminate [334]. Naturally, if one were to elect to use an expression language with arbitrarily expressive *primitives*, the ability to reason about them is *formally* bounded by Gödel’s incompleteness theorems [115, 319].[3] In practice, the entities being reasoned about are grounded manipulations of the environment, rather than abstract formal proof objects, hence it is anticipated that relatively simple expression languages and learned denotational interpretations will suffice.

In accordance with the Curry–Howard isomorphism [176], a constrained subregion of the state space can alternatively be considered as a type T, with the constraints reflectively represented as predicates denoting the invariant properties of that type. The instantiation of an object of type T is therefore synonymous with a sequence of state space transformations, the result of which lies within a subregion of the state space defined by T. In programming language terms, there are a range of options for trading expressiveness of compositional properties against decidability and/or efficient synthesis procedures [263]. For example, the current notion of ‘differen-

[3] For completeness: some heuristics are also possible here [192].

tiable programming' can be represented via an expression language of 'higher-order functions which return locally-linear functions' [77, 78]. It has also recently been proposed [321] that the category **Poly** (of so-called *polynomial functors* a.k.a. *dependent lenses*) is particularly well-suited for representing both context-sensitive interaction and semantic closure.

Regarding abstraction, many variant algorithms have been devised for anti-unification, differing in the expressiveness of the underlying expression language. The simplest is *syntactic anti-unification*, which has complexity $\mathcal{O}(\max(|e_1|, |e_2|))$, where $|e|$ is the number of nodes in the abstract syntax tree (AST) of an expression e. Syntactic anti-unification is at its most useful whenever the expression language has a *normal form*, i.e., for any expression e, there is some sequence of transformations which yield a unique expression e' that acts as a representative for all such e. Languages such as the Simply Typed Lambda Calculus or System F have a normal form [265]. More generally, the existence of normal forms and Turing completeness are mutually exclusive; normal forms guarantee termination.

The Expression Language of 'Conceptual Spaces'

As an additional example to that of first-order linear arithmetic previously-described in Section , a simple but concrete example of a possible expression language is that of 'conceptual spaces' [103, 104], proposed by Gärdenfors as a bridge between symbolist and connectionist approaches. It is conjectured that naturally-occurring concepts are characterized by subspace regions with a topological structure that is connected and convex. For example, the topology of color corresponds to (some variant of) the color wheel, that of time to the real line. Qualitative notions (e.g. 'relative temperature') are supported via the topology of ordered intervals. Goguen has previously proposed a system which uses conceptual spaces for describing anthropocentric reasoning about space and time [118], observing:

> *Sensors, effectors, and world models ground elements of conceptual spaces in reality, where the world models are geometrical spaces. This implies that the symbol grounding problem is artificial, created by a desire for something that is not possible for purely symbolic systems, as in classic logic-based AI, but which is natural for situated systems.*

Compositionality of conceptual spaces has been explicitly studied: in recent work, Bolt et al. [28] employ category theory to perform compositional interpretation of natural language via a grammar over convex relations. This work is a foundation for future research into highly-expressive and principled compositionality. For SCL purposes, the 'expression language' of conceptual spaces is therefore that of affine transformations of convex (sub)regions of the state-space, and the associated truth predicate can always be determined as a function of the set of points in the intersection of two regions [342]. Expressing the operations of SCL (as described in Sect. 7.2) in the language of conceptual spaces requires much less sophisticated interpretation than does natural language, with geometric operations being sufficient to support these inference mechanisms.

11.2.2 Compositional Primitives

Previous chapters have argued that compositionality is key to addressing the foundational issues of generalization and representing long-tailed distributions with high sample efficiency. This suggests that what is desired are a collection of representations which form a 'basis set', the elements of which can be composed (perhaps nontrivially, via analogy) to describe a wide range of phenomena. We therefore claim that the strongest possible emphasis should be placed on the search for *compositional primitives*, i.e., compressed parametric representations of recurring phenomena, captured across multiple sensorimotor modalities. In cognitive science, such abstractions are known as *image schema* [159, 188] and are intended to represent common patterns in the embodied experience of space, force, motion, etc. Early work in this area induced image schema corresponding to spatial propositions from video data [96, 285]. There have also been attempts to model image schema symbolically [3, 139, 182], with recent work on a qualitative representation of containers in a sorted first order language [60]. It is clearly desirable that computational representations of image schema enjoy the cross-domain ubiquity ascribed to their cognitive counterparts. Concurrently with the development of the present work, a recent trends in deep learning is the proposed universality of so-called 'foundation models' [29] which provide a broad basis of representations for downstream tasks through large-scale self-supervised training. While this paradigm offers the advantage of well-known engineering pipelines, we saw in Chap. 4 that compositionality in the algebraic sense is essentially absent from deep learning, as considered across heterogeneous architectures and arbitrary constraints. Furthermore, the full grounding of language and other symbols will require representations which support strong invariant propagation and the ability to produce reasonable counterfactual statements. Since achieving the associated 'malleability of representation' enjoyed by humans has so far proved elusive, it is perhaps useful to focus initially on a related, but more overtly embodied notion: that of 'affordances'.

Affordances

The term 'affordance' was coined by Gibson [109] to describe a relation between an agent and its environment, grounded by the physical embodiment of the agent and the recognition capacity of its perception system:

> *If you know what can be done with a graspable detached object, what it can be used for, you can call it whatever you please. The theory of affordances rescues us from the philosophical muddle of assuming fixed classes of objects, each defined by its common features and then given a name.* [...] *But this does not mean you cannot learn how to use things and perceive their uses. You do not have to classify and label things in order to perceive what they afford.*

Although Gibson [109] initially described affordances as being 'directly perceivable', it is more useful for general intelligence purposes to equate (at least *ex nihilo*) perception of an affordance to be equivalent to hypothesis generation, i.e., to potentially require nontrivial computational work. Supporting evidence that affordances are anyway not 'directly perceivable':

- Tool use in crows, where previous work [301] implies that affordances are not *simply* a function of the relation between body and environment, and have (at the very least) a memetic component.
- Although the ability to create fire (from flint and tinder) or steel (from iron and carbon) could be said to be *inherent* in their component parts, their manufacture required nontrivial insight.

Hence affordances offer an overarching perspective on situated representations. We believe that composition of affordances is a key step towards general intelligence and that the category theoretic machinery of Sect. 9.1 provides a suitable framing for processes that abstract and generalize across complex configuration spaces. The initial research task is then to determine the right 'expression language' for describing the affordances of simple agents in simple domains (which are nonetheless 'noisy and real-world' [35]).

Subject to the ability to generalize from initial results to more complex domains, it is then appropriate to progress from explicitly agent-centric affordances to the more general patterns described as image schema, which might then have a greater prospect of being more independent of any specific embodied configuration. By these means, it may be possible to determine whether image schema do indeed exist as universal compositional primitives and whether—as has variously been suggested [145, 228, 229, 320]—analogy has a vital role as a universal mechanism for leveraging existing knowledge.

11.2.3 Links with Behavioral Control

For purposes of building links with existing approaches, it is illustrative to revisit SCL from the perspective of behavioural control of open systems. In this setting, the system can be seen as taking as input a continual stream of data, consisting of sensor and monitor states, and performing open-ended learning in the following manner:

- *Observe* some input and use it to progressively learn increasingly hierarchical SCL expressions.
- *Interpret* some applicable subset of SCL expressions to yield predictions and/or effector actions.
- *Feed-back* information about 'surprising' environmental transitions into the stateful parts of the observation and interpretation processes.

At the intersection of control theory and applied category theory, there is increasing interest in the *behavioral approach*, in which models are relations rather than merely functions. The methodological treatment due to Willems [365] comprises phases referred to as 'tearing, zooming, and linking'. 'Tearing' is the transformation of observed behavior into a collection of models, performed recursively ('zooming') until some elementary level of model complexity is reached. 'Linking' is then the composition of this collection of models. The process is refined until it can obtain predictions which match the observed behavior.

A previous category-theoretic treatment of control [15] has used relations on finite-dimensional vector spaces (the category $\mathbf{FinRel}_k$). The fact that relations are well-suited to capturing invariants is of particular interest for control and state estimation. Most interestingly, it has been shown that an analog of Lagrangean conservation laws [329] can be obtained in the very general setting of typed relations [10]. Relations also fit well with the 'inference as typed program synthesis' approach of SCL, since they are far better suited than functional descriptions for transforming specifications (e.g. the 'task language' of work on command) into implementation (the corresponding hypothesis chain) [289].

11.2.4 *Pragmatics via 'Causal Garbage Collection'*

As discussed in Sect. 9.2, systems that operate without 'closed-world' assumptions can always encounter transitions which confound the expectations of their world model. Since this necessitates some form of context-specific repair to the learned denotational semantics of the model, we term this to be a 'pragmatic' activity. We now describe a prospective means of applying a repair that is then made consistent across the entire ruleset of the world model.

For different kinds of algebraic structure, it is common to generalize the notion of 'basis set' familiar from linear algebra to that of 'generators and relators'. For example, C_n the cyclic group of order n has a single generator (g, say) and a single relator (an equation defining equivalence in the algebraic structure), $g^n = 1$, where 1 is the identity element of the group. This is notated as a so-called finite presentation: $\langle g \mid g^n = 1\rangle$. Similarly, the symmetry group of the square has presentation

$$\langle r, s \mid r^4 = s^2 = (sr)^2 = 1\rangle$$

where r corresponds to rotation by 90 degrees and s to reflection. The relator equations collectively define a *rewriting system* [12]. For certain classes of algebraic structure, this rewriting system can be iteratively applied to yield a *unique normal form* for any algebraic expression. Hence, since e.g. exponents in C_3 are modulo 3, then all of $g^5, g^8, g^{11} \ldots$ are rewritten to the normal form g^2.

With regard to control, Baez [15] similarly defines generators and relators for composition of expressions in $\mathbf{FinRel}_k$. Hence, a particular combination of elements may be reducible to a unique simpler representation. When the system is 'surprised' by a discrepancy between prediction and observation, it must be because the current interpretation of an expression does not accord with reality. This means that there are latent interactions which are not captured by the default denotational semantics. Hence, either an existing relator is invalid (as far as the world is concerned) or else there must be additional unknown relators. It is possible in principle to simultaneously make all existing invalid inferences consistent by constructing a new rewriting system. For certain classes of algebraic structure, this can achieved via the 'Knuth–Bendix Algorithm for Strings' [175]. This algorithm (strictly, procedure) is semidecidable; it will halt for all finitely presented algebras, but it is not possible to determine in advance how long this will take. One option would therefore be to run it as a background monitor or (typically) low-priority task, as a form of 'garbage collection' for causal inconsistencies.

11.3 Conclusion

Recent advances in machine learning have led to it being considered synonymous with the entirety of artificial intelligence, at least in popular conception. However, as exemplified by deep learning, this represents a very specific form of *program synthesis*, in which:

- Mission objectives/constraints are specified a priori.
- Voluminous data and potentially massive computational resources are available for training.
- The trained system is deployed into the production environment and remains unchanged thereafter.
- It is assumed that the training data/learning algorithm suffices for generalization to the production environment, even over time.

In order to solve a problem via machine learning, it is therefore necessary to impose strong a priori constraints, both on the design space and the production environment. Constraining the design space of the learner is almost always done via specialized human labor; for example, pruning the space of possible input features, crafting a reward/objective function, selecting and optimizing hyperparameters, etc. While objectives can readily be specified for simple domains (e.g. board games), in complex application domains such as those in the real world, the practical difficulties have caused initial high expectations (e.g. for autonomous vehicles) to be repeatedly revised downwards.

