"Go to the Children": Rethinking Intelligent Agent Design and Programming in a Developmental Learning Perspective

Blue Sky Ideas Track

Alessandro Ricci
Alma Mater Studiorum - Università di Bologna
Cesena Campus, Italy
a.ricci@unibo.it

ABSTRACT

In this paper we propose to rethink the development of intelligent agents based on cognitive architectures as a developmental learning process, inspired by theories of learning in children and cognitive development. The idea is targeted to explore architectures, methods and tools to systematically develop intelligent agents capable of integrating both practical knowledge and skills designed by developers and knowledge and skills acquired by interacting in properly designed learning environments.

KEYWORDS

Agent Programming, Developmental Learning, Cognitive Architectures, Causal Learning

ACM Reference Format:
Alessandro Ricci. 2022. "Go to the Children": Rethinking Intelligent Agent Design and Programming in a Developmental Learning Perspective: Blue Sky Ideas Track. In *Proc. of the 21st International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2022), Online, May 9–13, 2022*, IFAAMAS, 6 pages.

1 INTRODUCTION

In the context of agent programming and intelligent agent design based on cognitive architectures, *learning* has been explored so far mainly as an effective AI technique to tackle specific problems, improving agent and MAS (Multi-Agent Systems) adaptability and performance at runtime [2, 57]. In this paper we consider a broader role of learning, impacting on the full spectrum of agent programming and engineering—from design to runtime.

In particular, we introduce a perspective in which agent development is conceived as a *developmental learning* process of the agent itself. We will refer to this approach as DevL (Developmental Learning). In that view, a software agent "is born" with some core domain-independent learning capabilities at the architectural level and is grown up in a proper *learning environment* to acquire the skills to be able then to achieve its designed objectives once it will be deployed.

This idea relies on – on the one hand – existing literature about how learning has been injected in agent architectures such as BDI (Belief-Desire-Intention) [44] as well as in cognitive architectures originated from cognitive science (e.g. SOAR [27, 29]). On the other hand, the idea draws inspiration from human science, in particular the literature investigating the role of learning in human development, children in particular, and the impact of learning capabilities on human autonomy and practical reasoning.

From a software engineering point of view, this idea is meant to be a conceptual baseline to explore general-purpose architectures and methodologies – eventually extending existing ones – that are effective to seamlessly and systematically integrate both parts engineered by developers (e.g. plans in the BDI case) and parts learnt by the agent itself, eventually adopting different learning strategies. This is meant to be a contribution useful to devise new perspectives about agent programming in the cognitive era [10], in which learning is explored as a core capability, shedding a new light on programming, languages and software development tools.

In the remainder of the paper,fi rst we provide a background about related works in literature (Section 2), then we describe the key points of the DevL idea (Section 3) and how agent development can be rethought accordingly (Section 4). Finally, we provide an overview of some main research directions useful to develop the idea (Section 5).

Proc. of the 21st International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2022), P. Faliszewski, V. Mascardi, C. Pelachaud, M.E. Taylor (eds.), May 9–13, 2022, Online. © 2022 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org). All rights reserved.

2 BACKGROUND: THE ROLE OF LEARNING IN AGENT DEVELOPMENT BASED ON COGNITIVE ARCHITECTURES

Learning is a main topic in agent and MAS research [2]. The idea proposed in this paper is related in particular to research works exploring the integration of learning capabilities to support the development of intelligent agents, based on cognitive architectures. This integration is already occurring in specific application domains. A main one is the development of self-driving vehicles, which involves a large amount of learning in carefully engineered simulated environments [15]. Another main one is in the development of human-like virtual agents and avatars by means of an "experiential learning" approach (e.g. [32]), in which the agent learns through direct experience and playing with people and the world around.

In the context of agent programming and MAS development, various research works have explored the extension of the BDI (Belief-Desire-Intention) architecture with learning [1, 20, 21, 37, 47, 48, 52, 54], mainly to improve plan/action selection capability at runtime.