An additional vital concern for the artificial intelligence community is the increasing evidence that machine learning is not operating at the appropriate causal level. This is problematic since it is likely to lead to overfitting, something which is anyway encouraged by the trend for huge parameter spaces. More generally, machine learning is good at *manipulating* data, but this has not been demonstrated to lead to *understanding*, i.e., the ability to represent the space of possibilities spanned by the constraints that are latent in the training set. This has corresponding implications for robustness and safety which we discussed in detail. We have therefore argued that, in order for machine learning to progress, it must not only embrace a stronger notion of causality, but also embed it in a more comprehensive learning framework that supports reflective reasoning, which we term 'Semantically Closed Learning' (SCL).

It might nonetheless be claimed that the mechanisms of SCL are superfluous for general intelligence, given that reinforcement learning agents can be specified as "taking action so as to maximize an aggregate of future rewards, as a function of previous environmental observations” [143]. This statement is extremely broad, and could be argued to be AI-complete in capability. However, the broadness of the associated function signature does not automatically imbue RL with the required learning ability. Indeed, we argue that the interpretation of the above specification *via common practice* has canalized the expressiveness, generalization, and learning efficiency of RL. This canalization proceeds via:

- The assumption that rewards commensurate with general intelligence can meaningfully be specified a priori.
- The notion that feedback is best propagated via numeric (indeed, often scalar) rewards.
- The notion that ‘learn, then deploy’ is sufficient.

It could be argued that alternatives to each of these default assumptions have been separately explored at the boundaries of RL research. However:

- Default practice is so strongly culturally ingrained that it is necessary to explicitly delineate the alternatives.
- There is no singular framework that simultaneously moves beyond all of these assumptions in an integrated manner. If all these assumptions are simultaneously removed, we claim this inevitably requires that RL simultaneously integrates semantic closure, anytime-bounded rationality, and second-order automation, which then effectively makes the ‘new’ RL synonymous with the proposed compositional framework of Semantically Closed Learning.

We have presented a roadmap which re-asserts the importance of embodiment and the ‘Physical Grounding Hypothesis’ [35], i.e., the necessity of making decisions, anytime, in a complex, noisy environment. We strongly believe that the artificial intelligence community must finally embrace ‘the whole iguana’ [62] of general intelligence, i.e., to design systems which are capable of open-ended learning in inevitably-changing, real-world production environments, starting from minimal objectives. The value proposition for general intelligence is then the elimination of the need to specify meaningful reward functions upfront and maintain them in tandem with a changing environment, which cannot scale in practice.

Our concept of 2nd *order automation engineering* realizes a specific implementation of SCL as a minimal yet concrete technical design. Inspired by biological growth, where system agency is adapted and maintained despite evolutionary pressure, we have presented three automated procedures for autonomous system engineering: self-identification, synthesis, and maintenance *with guarantees*. These constitute the necessary developmental dynamics of machines designed to abide to the non-stationary arbitrary expression of value, conditioned by a reflective evaluation of risk. To lift this design to a fully developed general and *generative* system theory will be the continuation of our work. This will inevitably demand the departure from the prevalent and exclusively algorithmic (or computationalist) world-view, towards a science of self-organized, grounded, and constructive control processes.

Bibliography

1. P. Abbeel, A.Y. Ng, Apprenticeship learning via inverse reinforcement learning, in *Twenty-First International Conference on Machine Learning—ICML*'04 (ACM Press, 2004)
2. M. Alzantot, Y. Sharma, A. Elgohary, B.-J. Ho, M.B. Srivastava, K.-W. Chang, Generating natural language adversarial examples, in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, Oct. 31–Nov. 4, 2018*, ed. by E. Riloff, D. Chiang, J. Hockenmaier, J. Tsujii (Association for Computational Linguistics, 2018), pp. 2890–2896
3. S.R. Amant, T. Clayton, (Paul R. Cohen, and Carole R. Beal. An Image Schema Language, Morrison, Yu-Han Chang, Wei Mu, 2006)
4. D. Amodei, C. Olah, J. Steinhardt, P.F. Christiano, J. Schulman, D. Mané, Concrete Problems in AI Safety (2016). arxiv: abs/1606.06565
5. J. Andreas, Measuring compositionality in representation learning, in *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, 6–9 May 2019* (2019). https://www.OpenReview.net
6. M. Andrychowicz, D. Crow, A. Ray, J. Schneider, R. Fong, P. Welinder, B. McGrew, J. Tobin, P. Abbeel, W. Zaremba. Hindsight experience replay, in *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4–9 Dec. 2017, Long Beach, CA, USA*, ed. by I. Guyon, U. von Luxburg, S. Bengio, H.M. Wallach, R. Fergus, S.V.N. Vishwanathan, R. Garnett, pp. 5048–5058 (2017)
7. F. Arabshahi, Z. Lu, S. Singh, A. Anandkumar, *Memory Augmented Recursive Neural Networks* (2015). arxiv:abs/1911.01545
8. W.R. Ashby, *Design for a Brain* (Wiley, Science Paperbacks, 1960)
9. A. Athalye, L. Engstrom, A. Ilyas, K. Kwok, Synthesizing robust adversarial examples, in *Proceedings of the 35th International Conference on Machine Learning*, *Proceedings of Machine Learning Research*, PMLR, Stockholm, Sweden, ed. by J. Dy, A. Krause, vol. 80, 10–15 Jul. 2018, pp. 284–293
10. R. Atkey, From parametricity to conservation laws, via Noether's theorem, in *The 41st Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL'14, San Diego, CA, USA*, vol. 20–21, pp. 491–502, Jan. 2014
11. T. Atkinson, H. Baier, T. Copplestone, S. Devlin, J. Swan, The text-based adventure AI competition. IEEE Trans. Games **11**(3), 260–266 (2019)
12. F. Baader, T. Nipkow, *Term Rewriting and All That* (Cambridge University Press, 1999)
13. P.-L. Bacon, J. Harb, D. Precup, *The Option-Critic Architecture* (2016). arxiv: bs/1609.05140

J. Swan et al., *The Road to General Intelligence*,
Studies in Computational Intelligence 1049,
https://doi.org/10.1007/978-3-031-08020-3

14. A.P. Badia, B. Piot, S. Kapturowski, P. Sprechmann, A. Vitvitskyi, Z.D. Guo, C. Blundell, *Agent57: Outperforming the Atari Human Benchmark* (2020). arxiv:abs/2003.13350
15. J.C. Baez, J. Erbele, Categ. Control **1405**, 6881 (2015)
16. D. Bahdanau, K. Cho, Y. Bengio, *Neural Machine Translation by Jointly Learning to Align and Translate* (2014). arxiv:abs/1409.0473
17. T. Bansal, J. Pachocki, S. Sidor, I. Sutskever, I. Mordatch, Emergent complexity via multi-agent competition. In: *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, 30 Apr.–3 May 2018, Conference Track Proceedings* (2018). https://www.OpenReview.net
18. R. Battiti, G. Tecchiolli, The reactive tabu search. Informs. J. Comput. **6**(2), 126–140 (1994)
19. R. Battiti, G. Tecchiolli, The continuous reactive tabu search: blending combinatorial optimization and stochastic search for global optimization. Ann. Oper. Res. **63**(2), 151–188 (1996)
20. F. Bellas, R.J. Duro, A. Faina, D. Souto, Multilevel Darwinist Brain (MDB): artificial evolution in a cognitive architecture for real robots. IEEE Trans. Auton. Mental Dev. **2**(4), 340–354 (2010)
21. Y. Bengio, Y. Lecun, G. Hinton, Deep learning for AI. Commun. ACM **64**(7), 58–65 (2021)
22. C. Berner, G. Brockman, B. Chan, V. Cheung, P. Debiak, C. Dennison, D. Farhi, Q. Fischer, S. Hashme, C. Hesse, R. Józefowicz, S. Gray, C. Olsson, J. Pachocki, M. Petrov, H. Pondé de Oliveira Pinto, J. Raiman, T. Salimans, J. Schlatter, J. Schneider, S. Sidor, I. Sutskever, J. Tang, F. Wolski, S. Zhang, *Dota 2 with Large Scale Deep Reinforcement Learning* (2019). arxiv:abs/1912.06680
23. T.R. Besold, A.S. d'Avila Garcez, S. Bader, H. Bowman, P.M. Domingos, P. Hitzler, K.-U. Kühnberger, L.C. Lamb, D. Lowd, P.M.V. Lima, L. de Penning, G. Pinkas, H. Poon, G. Zaverucha, *Neural-Symbolic Learning and Reasoning: a Survey and Interpretation* (2017). arxiv:abs/1711.03902
24. H. Bharadhwaj, A. Kumar, N. Rhinehart, S. Levine, F. Shkurti, A. Garg, Conserv. Saf. Crit. Explor. **2010**, 14497 (2020)
25. C.M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics)* (Springer, Berlin, Heidelberg, 2006)
26. C. Blum, A. Roli, Metaheuristics in combinatorial optimization: overview and conceptual comparison. ACM Comput. Surv. **35**(3), 268–308 (2003)
27. M.S. Boddy, T. Dean, Deliberation scheduling for problem solving in time-constrained environments. Artif. Intell. **67**, 245–285 (1994)
28. J. Bolt, B. Coecke, F. Genovese, M. Lewis, D. Marsden, R. Piedeleu, *Interacting Conceptual Spaces I : Grammatical Composition of Concepts* (2017). arxiv:abs/1703.08314
29. R. Bommasani, D.A. Hudson, E. Adeli, R. Altman, S. Arora, S. von Arx, M.S. Bernstein, J. Bohg, A. Bosselut, E. Brunskill, E. Brynjolfsson, S. Buch, D. Card, R. Castellon, N. Chatterji, A. Chen, K. Creel, J.Q. Davis, D. Demszky, C. Donahue, M. Doumbouya, E. Durmus, S. Ermon, J. Etchemendy, K. Ethayarajh, L. Fei-Fei, C. Finn, T. Gale, L. Gillespie, K. Goel, N. Goodman, S. Grossman, N. Guha, T. Hashimoto, P. Henderson, J. Hewitt, D.E. Ho, J. Hong, K. Hsu, J. Huang, T. Icard, S. Jain, D. Jurafsky, P. Kalluri, S. Karamcheti, G. Keeling, F. Khani, O. Khattab, P.W. Koh, M. Krass, R. Krishna, R. Kuditipudi, A. Kumar, F. Ladhak, M. Lee, T. Lee, J. Leskovec, I. Levent, X.L. Li, X. Li, T. Ma, A. Malik, C.D. Manning, S. Mirchandani, E. Mitchell, Z. Munyikwa, S. Nair, A. Narayan, D. Narayanan, B. Newman, A. Nie, J.C. Niebles, H. Nilforoshan, J. Nyarko, G. Ogut, L. Orr, I. Papadimitriou, J.S. Park, C. Piech, E. Portelance, C. Potts, A. Raghunathan, R. Reich, H. Ren, F. Rong, Y. Roohani, C. Ruiz, J. Ryan, C. Ré, D. Sadigh, S. Sagawa, K. Santhanam, A. Shih, K. Srinivasan, A. Tamkin, R. Taori, A.W. Thomas, F. Tramèr, R.E. Wang, W. Wang, B. Wu, J. Wu, Y. Wu, S.M. Xie, M. Yasunaga, J. You, M. Zaharia, M. Zhang, T. Zhang, X. Zhang, Y. Zhang, L. Zheng, K. Zhou, P. Liang, On Oppor. Risks Found. Models **108**, 07258 (2021)
30. N. Bostrom, Ethical issues in advanced artificial intelligence, in *Science Fiction and Philosophy: From Time Travel to Superintelligence*, ed by S. Schneider (Wiley, 2009)
31. K. Bousmalis, A. Irpan, P. Wohlhart, Y. Bai, M. Kelcey, M. Kalakrishnan, L. Downs, J. Ibarz, P. Pastor, K. Konolige, S. Levine, V. Vanhoucke, Using simulation and domain adaptation

to improve efficiency of deep robotic grasping, in *2018 IEEE International Conference on Robotics and Automation, ICRA 2018, Brisbane, Australia, 21–25 May 2018* (IEEE, 2018) pp. 4243–4250

32. B. Brembs, Operant reward learning in Aplysia. Curr. Dir. Psychol. Sci. **12**(6), 218–221 (2003). https://doi.org/10.1046/j.0963-7214.2003.01265.x
33. M. Bromberger. *Decision Procedures for Linear Arithmetic. (Quelques procédures de décision pour l'arithmétique linéaire)*. Ph.D. thesis, Saarland University, Saarbrücken, Germany (2019)
34. R.A. Brooks, *Cambrian Intelligence: The Early History of the New AI* (MIT Press, Bradford Book, 1999)
35. R.A. Brooks, Elephants don't play chess. Robot. Auton. Syst. **6**(1–2), 3–15 (1990)
36. R.A. Brooks, *A Robust Layered Control System for a Mobile Robot* (MIT Press, Cambridge, MA, USA, 1991), pp. 2–27
37. R.A. Brooks, Intelligence without representation. Artif. Intell. **47**(1), 139–159 (1991)
38. G. Bruce, *Buchanan and Joshua Lederberg* (The Heuristic DENDRAL Program for Explaining Empirical Data. Technical report, Stanford, CA, USA, 1971)
39. P.E. Bulychev, E.V. Kostylev, V.A. Zakharov, Anti-Unification algorithms and their applications in program analysis, in *Proceedings of the 7th International Andrei Ershov Memorial Conference on Perspectives of Systems Informatics*, PSI'09, Berlin, Heidelberg (Springer, 2009), pp. 413–423
40. A. Bundy, F. McNeill, Representation as a fluent: an AI challenge for the next half century. IEEE Intell. Syst. **21**(3), 85–87 (2006)
41. J. Cai, R. Shin, D. Song, Making neural programming architectures generalize via recursion, in *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, 24–26 Apr. 2017, Conference Track Proceedings* (2017). https://www.OpenReview.net
42. M. Capucci, B. Gavranović, J. Hedges, E.F. Rischel, Towards Found. Categ. Cybern. **2105**, 06332 (2021)
43. G.J. Chaitin, *Algorithmic Information Theory*, *Cambridge tracts in theoretical computer science*, vol. 1 (Cambridge University Press, 1987)
44. O. Chang, Maschinen, Die (Nicht) Denken (2021). https://www.forbes.at/artikel/maschinen-die-nicht-denken.html. Accessed 3rd Sept. 2021
45. W.-L. Chao, H.-J. Ye, D.-C. Zhan, M. Campbell, K.Q. Weinberger, *A Meta Understanding of Meta-Learning* (2019)
46. X. Chen, C. Liu, D. Song, Towards synthesizing complex programs from input-output examples, in *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, 30 Apr.–3 May, 2018, Conference Track Proceedings* (2018). https://www.OpenReview.net
47. A. Church, A formulation of the simple theory of types. J. Symb. Log **5**(2), 56–68 (1940)
48. M. Ciccone, M. Gallieri, J. Masci, C. Osendorfer, F.J. Gomez, *Nais-net: Stable Deep Networks from Non-autonomous Differential Equations* (2018). arxiv:abs/1804.07209
49. P. Cisek, Beyond the computer metaphor: behavior as interaction. J. Conscious. Stud. **6**(11), 125–142 (1999)
50. A. Clark, D. Chalmers, The extended mind. Analysis **58**(1), 7–19 (1998)
51. E.B. Clark, S.J. Hickinbotham, S. Susan, Semantic closure demonstrated by the evolution of a universal constructor architecture in an artificial chemistry. J. R. Soc. Interf. (2017)
52. H.P. Cooke, H. Tredennick, *Aristotle: The Organon* (Harvard University Press, 1938)
53. R. Cremonini, K. Marriott, H. Søndergaard, A general theory for abstraction, in *AI '90: Proceedings of the Fourth Australian Joint Conferences on Artificial Intelligence*, ed. by C.P. Tsang (World Scientific Publishing, 1990), pp. 121–134
54. L.R. Crole, *Categories for Types* (Cambridge University Press, 1994)
55. R. Csordás, J. Schmidhuber, *Improving Differentiable Neural Computers Through Memory Masking, De-allocation, and Link Distribution Sharpness Control* (2019). arxiv:abs/1904.10278
56. G. Cybenko, Approximation by superpositions of a sigmoidal function. Math. Control Signals Syst. **2**(4), 303–314 (1989)

57. A. D'Amour, K. Heller, D. Moldovan, B. Adlam, B. Alipanahi, A. Beutel, C. Chen, J. Deaton, J. Eisenstein, M.D. Hoffman, F. Hormozdiari, N. Houlsby, S. Hou, G. Jerfel, A. Karthikesalingam, M. Lucic, Y. Ma, C. McLean, D. Mincu, A. Mitani, A. Montanari, Z. Nado, V. Natarajan, C. Nielson, T.F. Osborne, R. Raman, K. Ramasamy, R. Sayres, J. Schrouff, M. Seneviratne, S. Sequeira, H. Suresh, V. Veitch, M. Vladymyrov, X. Wang, K. Webster, S. Yadlowsky, T. Yun, X. Zhai, D. Sculley, Underspecification Presents Challenges for Credibility in Modern. Machine Learning **2011**, 03395 (2020)
58. I. Dasgupta, D. Guo, A. Stuhlmüller, S. Gershman, N.D. Goodman, Evaluating Compositionality in Sentence Embeddings, in *Proceedings of the 40th Annual Meeting of the Cognitive Science Society, CogSci 2018, Madison, WI, USA, 25–28 Jul 2018*, ed. by C. Kalish, M.A. Rau, X. (Jerry) Zhu, T.T. Rogers (2018). https://www.cognitivesciencesociety.org
59. A. d'Avila Garcez, L.C. Lamb, *Neurosymbolic AI: The 3rd Wave* (2020). arxiv:2012.05876
60. E. Davis, G. Marcus, N. Frazier-Logue, Commonsense reasoning about containers using radically incomplete information. Artif. Intell. **248**, 46–84 (2017)
61. H.W. de Regt, The epistemic value of understanding. Philosop. Sci. **76**, 585–597 (2009)
62. D.C. Dennett, Why not the whole Iguana? Behav. Brain Sci. **1**(1), 103–104 (1978)
63. D. Deutsch, *The Fabric of Reality* (Penguin Books Limited, 2011)
64. D. Deutsch, The logic of experimental tests, particularly of everettian quantum theory. Stud. Hist. Philosop. Sci. Part B: Stud. Hist. Philos. Mod. Phys. **55**, 08 (2015)
65. C. Devin, D. Geng, P. Abbeel, T. Darrell, S. Levine, Compositional plan vectors, in *Advances in Neural Information Processing Systems*, ed. by H. Wallach, H. Larochelle, A. Beygelzimer, F. d' Alché-Buc, E. Fox, R. Garnett, vol. 32 (Curran Associates, Inc., 2019), pp. 14963–14974
66. J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, BERT: pre-training of deep bidirectional transformers for language understanding, in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, 2–7 June 2019 (Long and Short Papers)*, ed. by J. Burstein, C. Doran, T. Solorio, vol. 1 (Association for Computational Linguistics, 2019), pp. 4171–4186
67. J. Dewey, The reflex arc concept in psychology. Psychol. Rev. **3**(4), 357–370 (1896)
68. S. Doncieux, N. Bredèche, L.K.L. Goff, B. Girard, A. Coninx, O. Sigaud, M. Khamassi, N.D. Rodríguez, D. Filliat, T.M. Hospedales, A. Eiben, R. Duro, *DREAM Architecture: a Developmental Approach to Open-Ended Learning in Robotics* (2020). arxiv:abs/2005.06223
69. H. Dong, J. Mao, T. Lin, C. Wang, L. Li, D. Zhou, *Neural Logic Machines* (2019). arxiv:abs/1904.11694
70. M. Dorigo, T. Stützle, *Ant Colony Optimization* (Bradford Company, USA, 2004)
71. L. Gary, *Drescher* (MIT Press, Made-Up Minds—A Constructivist Approach to Artificial Intelligence, 1991)
72. F. Drewes, B. Hoffmann, D. Plump, Hierarchical graph transformation. J. Comput. Syst. Sci. **64**(2), 249–283 (2002)
73. H.L. Dreyfus, A history of first step fallacies. Minds Mach. **22**(2), 87–99 (2012)
74. J. Drolet, C.L. Moodie, B. Montreuil, Scheduling factories of the future. J. Mech. Work. Technol. **20**, 183–194 (1989)
75. E. Eaton, T. Dietterich, M. Gini, B.J. Grosz, C.L. Isbell, S. Kambhampati, M. Littman, F. Rossi, S. Russell, P. Stone, T. Walsh, M. Wooldridge, Who speaks for AI? AI Matters **2**(2), 4–14 (2016)
76. H. Ehrig, K. Ehrig, U. Prange, G. Taentzer, *Fundamentals of Algebraic Graph Transformation* (2006)
77. C. Elliott, Compiling to categories, in *Proceedings of the ACM on Programming Languages (ICFP)* (2017)
78. C. Elliott, The simple essence of automatic differentiation. Proc. ACM Program. Lang. **2**(ICFP) (2018)
79. D.C. Elton, Applying Deutsch's concept of good explanations to artificial intelligence and neuroscience–an initial exploration. Cogn. Syst. Res. **67**, 9–17 (2021)

80. O. Evans, A. Stuhlmüller, N.D. Goodman, Learning the preferences of ignorant, inconsistent agents, in *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, 12–17 Feb. 2016, Phoenix, Arizona, USA*, ed. by D. Schuurmans, M.P. Wellman (AAAI Press, 2016), pp. 323–329
81. U. Evci, F. Pedregosa, A. Gomez, E. Elsen, Difficul. Train. Sparse. Neural Netw. **1906**, 10732 (2020)
82. T. Everitt, M. Hutter, Avoiding wireheading with value reinforcement learning, in: *Artificial General Intelligence—9th International Conference, AGI 2016, New York, NY, USA, 16–19 Jul 2016*. Lecture Notes in Computer Science, vol. 9782 (Springer, 2016), pp. 12–22
83. B. Eysenbach, S. Gu, J. Ibarz, S. Levine, Leave no trace: learning to reset for safe and autonomous reinforcement learning, in *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, Conference Track Proceedings*, 30 Apr.–3 May 2018 . https://www.OpenReview.net
84. G. Fauconnier, M. Turner. *The Way We Think: Conceptual Blending and the Mind's Hidden Complexities* (Basic Books, 2002)
85. M. Fernández, H. Kirchner, B. Pinaud, Strategic port graph rewriting: an interactive modelling framework. Math. Struct. Comput. Sci. **29**(5), 615–662 (2019)
86. C. Finn, P. Abbeel, S. Levine, Model-Agnostic meta-learning for fast adaptation of deep networks, in *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6–11 Aug. 2017*, *Proceedings of Machine Learning Research* PMLR, ed. by D. Precup, Y.W. Teh, vol. 70, pp. 1126–1135 (2017)
87. J.A. Fodor, *The Language of Thought*. Language and Thought Series (Harvard University Press, 1975)
88. B. Fong, M. Johnson, Lenses and learners, in *Proceedings of Bx 2019, CEUR Workshop Proceedings*, vol. 2355, pp. 16–29 (2019)
89. B. Fong, D. Spivak, *An Invitation to Applied Category Theory: Seven Sketches in Compositionality* (Cambridge University Press, 2019)
90. B. Fong, D.I. Spivak, R. Tuyéras, Backprop as functor: a compos. Perspect. Superv. Learn. **1711**, 10455 (2017)
91. N. Foster, M. Greenwald, J. Moore, B. Pierce, A. Schmitt, Combinators for bi-directional tree transformations: a linguistic approach to the view update problem. ACM Trans. Programm. Lang. Syst. **29**(3) (2007)
92. J. Franke, J. Niehues, A. Waibel, Robust and scalable differentiable neural computer for question answering, in *Proceedings of the Workshop on Machine Reading for Question Answering@ACL 2018, Melbourne, Australia*, 19 Jul 2018, ed. by E. Choi, M. Seo, D. Chen, R. Jia, J. Berant (Association for Computational Linguistics, 2018), pp. 47–59
93. J. Frankle, M. Carbin, The lottery ticket hypothesis: finding sparse, trainable neural networks, in *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, 6–9 May 2019*. https://www.OpenReview.net
94. K. Frans, J. Ho, X. Chen, P. Abbeel, J. Schulman, Meta learning shared hierarchies, in *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, 30 Apr.–3 May 2018, Conference Track Proceedings* (2018). https://www.OpenReview.net
95. R.M. French *The Subtlety of Sameness: A Theory and Computer Model of Analogy-Making* (The MIT Press, 1995)
96. R.M. French, Constrained connectionism and the limits of human semantics. Philosop. Psychol. **12**(4), 515–523 (1999)
97. R.M. French, P. Anselme, Interactively converging on context-sensitive representations: a solution to the frame problem. Revue Internationale de Philosophie **53**(209 (3)), 365–385 (1999)
98. T. Fritz, A synthetic approach to Markov kernels, conditional independence and theorems on sufficient statistics. Adv. Math. **370** (2020)
99. T. Gale, E. Elsen, S. Hooker, State sparsity deep. Neural Netw. **1902**, 09574 (2019)
100. M. Gallieri, S.S.M. Salehian, N.E. Toklu, A. Quaglino, J. Masci, J. Koutník, F.J. Gomez, *Safe Interactive Model-Based Learning* (2019). arxiv:abs/1911.06556

101. E. Gamma, R. Helm, R. Johnson, J.M. Vlissides, *Design Patterns: Elements of Reusable Object-Oriented Software* (Addison-Wesley Professional, 1 edn., 1994)
102. Y. Ganin, V.S. Lempitsky, Unsupervised domain adaptation by backpropagation, in *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France*, 6–11 Jul 2015, *JMLR Workshop and Conference Proceedings*, ed. by F.R. Bach, D.M. Blei, vol. 37, pp. 1180–1189 (2015). https://www.JMLR.org
103. P. Gärdenfors, *The Geometry of Meaning: Semantics Based on Conceptual Spaces* (The MIT Press, MIT Press, 2014)
104. P. Gärdenfors, *Conceptual Spaces: The Geometry of Thought* (MIT Press, Cambridge, MA, USA, 2000)
105. M. Garnelo, K. Arulkumaran, M. Shanahan, *Towards Deep Symbolic Reinforcement Learning* (2016). arxiv:abs/1609.05518
106. D. Gentner, K.D. Forbus, Computational models of analogy. Wiley Interdiscip. Rev.: Cogn. Sci. **2**(3), 266–276 (2011)
107. D. Gentner, C. Hoyos, Analogy and abstraction. Top. Cogn. Sci **9**(3):672–693 (2017). https://onlinelibrary.wiley.com, https://doi.org/10.1111/tops.12278
108. N. Ghani, J. Hedges, V. Winschel, P. Zahn, Compositional game theory, in *Proceedings of Logic in Computer Science (LiCS) 2018* (ACM, 2018), pp. 472–481
109. J.J. Gibson, *The Ecological Approach to Visual Perception* (Houghton Mifflin, 1979)
110. M. Giry, A categorical approach to probability theory. *Categorical Aspects of Topology and Analysis*, pp. 68–85 (1982)
111. F. Giunchiglia, T. Walsh, A theory of abstraction. Artif. Intell. **57**(2), 323–389 (1992)
112. A. Gleave, M. Dennis, N. Kant, C. Wild, S. Levine, S. Russell, *Adversarial Policies: Attacking Deep Reinforcement Learning* (2019). arxiv:abs/1905.10615
113. F. Glover, M. Laguna, *Tabu Search* (Kluwer Academic Publishers, Norwell, MA, USA, 1997)
114. F. Gobet, P.C.R. Lane, S. Croker, P.C.-H. Cheng, G. Jones, I. Oliver, J.M. Pine, Chunking mechanisms in human learning. Trends Cogn. Sci. **5**(6), 236–243 (2001)
115. K. Gödel, Über formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme I. Monatshefte für Mathematik und Physik **38**(1), 173–198 (1931)
116. B. Goertzel, *Patterns of Cognition: Cognitive Algorithms as Galois Connections Fulfilled by Chronomorphisms On Probabilistically Typed Metagraphs* (2021). arxiv:abs/2102.10581
117. J. Goguen, An introduction to algebraic semiotics, with application to user interface design, in *Computation for Metaphors, Analogy, and Agents, Berlin, Heidelberg*. ed. by C.L. Nehaniv (Springer, Berlin, Heidelberg, 1999), pp. 242–291
118. J. Goguen, Mathematical models of cognitive space and time, in *Reasoning and Cognition*, ed. by D. Andler, M. Okada, I. Watanabe, pp. 125–128 (2006)
119. J.A. Goguen, What is unification?—A categorical view of substitution, equation and solution, in *Resolution of Equations in Algebraic Structures, Algebraic Techniques*, vol. 1 (Academic, 1989), , pp. 217–261
120. J.I. Goodfellow, Y. Bengio, A. Courville, *Deep Learning* (MIT Press, Cambridge, MA, USA, 2016)
121. I.J. Goodfellow, J. Shlens, C. Szegedy, Explaining and harnessing adversarial examples, in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, 7–9 May 2015*, *Conference Track Proceedings*, ed. by Y. Bengio, Y. LeCun (2015)
122. C.A.E. Goodhart, *Problems of Monetary Management: the UK experience, in Monetary Theory and Practice* (Macmillan Education, UK, 1984), pp. 91–121
123. A. Graves, G. Wayne, I. Danihelka, *Neural Turing Machines* (2014). arxiv:abs/1410.5401
124. A. Graves, G. Wayne, M. Reynolds, T. Harley, I. Danihelka, A. Grabska-Barwińska, S.G. Colmenarejo, E. Grefenstette, T. Ramalho, J. Agapiou, A.P. Badia, K.M. Hermann, Y. Zwols, G. Ostrovski, A. Cain, H. King, C. Summerfield, P. Blunsom, K. Kavukcuoglu, D. Hassabis, Hybrid computing using a neural network with dynamic external memory. Nature **538**(7626), 471–476 (2016)
125. K. Guu, K. Lee, Z. Tung, P. Pasupat, M.-W. Chang, *REALM: Retrieval-Augmented Language Model Pre-Training*. arxiv:abs/2002.08909

126. A. Habel, D. Plump, Relabelling in graph transformation, in *International Conference on Graph Transformation* (Springer, 2002), pp. 135–147
127. R. Hackforth, *Plato: Phaedrus* (Cambridge University Press, 1972)
128. D. Hadfield-Menell, A.D. Dragan, P. Abbeel, S.J. Russell, The off-switch game, in *The Workshops of the The Thirty-First AAAI Conference on Artificial Intelligence, Saturday, 4–9 Feb. 2017, San Francisco, California, USA*, *AAAI Workshops*, vol. WS-17 (AAAI Press, 2017)
129. D. Hadfield-Menell, S. Milli, P. Abbeel, S.J. Russell, A.D. Dragan, Inverse reward design, in *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4–9 Dec. 2017, Long Beach, CA, USA*, ed. by I. Guyon, U. von Luxburg, S. Bengio, H.M. Wallach, R. Fergus, S.V.N. Vishwanathan, R. Garnett, pp. 6765–6774 (2017)
130. D. Hadfield-Menell, S.J. Russell, P. Abbeel, A.D. Dragan, Cooperative inverse reinforcement learning, in *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, 5–10 Dec. 2016, Barcelona, Spain*, ed. by D.D. Lee, M. Sugiyama, U. von Luxburg, I. Guyon, R. Garnett, pp. 3909–3917 (2016)
131. E. Hainry, Reachability in linear dynamical systems, in *Logic and Theory of Algorithms, Berlin, Heidelberg*. ed. by A. Beckmann, C. Dimitracopoulos, B. Löwe (Springer, Heidelberg, Berlin, 2018), pp. 241–250
132. K. Hao, *AI is Dending People to Jail-and Getting it Wrong*, Apr. 2020. https://www.technologyreview.com/2019/01/21/137783/algorithms-criminal-justice-ai/
133. Y.N. Harari. *Sapiens: A Brief History of Humankind* (Harvill Secker, 2014)
134. S. Harnad, The symbol grounding problem. Physica D **42**, 335–346 (1990)
135. P.E. Hart, R.O. Duda, Prospector–A computer based consultation system for mineral exploration. Technical report 155, AI Center, SRI International, 333 Ravenswood Ave, Menlo Park, CA 94025, Oct 1977
136. M. Haruno, D.M. Wolpert, M. Kawato, Multiple paired forward-inverse models for human motor learning and control, in *Advances in Neural Information Processing Systems 11, [NIPS Conference, Denver, Colorado, USA, 30 Nov.–5 Dec. 1998]*, ed. by M.J. Kearns, S.A. Solla, D.A. Cohn, pp. 31–37 (The MIT Press, 1998)
137. J. Haugeland, *Artificial Intelligence: The Very Idea* (Massachusetts Institute of Technology, USA, 1985)
138. N. Hay, M. Stark, A. Schlegel, C. Wendelken, D. Park, E. Purdy, T. Silver, D. Scott Phoenix, D. George, Behavior is everything: towards representing concepts with sensorimotor contingencies, in *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, 2–7 Feb. 2018*, ed. by S.A. McIlraith, K.Q. Weinberger (AAAI Press, 2018), pp. 1861–1870
139. M.M. Hedblom, O. Kutz, R. Peñaloza, G. Guizzardi, Image schema combinations and complex events. KI - Künstliche Intelligenz **33**(3), 279–291 (2019)
140. J. Hedges, *Categorical Cybernetics: A Manifesto* (2019). https://julesh.com/2019/11/27/categorical-cybernetics-a-manifesto. Accessed 16th Sept 2021
141. M. Heßler, *Mensch-Maschine-Interaktion: Handbuch zu Geschichte—Kultur—Ethik*, Chapter Industrie 4.0 (J.B. Metzler, Stuttgart, 2019), pp. 269–271
142. C. Hewitt, PLANNER: a language for proving theorems in robots, in *Proceedings of the 1st International Joint Conference on Artificial Intelligence, Washington, DC, USA, 7–9 May 1969*, ed. by D.E. Walker, L.M. Norton (William Kaufmann, 1969), pp. 295–302
143. M. Hoffman, B. Shahriari, J. Aslanides, G. Barth-Maron, F. Behbahani, T. Norman, A. Abdolmaleki, A. Cassirer, F. Yang, K. Baumli, S. Henderson, A. Novikov, S.G. Colmenarejo, S. Cabi, Ç. Gülçehre, T. Le Paine, A. Cowie, Z. Wang, B. Piot, N. de Freitas, *Acme: A Research Framework for Distributed Reinforcement Learning* (2020). arxiv:abs/2006.00979
144. D. Hofstadter, *Fluid Concepts and Creative Analogies: Computer Models of the Fundamental Mechanisms of Thought, chapter To Seek Whence Cometh a Sequence* (Basic Books Inc, USA, 1995), pp. 13–86

145. D. Hofstadter, Analogy as the core of cognition, in *The Analogical Mind: Perspectives from Cognitive Science*, ed. by D. Gentner, K.J. Holyoak, B.N. Kokinov. (MIT Press, 2001), pp. 499–538
146. D. Hofstadter, M. Mitchell, *Fluid Concepts and Creative Analogies: Computer Models of the Fundamental Mechanisms of Thought, chapter The Copycat Project: A Model of Mental Fluidity and Analogy-Making* (Basic Books Inc, USA, 1995), pp. 205–267
147. D.R. Hofstadter, *Godel, Escher, Bach: An Eternal Golden Braid* (Basic Books Inc, USA, 1979)
148. D.R. Hofstadter, E. Sander. *Surfaces and Essences: Analogy as the Fuel and Fire of Thinking* (Basic Books, 2013)
149. J.H. Holland, K.J. Holyoak, R.E. Nisbett, P.R. Thagard, *Induction: Processes of Inference, Learning, and Discovery* (MIT Press, Cambridge, MA, USA, 1986)
150. E.J. Horvitz, G. Rutledge, Time-Depend. Util. Action Under Uncertain. **1303**, 5722 (2013)
151. S.H. Huang, N. Papernot, I.J. Goodfellow, Y. Duan, P. Abbeel, Adversarial attacks on neural network policies, in *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, Apr 2017, Workshop Track Proceedings* (2017). https://www.OpenReview.net
152. D.A. Hudson, C.D. Manning, Learning by abstraction: the neural state machine, in *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8–14 Dec. 2019, Vancouver, BC, Canada*, ed. by H.M. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E.B. Fox, R. Garnett, pp. 5901–5914 (2019)
153. M. Hutter, *A Theory of Universal Artificial Intelligence based on Algorithmic Complexity* (2000). arxiv:cs.AI/0004001
154. G. Hutton, A Tutorial on the Universality and Expressiveness of Fold. J. Funct. Program. **9**(4), 355–372 (1999)
155. F.N. Iandola, M.W. Moskewicz, K. Ashraf, S. Han, W.J. Dally, K. Keutzer, *SqueezeNet: AlexNet-level Accuracy with 50x Fewer Parameters and* <1MB *Model Size* (2016). arxiv:abs/1602.07360
156. M. Jaderberg, V. Dalibard, S. Osindero, W.M. Czarnecki, J. Donahue, A. Razavi, O. Vinyals, T. Green, I. Dunning, K. Simonyan, C. Fernando, K. Kavukcuoglu, *Population Based Training of Neural Networks* (2017). arxiv:abs/1711.09846
157. R. Jhala, N. Vazou, Refin. Types: A Tutor. **2010**, 07763 (2020)
158. R. Jia, P. Liang, Adversarial examples for evaluating reading comprehension systems, in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, 9–11 Sep. 2017*, ed. by M. Palmer, R. Hwa, S. Riedel (Association for Computational Linguistics, 2017), pp. 2021–2031
159. M. Johnson, *The Body in the Mind: The Bodily Basis of Meaning, Imagination, and Reason* (University of Chicago Press, 1987)
160. J.P.J. Neto, Solving the nonlinear pendulum equation with nonhomogeneous initial conditions. Int. J. Apll. Math. **30**, 06 (2017)
161. L. Kaelbling, Learning to achieve goals. IJCAI (1993)
162. W. Kahl, *Finite Limits and Anti-unification in Substitution Categories*, pp. 87–102, June 2019
163. L. Kaiser, I. Sutskever, Neural GPUs learn algorithms, in *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, 2–4 May 2016, Conference Track Proceedings*, ed. by Y. Bengio, Y. LeCun (2016)
164. K. Kansky, T. Silver, D.A. Mély, M. Eldawy, M. Lázaro-Gredilla, X. Lou, N. Dorfman, S. Sidor, D. Scott Phoenix, D. George, Schema networks: zero-shot transfer with a generative causal model of intuitive physics, in *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6–11 Aug. 2017*, *Proceedings of Machine Learning Research, PMLR*, ed. by D. Precup, Y.W. Teh, vol. 70, pp. 1809–1818 (2017)
165. J. Kaplan, Artificial intelligence: think again. Commun. ACM **60**(1), 36–38 (2017)

166. G. Karunaratne, M. Schmuck, M. Le Gallo, G. Cherubini, L. Benini, A. Sebastian, A. Rahimi, *Robust High-Dimensional Memory-Augmented Neural Networks* (2020). arxiv:abs/2010.01939
167. S.A. Kauffman. *The Origins of Order: Self-organization and Selection in Evolution* (Oxford University Press, 1993)
168. S.A. Kauffman, *Reinventing the Sacred: A New View of Science, Reason, and Religion* (ISSR Library, Basic Books, 2008)
169. V. Kazemi, A. Elqursh, *Show, Ask, Attend, and Answer: A Strong Baseline For Visual Question Answering* (2017). arxiv:abs/1704.03162
170. K. Khetarpal, M. Klissarov, M. Chevalier-Boisvert, P.-L. Bacon, D. Precup, *Options of Interest: Temporal Abstraction with Interest Functions* (2020). arxiv:abs/2001.00271
171. K. Khetarpal, M. Riemer, I. Rish, D. Precup, Towards Continual Reinf. Learn.: A Rev. Perspect **2012**, 13490 (2020)
172. S. Kirkpatrick, C.D. Gelatt, M.P. Vecchi, Optimization by simulated annealing. Science **220**(4598), 671–680 (1983)
173. R. Kiros, R. Salakhutdinov, R. Zemel, Multimodal neural language models, in *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research, PMLR*, Bejing, China, 22–24 Jun 2014, ed. by E.P. Xing, T. Jebara, pp. 595–603 (2014)
174. R. Kline, Cybernetics, automata studies, and the dartmouth conference on artificial intelligence. IEEE Ann. Hist. Comput. **33**, 5–16, 04 (2011)
175. D.E. Knuth, P.B. Bendix, *Automation of Reasoning: 2: Classical Papers on Computational Logic 1967–1970, chapter Simple Word Problems in Universal Algebras* (Springer, Berlin, Heidelberg, 1983)
176. Z.A. Kocsis, J. Swan, Genetic Programming + Proof Search = Automatic Improvement. J. Autom. Reason. **60**(2), 157–176 (2018)
177. B.N. Kokinov, The DUAL cognitive architecture: a hybrid multi-agent approach, in *Proceedings of the Eleventh European Conference on Artificial Intelligence, Amsterdam, The Netherlands, 8–12 Aug. 1994*, ed. by A.G. Cohn (Wiley, Chichester, 1994), pp. 203–207
178. B.N. Kokinov, M. Grinberg, Simulating context effects in problem solving with AMBR, in *Modeling and Using Context, Proceedings of the Third International and Interdisciplinary Conference, CONTEXT, 2001, Dundee, UK, 27–30 July 2001*, ed. by V. Akman, P. Bouquet, R.H. Thomason, R.A. Young. Lecture Notes in Computer Science, vol. 2116 (Springer, 2001), pp. 221–234
179. B. Kovitz, J. Swan, Structural stigmergy: a speculative pattern language for metaheuristics, in *Proceedings of the Companion Publication of the 2014 Annual Conference on Genetic and Evolutionary Computation*, GECCO Comp '14, New York, NY, USA. (Association for Computing Machinery, 2014) pp. 1407–1410
180. J.R. Koza, *Genetic Programming: On the Programming of Computers by Means of Natural Selection* (MIT Press, Cambridge, MA, USA, 1992)
181. V. Krakovna, L. Orseau, M. Martic, S. Legg, Penalizing side effects using stepwise relative reachability, in *Proceedings of the Workshop on Artificial Intelligence Safety 2019 co-located with the 28th International Joint Conference on Artificial Intelligence, AISafety@IJCAI 2019, Macao, China, 11–12 Aug. 2019*, *CEUR Workshop Proceedings*, ed. by H. Espinoza, H. Yu, X. Huang, F. Lécué, C. Chen, J. Hernández-Orallo, S. Ó hÉigeartaigh, R. Mallah, vol. 2419 (2019). https://www.CEUR-WS.org
182. W. Kuhn, An image-schematic account of spatial categories, in *Spatial Information Theory, Berlin, Heidelberg*. ed. by S. Winter, M. Duckham, L. Kulik, B. Kuipers (Springer, Berlin, Heidelberg, 2007), pp. 152–168
183. K. Kurach, M. Andrychowicz, I. Sutskever, Neural random-access machines, in *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, 2–4 May 2016, Conference Track Proceedings*, ed. by Y. Bengio, Y. LeCun (2016)
184. A. Kurakin, I.J. Goodfellow, S. Bengio, Adversarial examples in the physical world, in *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, 24–26 Apr. 2017, Workshop Track Proceedings* (2017). https://www.OpenReview.net

185. S. Lack, P. Sobociński, Adhesive categories, in *Foundations of Software Science and Computation Structures, Berlin, Heidelberg*. ed. by I. Walukiewicz (Springer, Berlin, Heidelberg, 2004), pp. 273–288
186. I. Lakatos. *Criticism and the Growth of Knowledge: Proceedings of the International Colloquium in the Philosophy of Science, London, 1965*, chapter Falsification and the Methodology of Scientific Research Programmes, vol. 4 (Cambridge University Press, 1970), pp. 91–196
187. G. Lakoff, R.E. Núñez, *Where Mathematics Comes from: How the Embodied Mind Brings Mathematics Into Being* (Basic Books, 2000)
188. G. Lakoff, *Women, Fire and Dangerous Things: What Categories Reveal About the Mind* (University of Chicago Press, Chicago, 1987)
189. G. Lakoff, M. Johnson, *Metaphors we Live by* (University of Chicago Press, Chicago, 1980)
190. F. Lara-Dammer, D.R. Hofstadter, R.L. Goldstone, A computational model of scientific discovery in a very simple world, aiming at psychological realism. J. Exp. Theor. Artif. Intell. **31**(4), 637–658 (2019). https://doi.org/10.1080/0952813X.2019.1592234
191. K.Y.H. Largerspetz, Jakob von Uexküll and the origins of cybernetics. Semiotica **2001**(134), 643–651 (2001)
192. R.H. Lathrop, On the learnability of the uncomputable, in *Proceedings of the Thirteenth International Conference on International Conference on Machine Learning*, ICML'96, San Francisco, CA, USA. (Morgan Kaufmann Publishers Inc, 1996), pp. 302–309
193. M. Lázaro-Gredilla, D. Lin, J. Swaroop Guntupalli, D. George, *Beyond Imitation: Zero-Shot Task Transfer on Robots by Learning Concepts as Cognitive Programs* (2018). arxiv:abs/1812.02788
194. S. Legg, M. Hutter, Universal intelligence: a definition of machine intelligence. Minds Mach. **17**(4), 391–444 (2007)
195. J. Leike, D. Krueger, T. Everitt, M. Martic, V. Maini, S. Legg, *Scalable Agent Alignment via Reward Modeling: a Research Direction* (2018). arxiv:abs/1811.07871
196. T. Leinster, Basic category theory. *Cambridge Studies in Advanced Mathematics* (Cambridge University Press, 2014)
197. D. Lenat, M. Prakash, M. Shepherd, CYC: using common sense knowledge to overcome brittleness and knowledge Acquistion Bottlenecks. AI Mag. **6**(4), 65–85 (1986)
198. M. Leshno, V.Y. Lin, A. Pinkus, S. Schocken, Multilayer feedforward networks with a non-polynomial activation function can approximate any function. Neural Netw. **6**(6), 861–867 (1993)
199. S. Li, R. Wang, M. Tang, C. Zhang, Hierarchical reinforcement learning with advantage-based auxiliary rewards, in *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8–14 Dec. 2019*. ed. by H.M. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E.B. Fox, R. Garnett (Canada, Vancouver, BC, 2019), pp. 1407–1417
200. Z. Li, F. Zhou, F. Chen, H. Li, Meta-SGD: *Learning to Learn Quickly for Few Shot Learning* (2017). arxiv:abs/1707.09835
201. V. Lifschitz, What is answer set programming? in *Proceedings of the 23rd National Conference on Artificial Intelligence*, AAAI'08, vol. 3 (AAAI Press, 2008), pp. 1594–1597
202. J. Lighthill, *Artificial Intelligence: A General Survey*. Technical report, UK Science Research Council (1973)
203. H.W. Lin, M. Tegmark, D. Rolnick, Why does deep and cheap learning work so well? J. Stat. Phys. **168**(6), 1223–1247 (2017)
204. A. Liska, G. Kruszewski, M. Baroni, *Memorize or Generalize? Searching for a Compositional RNN in a Haystack* (2018). arxiv:abs/1802.06467
205. H. Lofting, *The Voyages of Doctor Dolittle* (Doctor Dolittle. Frederick A. Stokes, 1922)
206. R. Logan, N.F. Liu, M.E. Peters, M. Gardner, S. Singh, Barack's wife Hillary: using knowledge graphs for fact-aware language modeling, in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, Florence, Italy, July 2019 (Association for Computational Linguistics, 2019), pp. 5962–5971

207. J. Loula, M. Baroni, B. Lake, Rearranging the familiar: testing compositional generalization in recurrent networks, in *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, Brussels, Belgium, Nov. 2018 (Association for Computational Linguistics, 2018), pp. 108–114
208. S. Luke, *Essentials of Metaheuristics* (Lulu, 2nd edn., 2013). http://cs.gmu.edu/∼sean/book/metaheuristics/
209. S. MacLane, *Categories for the Working Mathematician*. Graduate Texts in Mathematics, vol. 5 (Springer, New York, 1971)
210. G. Marcus, E. Davis, *Rebooting AI: Building Artificial Intelligence We Can Trust* (Knopf Doubleday Publishing Group, 2019)
211. G. Marcus, *An Epidemic of AI Misinformation* (2019). https://thegradient.pub/an-epidemic-of-ai-misinformation/. Accessed 3rd Sep. 2021
212. G. Marcus, *GPT-2 and the Nature of Intelligence* (2019). https://thegradient.pub/gpt2-and-the-nature-of-intelligence/. Accessed 28th Jan. 2020
213. G. Marcus, The next decade in AI: four steps towards robust. Artif. Intell. **2002**, 06177 (2020)
214. F.G. Marcus, *The Algebraic Mind* (MIT Press, 2001)
215. Mars Climate Orbiter Mishap Investigation Board. Phase I Report. Technical report (1999)
216. D. Marsden, Categ. Theory Using String Diag. **1401**, 7220 (2014)
217. P. Martin-Löf, G. Sambin, *Bibliopolis Intuitionistic Type Theory* Studies in Proof Theory (Bibliopolis, Napoli, 1984)
218. H.R. Maturana, F.J. Varela, *Autopoiesis and Cognition: The Realization of the Living* (Boston Studies in the Philosophy and History of Science. Springer, Netherlands, 1991)
219. J. McCarthy, P.J. Hayes, *Some Philosophical Problems from the Standpoint of Artificial Intelligence* (Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1987), pp. 26–45
220. J. McCarthy, M.L. Minsky, N. Rochester, C.E. Shannon, A proposal for the dartmouth summer research project on artificial intelligence, 31 Aug. 1955. AI Mag. **27**(4), 12 (2006)
221. P. Medawar, *British Annual Gathering of MENSA* (Annual Lecture, 1962)
222. J. Merel, A. Ahuja, V. Pham, S. Tunyasuvunakool, S. Liu, D. Tirumala, N. Heess, G. Wayne, Hierarchical visuomotor control of humanoids, in *International Conference on Learning Representations* (2019)
223. T. Mikolov, I. Sutskever, K. Chen, G. Corrado, J. Dean, Distributed representations of words and phrases and their compositionality, in *Proceedings of the 26th International Conference on Neural Information Processing Systems*, NIPS'13, Red Hook, NY, USA, vol. 2. (Curran Associates Inc, 2013), pp. 3111–3119
224. R. Milner. *A Calculus of Communicating Systems*. Lecture Notes in Computer Science, vol. 92 (Springer, 1980)
225. P. Minervini, M. Bosnjak, T. Rocktäschel, S. Riedel, E. Grefenstette, Differentiable reasoning on large knowledge bases and natural language, in *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, 7–12 Feb. 2020* (AAAI Press, 2020), pp. 5182–5190
226. M. Minsky, S. Papert, *Perceptrons: An Introduction to Computational Geometry* (The MIT Press, 1969)
227. N. Mishra, M. Rohaninejad, X. Chen, P. Abbeel, A simple neural attentive meta-learner, in *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, 30 Apr.–3 May 2018, Conference Track Proceedings* (2018). https://www.OpenReview.net
228. M. Mitchell, *Can Analogy Unlock AI's Barrier of Meaning*? https://www.youtube.com/watch?v=QvLEmueHhqY. Accessed 28th Feb. 2020
229. M. Mitchell, Abst. Anal.-Making Artif. Intell. **2102**, 10717 (2021)
230. M. Mitchell, Why AI is Harder Than We Think **2104**, 12871 (2021)
231. M. Mittal, M. Gallieri, A. Quaglino, S. Sina Mirrazavi Salehian, J. Koutník, Neural lyapunov model predictive control (2020). arxiv:abs/2002.10451

232. P. Morerio, J. Cavazza, V. Murino, Minimal-Entropy correlation alignment for unsupervised deep domain adaptation, in *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, 30 Apr.–3 May 2018, Conference Track Proceedings* (2018). https://www.OpenReview.net
233. P.K. Mudrakarta, A. Taly, M. Sundararajan, K. Dhamdhere, Did the model understand the question? in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, 15–20 July 2018: Long Papers*, vol. 1 (Association for Computational Linguistics, 2018), pp. 1896–1906
234. R.M. Murphey, Instrumental conditioning of the fruit fly, Drosophila melanogaster. Anim. Behav. **15**(1), 153–161 (1967)
235. O. Nachum, H. Tang, X. Lu, S. Gu, H. Lee, S. Levine, *Why Does Hierarchy (Sometimes) Work So Well in Reinforcement Learning*? (2019). arxiv:abs/1909.10618
236. S. Narang, E. Undersander, G. Diamos, Block-sparse Recurr. Neural Netw. **1711**, 02782 (2017)
237. S. Nasiriany, V. Pong, S. Lin, S. Levine, Planning with goal-conditioned policies, in *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8–14 Dec. 2019, Vancouver, BC, Canada*. ed. by H.M. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E.B. Fox, R. Garnett (2019), pp. 14814–14825
238. A. Newell, *A Guide to the General Problem-solver Program GPS-2-2*. Memorandum (Rand Corporation) (Rand Corporation, 1963)
239. A. Nichol, J. Achiam, J. Schulman, *On First-Order Meta-Learning Algorithms* (2018). arxiv:abs/1803.02999
240. N.J. Nilsson, Shakey the Robot. Technical Report 323, AI Center, SRI International, 333 Ravenswood Ave., Menlo Park, CA 94025, Apr. 1984
241. E. Nivel, K.R. Thórisson, B. Steunebrink, J. Schmidhuber, Anytime bounded rationality, in *Artificial General Intelligence* (Springer International Publishing, 2015), pp. 121–130
242. E. Nivel, K.R. Thórisson, B.R. Steunebrink, H. Dindo, G. Pezzulo, M. Rodriguez, C. Hernandez, D. Ognibene, J. Schmidhuber, R. Sanz, H.P. Helgason, A. Chella, G.K. Jonsson, *Bounded Recursive Self-Improvement* (2013). arxiv:abs/1312.6764
243. E. Nivel, K.R. Thórisson, B.R. Steunebrink, H. Dindo, G. Pezzulo, M. Rodríguez, C. Hernández, D. Ognibene, J. Schmidhuber, R. Sanz, H.P. Helgason, A. Chella, Bounded seed-AGI, in *Artificial General Intelligence*, Cham, ed. by B. Goertzel, L. Orseau, J Snaider (Springer International Publishing, 2014), pp. 85–96
244. A. Nowak, J. Bruna, *Divide and Conquer with Neural Networks* (2016). arxiv:abs/1611.02401
245. J.N. Oliveira, *Program Design by Calculation* (2021). https://www4.di.uminho.pt/~jno/ps/pdbc.pdf. draft of textbook in preparation
246. J.H. Oort, The force exerted by the stellar system in the direction perpendicular to the galactic plane and some related problems. Bull. Astronom. Inst. Netherlands **6**, 249 (1932)
247. O. AI, I. Akkaya, M. Andrychowicz, M. Chociej, M. Litwin, B. McGrew, A. Petron, A. Paino, M. Plappert, G. Powell, R. Ribas, J. Schneider, N. Tezak, J. Tworek, P. Welinder, L. Weng, Q. Yuan, W. Zaremba, L. Zhang, *Solving Rubik's Cube with a Robot Hand* (2019). arxiv:abs/1910.07113
248. B. Paaßen, A. Schulz, T.C. Stewart, B. Hammer, *Reservoir Memory Machines as Neural Computers* (2020). arxiv:abs/2009.06342
249. N. Papernot, P.D. McDaniel, A. Swami, R.E. Harang, Crafting adversarial input sequences for recurrent neural networks, in *2016 IEEE Military Communications Conference, MILCOM 2016, Baltimore, MD, USA, 1–3 Nov. 2016*, ed. by J. Brand, M.C. Valenti, A. Akinpelu, B.T. Doshi, B.L. Gorsic (IEEE, 2016), pp. 49–54
250. A. Papoulis, S. Unnikrishna Pillai, *Probability, Random Variables, and Stochastic Processes* (McGraw Hill, Boston, 2002)
251. D. Parton, 9 to 5. https://www.youtube.com/watch?v=UbxUSsFXYo4. Accessed 17th Dec. 2020

252. H. Pattee, Evolving self-reference: matter, symbols, and semantic closure. Commun. Cogn.—Artif. Intell. **12**, 9–27 (1995)
253. H. Pattee, *Laws, Language and Life: Howard Pattee's classic papers on the physics of symbols with contemporary commentary, chapter Cell Psychology: An Evolutionary Approach to the Symbol-Matter Problem* (Springer, Netherlands, Dordrecht, 2012), pp. 165–179
254. J. Pearl, *Causality: Models, Reasoning and Inference*, 2nd edn. (Cambridge University Press, New York, NY, USA, 2009)
255. X.B. Peng, M. Andrychowicz, W. Zaremba, P. Abbeel, Sim-to-Real transfer of robotic control with dynamics randomization, in *2018 IEEE International Conference on Robotics and Automation, ICRA 2018, Brisbane, Australia, 21–25 May 2018*, pp. 1–8
256. M.E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, L. Zettlemoyer, Deep contextualized word representations, in *Proceedings of NAACL* (2018)
257. G. Pezzulo, G. Calvi, Dynamic computation and context effects in the hybrid architecture AKIRA, in *Proceedings of the 5th International Conference on Modeling and Using Context*, CONTEXT'05, Berlin, Heidelberg (Springer, 2005), pp. 368–381
258. S. Phillips, A general (category theory) principle for general intelligence: duality (adjointness), in *Artificial General Intelligence*, Cham, ed. by T. Everitt, B. Goertzel, A. Potapov (Springer International Publishing, 2017), pp. 57–66
259. S. Phillips, W.H. Wilson, Categorial compositionality III: F-(co)algebras and the systematicity of recursive capacities in human cognition. PloS One **7**(4), e35028 (2012)
260. D. Philps, A.S. d'Avila Garcez, T. Weyde, *Making Good on LSTMs Unfulfilled Promise* (2019). arxiv:abs/1911.04489
261. J. Piaget, *Biologie Et Connaissance: essai sur les relations entre les régulations organiques et les processus cognitifs* (Edinburgh University Press, 1971)
262. J. Piaget, M. Piercy, D.E. Berlyne, *The Psychology of Intelligence*. International library of psychology (Routledge & Paul, 1950)
263. B.C. Pierce, *Advanced Topics in Types and Programming Languages* (MIT Press, 2005)
264. C.B. Pierce, *Basic Category Theory for Computer Scientists* (MIT Press, 1991)
265. C.B. Pierce, *Types and Programming Languages* (The MIT Press, 2002)
266. G.D. Plotkin, A note on inductive generalization. Mach. Intell. **5**, 153–163 (1970)
267. D. Plump, Hypergraph rewriting: critical pairs and undecidability of confluence. Term Graph Rewrit: Theory Pract. **15**, 201–213 (1993)
268. K.R. Popper, *Conjectures and Refutations: The Growth of Scientific Knowledge* (Routledge Classics, Routledge, 1963)
269. T. William, *Powers, Behavior: The Control of Perception* (Benchmark Publications, Incorporated, 1973)
270. S. Pozzoli, M. Gallieri, R. Scattolini, *Tustin Neural Networks: A Class of Recurrent Nets for Adaptive MPC of Mechanical Systems* (2019). arxiv:abs/1911.01310
271. H. Prade, G. Richard, *Computational Approaches to Analogical Reasoning: Current Trends* (Springer, 2014)
272. E. Price, W. Zaremba, I. Sutskever, *Extensions and Limitations of the Neural GPU* (2016). arxiv:abs/1611.00736
273. A. Puigdoménech, B. Piot, S. Kapturowski, P. Sprechmann, A. Vitvitskyi, D. Guo, C. Blundell, Agent57: outperforming the human Atari benchmark (2020). https://deepmind.com/blog/article/Agent57-Outperforming-the-human-Atari-benchmark. Accessed 3rd Sep. 2021
274. qntm. https://twitter.com/badedgecases/status/1421855808699125762?s=20
275. A. Quaglino, M. Gallieri, J. Masci, J. Koutník, SNODE: spectral discretization of neural ODEs for system identification, in *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, 26–30 Apr. 2020*. https://www.OpenReview.net
276. A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, *Language Models are Unsupervised Multitask Learners* (2019). https://d4mucfpksywv.cloudfront.net/better-language-models/language-models.pdf
277. R. Raileanu, M. Goldstein, A. Szlam, R. Fergus, Fast Adapt. via Policy-Dyn. Value Fun. **2007**, 02879 (2020)

278. N. Ramanan, S. Natarajan, Causal learning from predictive modeling for observational data. Front. Big Data **3**, 34 (2020)
279. C.H. Rankin, Invertebrate learning: what can't a worm learn? Curr. Biol. **14**(15), R617–R618 (2004)
280. A. Rasheed, O. San, T. Kvamsdal, Digital twin: values, challenges and enablers from a modeling perspective. IEEE Access **8**, 21980–22012 (2020)
281. S. Rasti, N. Marandi, A. Abdoli, M. Delavari, S. Gholam Abbas, Mousavi, Serological and molecular detection of Toxoplasma gondii in sheep and goats in Kashan, Central Iran. J. Food Saf. **38**(2), e12425 (2018)
282. A. Ray, J. Achiam, D. Amodei, *Benchmarking Safe Exploration in Deep Reinforcement Learning* (2019). https://cdn.openai.com/safexp-short.pdf
283. S.E. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, H. Lee, Generative adversarial text to image synthesis, in *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, 19–24 June 2016*, *JMLR Workshop and Conference Proceedings*, ed. by M.-F. Balcan, K.Q. Weinberger, vol. 48 (2016), pp. 1060–1069. https://www.JMLR.org
284. S.E. Reed, N. de Freitas, Neural programmer-interpreters, in *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, 2–4 May 2016, Conference Track Proceedings*, ed. by Y. Bengio, Y. LeCun (2016)
285. T. Regier, *The Human Semantic Potential: Spatial Language and Constrained Connectionism* (MIT Press, Cambridge, MA, 1996)
286. M. Ren, E. Triantafillou, S. Ravi, J. Snell, K. Swersky, J.B. Tenenbaum, H. Larochelle, R.S. Zemel, Meta-Learning for semi-supervised few-shot classification, in *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, 30 Apr.–3 May 2018, Conference Track Proceedings*. https://www.OpenReview.net
287. J.C. Reynolds, Definitional interpreters for higher-order programming languages, in *Proceedings of the ACM Annual Conference*, ACM '72, New York, NY, USA, vol. 2 (Association for Computing Machinery, 1972), pp. 717–740
288. E. Rich, *Artificial Intelligence*, 2nd edn. (McGraw-Hill Higher Education, 1990)
289. S.B. Richard, O. de Moor, *Algebra of Programming* (Prentice Hall International Series in Computer Science, Prentice Hall, 1997)
290. E. Riehl, *Category Theory in Context Dover Modern Math Originals* (Dover Publications, Aurora, 2017)
291. M. Riley, *Categories of Optics* (2018). arXiv:1809.00738
292. M. Ring, L. Orseau, Delusion, survival, and intelligent agents, in *Artificial General Intelligence, Berlin, Heidelberg*. ed. by J. Schmidhuber, K.R. Thórisson, M. Looks (Springer, Berlin, Heidelberg, 2011), pp. 11–20
293. S. Ritter, R. Faulkner, L. Sartran, A. Santoro, M. Botvinick, D. Raposo, Rapid Task-Solv. Novel Environ. **2006**, 03662 (2020)
294. T. Rocktäschel, S. Riedel, End-to-end differentiable proving, in *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, ed. by I. Guyon, U. von Luxburg, S. Bengio, H.M. Wallach, R. Fergus, S.V.N. Vishwanathan, R. Garnett, pp. 3788–3800 (2017)
295. Towards Functionally Elegant Grand Unification, P. Rosenbloom, A. Demski, Volkan Ustun, The sigma cognitive architecture and system. J. Artif. Gen Intell. **7**, 1–103 (2016)
296. P.S. Rosenbloom, Deconstructing reinforcement learning in sigma, in *Artificial General Intelligence, Berlin, Heidelberg*. ed. by J. Bach, B. Goertzel, M. Iklé (Springer, Heidelberg, Berlin, 2012), pp. 262–271
297. P.S. Rosenbloom, Deconstructing episodic memory and learning in sigma, in *Proceedings of the 36th Annual Meeting of the Cognitive Science Society, CogSci 2014, Quebec City, Canada, 23–26 July 2014*, ed. by P. Bello, M. Guarini, M. McShane, B. Scassellati (2014). https://www.cognitivesciencesociety.org
298. P.S. Rosenbloom, Lessons from mapping sigma onto the standard model of the mind: self-monitoring, memory/learning, and symbols, in *2017 AAAI Fall Symposia, Arlington, Virginia, USA, 9–11 Nov. 2017* (AAAI Press, 2017), pp. 449–454

299. A. Rosenblueth, N. Wiener, J. Bigelow, Behavior, purpose and teleology. Philosop. Sci. **10**(1), 18–24 (1943)
300. S. Russell, P. Norvig, *Artificial Intelligence: A Modern Approach*, 3rd edn. (Prentice Hall Press, USA, 2009)
301. C. Rutz, G.R. Hunt, J.J.H. St Clair, Corvid technologies: how do new caledonian crows get their tool designs? Curr. Biol. **28**(18), R1109–R1111 (2018)
302. A.M. Saxe, Y. Bansal, J. Dapello, M. Advani, A. Kolchinsky, B.D. Tracey, D.D. Cox, On the information bottleneck theory of deep learning, in *ICLR (Poster)* (2018). https://www.OpenReview.net
303. D. Saxton, E. Grefenstette, F. Hill, P. Kohli, Analysing mathematical reasoning abilities of neural models, in *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, 6–9 May 2019* (2019). https://www.OpenReview.net
304. R.C. Schank, R.P. Abelson, *Scripts, Plans, Goals, and Understanding: An Inquiry Into Human Knowledge Structures* (Lawrence Erlbaum Associates, The Artificial Intelligence Series, 1977)
305. T. Schaul, D. Horgan, K. Gregor, D. Silver, Universal value function approximators, in *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, PMLR, Lille, France, 07–09 Jul 2015, ed. by F. Bach, D. Blei (2015), pp. 1312–1320
306. J. Schmidhuber, Gödel machines: self-referential universal problem solvers making provably optimal self-improvements. Technical Report IDSIA-19-03. IDSIA, Manno-Lugano, Switzerland (2003). arXiv:cs.LO/0309048. Accessed 2006
307. J. Schmidhuber, *POWERPLAY: Training an Increasingly General Problem Solver by Continually Searching for the Simplest Still Unsolvable Problem* (2011). arxiv:abs/1112.5309
308. M. Schmidt, U. Krumnack, H. Gust, K.-U. Kühnberger, *Computational Approaches to Analogical Reasoning: Current Trends, chapter Heuristic-Driven Theory Projection: An Overview* (Springer, Berlin Heidelberg, 2014), pp. 163–194
309. B. Schölkopf, F. Locatello, S. Bauer, N.R. Ke, N. Kalchbrenner, A. Goyal, Y. Bengio, Toward causal representation learning. Proc. IEEE—Adv. Mach. Learn. Deep Neural Netw. **109**(5), 612–634 (2021)
310. B. Schölkopf, *Causality for Machine Learning* (2019). arxiv:abs/1911.10500
311. M. Shanahan, K. Nikiforou, A. Creswell, C. Kaplanis, D.GT. Barrett, M. Garnelo, An explicitly relational neural network architecture, in *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13–18 July 2020, Virtual Event*, *Proceedings of Machine Learning Research*. PMLR, (2020), pp. 8593–8603
312. A. Sharma, S. Gu, Sergey L. Vikash Kumar, K. Hausman, *Dynamics-Aware Unsupervised Discovery of Skills* (2019). arxiv:abs/1907.01657
313. M. Shoeybi, M. Patwary, R. Puri, P. LeGresley, J. Casper, B. Catanzaro, *Megatron-LM: Training Multi-Billion Parameter Language Models Using Model Parallelism* (2019). arxiv:abs/1909.08053
314. D. Silver, A. Huang, C.J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, D. Hassabis, Mastering the game of Go with deep neural networks and tree search. Nature **529**, 484–503 (2016)
315. D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel, T.P. Lillicrap, K. Simonyan, D. Hassabis, *Mastering Chess and Shogi by Self-Play with a General Reinforcement Learning Algorithm* (2017). arxiv:abs/1712.01815
316. A. Sloman, The irrelevance of turing machines to AI, in *Computationalism: New Directions*, ed. by M. Scheutz (MIT Press, 2002)
317. A. Sloman, Kantian philosophy of mathematics and young robots, in *Intelligent Computer Mathematics, Berlin, Heidelberg*. ed. by S. Autexier, J. Campbell, J. Rubio, V. Sorge, M. Suzuki, F. Wiedijk (Springer, Berlin, Heidelberg, 2008), pp. 558–573
318. T.S.C. Smithe, *Bayesian Updates Compose Optically* (2020). arXiv:2006.01631

319. R.M. Smullyan, *Godel's Incompleteness Theorems* Oxford Logic Guides (Oxford University Press, 1992)
320. M. Sotoudeh, A.V. Thakur, Anal.-Making Core Primit. Softw. Eng Toolbox **2009**, 06592 (2020)
321. D.I. Spivak, *Functorial Dynamics and Interaction: A Computational Design Environment*. www.dspivak.net/grants/AFOSR2020-Topos-ContextDependence.pdf
322. A. Srinivas, A. Jabri, P. Abbeel, S. Levine, C. Finn, Univ. Plann. Netw. **1804**, 00645 (2018)
323. R.K. Srivastava, B.R. Steunebrink, J. Schmidhuber, First experiments with Power Play. Neural Netw. 41:130–136 (2013). Special Issue on Autonomous Learning
324. J.E.R. Staddon, D.T. Cerutti, Operant conditioning. Ann. Rev. Psychol. **54**(1), 115–144 (2003)
325. B.R. Steunebrink, J. Koutník, K.R. Thórisson, E. Nivel, J. Schmidhuber, Resource-Bounded machines are motivated to be effective, efficient, and curious, in *Artificial General Intelligence—6th International Conference, AGI 2013, Beijing, China, 31 July–3 Aug. 2013 Proceedings*, ed. by K.-U. Kühnberger, S. Rudolph, P. Wang. Lecture Notes in Computer Science, vol. 7999 (Springer, 2013), pp. 119–129
326. I. Stewart, J. Cohen. *The Collapse of Chaos: Discovering Simplicity in a Complex World* (Penguin Books Limited, 2000)
327. R. Sun, The motivational and metacognitive control in CLARION. *Modeling Integrated Cognitive Systems*, Apr. 2012
328. R. Sun, T. Peterson, C. Sessions, Beyond simple rule extraction: acquiring planning knowledge from neural networks, in *Neural Nets WIRN Vietri-01* (Springer, 2002), pp. 288–300
329. G.J. Sussman, J. Wisdom, *Structure and Interpretation of Classical Mechanics* (MIT Press, Cambridge, MA, USA, 2001)
330. I. Sutskever, O. Vinyals, Q.V. Le, Sequence to sequence learning with neural networks, in *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, 8–13 Dec. 2014, Montreal, Quebec, Canada*, ed. by Z. Ghahramani, M. Welling, C. Cortes, N.D. Lawrence, K.Q. Weinberger (2014), pp. 3104–3112
331. R.S. Sutton, A.G. Barto, *Reinforcement Learning: An Introduction* (The MIT Press, 2018)
332. J. Swan, C.G. Johnson, E.C. Brady, *Subtype polymorphism à la carte via machine learning on dependent types, in Companion Proceedings for the ISSTA/ECOOP 2018 Workshops, ISSTA'18* (NY, USA (Association for Computing Machinery, New York, 2018), pp. 14–16
333. J. Swan, K. Krawiec, Z.A. Kocsis, Stochastic program synthesis via recursion schemes, in *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, GECCO'19, New York, NY, USA (Association for Computing Machinery, 2019), pp. 35–36
334. J. Swan, K. Krawiec, Z.A. Kocsis, Stochastic synthesis of recursive functions made easy with bananas, lenses, envelopes and barbed wire. Gen. Programm. Evol. Mach. **20**(3), 327–350 (2019)
335. C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I.J. Goodfellow, R. Fergus, Intriguing properties of neural networks, in *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, 14–16 Apr. 2014, Conference Track Proceedings*, ed. by Y. Bengio, Y. LeCun (2014)
336. R. Thom, *Structural Stability and Morphogenesis-An Outline of a General Theory of Models* (Benjamin, W.A., 1972)
337. D.A.W. Thompson, *On Growth and Form* Cambridge paperbacks (Cambridge University Press, 1917)
338. K.R. Thórisson, D. Kremelberg, B.R. Steunebrink, E. Nivel, About understanding, in *Artificial General Intelligence—9th International Conference, AGI 2016, New York, NY, USA, 16–19 Jul. 2016, Proceedings* (2016), pp. 106–117
339. S. Thrun, *Flying Cars, Autonomous Vehicles, and Education*. https://www.youtube.com/watch?v=ZPPAOakITeQ. Accessed 28th Jan. 2020
340. S. Thrun, T.M. Mitchell, Lifelong robot learning, in *The Biology and Technology of Intelligent Autonomous Agents, Berlin, Heidelberg*. ed. by L. Steels (Springer, Berlin, Heidelberg, 1995), pp. 165–196

341. S. Thys, W. Van Ranst, T. Goedemé, Fooling automated surveillance cameras: adversarial patches to attack person detection, in *IEEE Conference on Computer Vision and Pattern Recognition Workshops, CVPR Workshops 2019, Long Beach, CA, USA, 16–20 June 2019* (Computer Vision Foundation/IEEE, 2019), p. 0
342. H.R. Tiwary, On the hardness of computing intersection, union and Minkowski sum of polytopes. Disc. Comput. Geom. **40**(3), 469–479 (2008)
343. J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, P. Abbeel, Domain randomization for transferring deep neural networks from simulation to the real world, in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2017, Vancouver, BC, Canada, 24–28 Sep. 2017* (IEEE, 2017), pp. 23–30
344. J. Togelius, *Empiricism and the Limits of Gradient Descent*, May 2018. http://togelius.blogspot.com/2018/05/empiricism-and-limits-of-gradient.html. Accessed 28th Jan. 2020
345. F. Tramèr, A. Kurakin, N. Papernot, I.J. Goodfellow, D. Boneh, P.D. McDaniel, Ensemble adversarial training: attacks and defenses, in *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, 30 Apr.–3 May 2018, Conference Track Proceedings* (2018). https://www.OpenReview.net
346. M. Turchetta, F. Berkenkamp, A. Krause, Safe exploration for interactive machine learning, in *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8–14 Dec. 2019, Vancouver, BC, Canada*, ed. by H.M. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E.B. Fox, R. Garnettpages pp. 2887–2897 (2019)
347. H. Turner, Representing actions in logic programs and default theories: a situation calculus approach. J. Log. Program. **31**(1–3), 245–298 (1997)
348. E. Tzeng, J. Hoffman, K. Saenko, T. Darrell, Adversarial discriminative domain adaptation, in *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, 21–26 July 2017* (IEEE Computer Society, 2017), pp. 2962–2971
349. W. van Melle, MYCIN: a knowledge-based consultation program for infectious disease diagnosis. Int. J. Man-Mach. Stud. **10**(3), 313–322 (1978)
350. K. Van Moffaert, M. Drugan, A. Nowe, Scalarized multi-objective reinforcement learning: novel design techniques in *IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning, ADPRL*, May 2013
351. O. Vinyals, I. Babuschkin, W.M. Czarnecki, M. Mathieu, A. Dudzik, J. Chung, D.H. Choi, R. Powell, T. Ewalds, P. Georgiev, O. Junhyuk, D. Horgan, M. Kroiss, I. Danihelka, A. Huang, L. Sifre, T. Cai, J.P. Agapiou, M. Jaderberg, A.S. Vezhnevets, R. Leblond, T. Pohlen, V. Dalibard, D. Budden, Y. Sulsky, J. Molloy, T.L. Paine, C. Gulcehre, Z. Wang, T. Pfaff, W. Yuhuai, R. Ring, D. Yogatama, D. Wünsch, K. McKinney, O. Smith, T. Schaul, T. Lillicrap, K. Kavukcuoglu, D. Hassabis, C. Apps, D. Silver, Grandmaster level in StarCraft II using multi-agent reinforcement learning. Nature **575**(7782), 350–354 (2019)
352. R. Volpi, H. Namkoong, O. Sener, J. Duchi, V. Murino, S. Savarese, Generalizing to unseen domains via adversarial data augmentation, in *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, Red Hook, NY, USA (Curran Associates Inc, 2018), pp. 5339–5349
353. J. von Neumann, O. Morgenstern. *Theory of Games and Economic Behavior* (Princeton University Press, 1947)
354. A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, S.R. Bowman, GLUE: a multi-task benchmark and analysis platform for natural language understanding, in *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, 6–9 May 2019* (2019). https://www.OpenReview.net
355. P. Wang, Embodiment: does a laptop have a body? In *Proceedings of the 2nd Conference on Artificial General Intelligence* (2009) (Atlantis Press, 2009/06)
356. P. Wang, *Non-Axiomatic Logic: A Model of Intelligent Reasoning* (World Scientific Publishing Co., Inc, USA, 2013)
357. P. Wang, On defining artificial intelligence. J. Artif. Gen. Intell. **10**(2), 1–37 (2019)

358. P. Wang, P. Hammer, H. Wang, An architecture for real-time reasoning and learning, in *Artificial General Intelligence*, Cham, ed. by B. Goertzel, A.I. Panov, A. Potapov, R. Yampolskiy (Springer International Publishing, 2020), pp. 347–356
359. M. Weber, M. Yurochkin, S. Botros, V. Markov, Black Loans Matter: Distribut. Robust Fairness Fight. Subgr. Discrim. **2012**, 01193 (2020)
360. S. Weller, U. Schmid, Solving proportional analogies by E-Generalization, in *KI 2006: Advances in Artificial Intelligence, 29th Annual German Conference on AI, KI 2006, Bremen, Germany, 14–17 June 2006, Proceedings*, pp. 64–75 (2006)
361. M. Werning, W. Hinzen, E. Machery, (eds.), *The Oxford Handbook of Compositionality* (Oxford University Press, 2012)
362. G. Wheeler, Bounded rationality, in *The Stanford Encyclopedia of Philosophy*, ed. by E.N. Zalta (Metaphysics Research Lab, Stanford University, fall 2020 edn., 2020)
363. E.P. Wigner, The unreasonable effectiveness of mathematics in the natural sciences. *Communications on Pure and Applied Mathematics* (Richard Courant Lecture in Mathematical Sciences Delivered at New York University, 11 May 1959)
364. A. Daniel, Understanding as representation manipulability. Synthese **190**(6), 997–1016 (2013)
365. J.C. Willems, The behavioral approach to open and interconnected systems. IEEE Control Syst. Mag. **27**(6), 46–99 (2007)
366. T. Winograd, *Understanding Natural Language* (Academic Press Inc, USA, 1972)
367. L. Wittgenstein. *Tractatus Logico-Philosophicus* (Dover Publications, 1922)
368. J. Woehr, An interview with Donald Knuth. Dr Dobb's J.-Softw. Tools Prof. Programm. **21**(4), 16–23 (1996)
369. J. Woodward, J. Swan, S. Martin, The 'composite' design pattern in metaheuristics. *GECCO 2014—Companion Publication of the 2014 Genetic and Evolutionary Computation Conference*, July 2014
370. A.W. Yu, D. Dohan, M.-T. Luong, R. Zhao, K. Chen, M. Norouzi, Q.V. Le, QANet: combining local convolution with global self-attention for reading comprehension, in *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, 30 Apr.–3 May 2018, Conference Track Proceedings* (2018). https://www.OpenReview.net
371. Yu. Tianhe, G. Shevchuk, D. Sadigh, C. Finn, Unsupervised Visuomotor Control Through Distrib. Plann. Netw. **1902**, 05542 (2019)
372. A.L. Zadeh, Fuzzy sets. Inf. Control **8**, 338–353 (1965)
373. W. Zaremba, I. Sutskever, *Learning to Execute* (2014). arxiv:abs/1410.4615
374. C. Zhang, S. Bengio, M. Hardt, B. Recht, O. Vinyals, Understanding deep learning requires rethinking generalization, in *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, 24–26 Apr. 2017, Conference Track Proceedings* (2017). https://www.OpenReview.net
375. M. Zhu, S. Gupta, *To Prune, or not to Prune: Exploring the Efficacy of Pruning for Model Compression* (2017). 1710.01878
376. P. Zucker, *Reverse Mode Differentiation is Kind of Like a Lens II* (2018). http://www.philipzucker.com/reverse-mode-differentiation-is-kind-of-like-a-lens-ii/. Accessed 11th Dec. 2020