A deeper integration of learning at the architectural level can be found in *cognitive architectures* [28, 30, 36], as developed in cognitive science but also applied to the design of autonomous agents featuring general artificial intelligence capabilities [31, 59]. Main examples are SOAR [27], ACT-R [4], SIGMA [45], LIDA [49] (see

[24] for a recent survey). In agents based on cognitive architectures, programs and data are ultimately intended to be acquired automatically from experience — that is, learned – rather than programmed [28]. Cognitive architectures thus induce languages that are geared toward yielding learnable intelligent behavior, in the form of knowledge and skills, and integrate general learning mechanisms at the architectural level. A main example is *chunking*, adopted e.g. in SOAR since its origin [26]. Chunking is focused in particular to improve performances by acquiring dynamically rules from goal-based experience. SOAR has been further extended in the last two decades to include also a native support for other forms of learning [25, 27], namely semantic learning (interacting with semantic long term memory), episodic learning (for episodic long-term memory) and reinforcement learning (acting on procedural memory, like chunking).

The idea proposed in this paper has been strongly inspired by the integration perspective of learning at the architectural level as found in cognitive architectures such as SOAR. Two key values are: *(i)* the aim of integrating learning with the other cognitive capabilities – reasoning and planning – as a whole, that was a landmark of Newell's approach to understand cognition as well as to design cognitive agents [36]; *(ii)* the aim of understanding learning at the *knowledge level* [12, 35], which accounts for embracing a proper level of abstractions in modelling and designing autonomous systems, useful to support the explainability of their behaviour as well as their robust engineering.

Then, the main aspect interesting for this paper that apparently has not fully explored so far in the state of the art is about exploring and exploiting learning as fundamental feature to support the *development process* of intelligent agents.

3 DEVELOPMENTAL LEARNING FOR INTELLIGENT AGENTS

The central idea proposed in this paper is that the development of an intelligent agent could be framed as a process that has the objective of seamlessly integrating practical knowledge provided by designers/programmers with knowledge and skills acquired i.e. *learnt* by the agent, by interacting or *growing up* in a *learning environment*, properly designed and configured for that purpose.

In that view, current approaches used to develop e.g. classic reinforcement learning (RL) [51] and Deep RL agents [5] could be thought as one extreme case, in which the behaviour of the agent is fully learnt, following the *reward is enough* hypothesis [46]. Instead, current approaches used to develop e.g. BDI agents could be thought as the other extreme, in which the behaviour of the agent is fully specified by an agent program, including plans embedding the practical knowledge to performs the tasks.

Like in the case of cognitive architectures, the idea calls for framing a support at the architectural level, in order to work in spite of any specific application domains or specific tasks that intelligent agents are meant to accomplish. More generally, the idea calls for a proper *conceptual framework*, in order to understand thefirst-class aspects related to learning impacting on the design of the agent architecture, as well as of the languages and tools used to support the development process.

To this purpose, the main source of inspiration and insights adopted in this paper comes from theories and research studies about learning in children, as developed in psychology. In particular, we consider research studies about *developmental learning*, investigating learning taking place as a normal part of cognitive development [17, 18, 55]. Such research studies appear strongly interesting here because children are an effective and impressive case of agents capable of somehow combining both a bottom-up and top-down approaches in learning, that come to be integrated in their continuous cognitive development process. In particular, according to the "Theory Theory" [16, 17], a reference cognitive development theory in psychology, children grow up by constructing intuitive theories of the world and alter and revise those theories as the result of new evidence, like theory change in science [18]. This view moves from the original constructivist theory by Piaget [56] towards a modern *rational constructivism*, featuring formalisations and computational descriptions based on probabilistic models and Bayesian learning [18].

Analogously to the children case, it appears interesting here to frame the development process of an intelligent agent – based on proper cognitive architecture – as a cognitive development process in which an agent incrementally constructs theories about the task environment for which the agent is being developed and revises them through a learning process in which the agent is situated and interacts with a proper learning environment. In this paper we refer to this approach as DevL (Developmental Learning) for intelligent agents.

3.1 The Role of Causal Learning and Reasoning

A main research question in this view is which core mechanisms and structures should be embedded directly into the agent architecture in order to effectively support the developmental learning process, in spite of the specific application domain, eventually supporting different learning methods during the development stage. That is: what kind of primitive learning system the agent could be equipped with, general and expressive enough to effectively support "learning to learn", as well as practical reasoning.

To that purpose, the rational constructivism and the "Theory Theory" provide interesting insights. In the case of children, *causal learning* plays a key role. Children – and adults as well – are remarkable causal learners [17]. Young children have both sophisticated domain-specific and domain-general causal reasoning abilities, and these abilities appear to be fundamental to enable their cognitive development. Children's abilities to integrate their existing *substantive* knowledge with formal mechanisms for causal learning have led researchers to propose that children are learning abstract causal models, represented by *causal maps* [18].

Causal maps are *nonegocentric*, abstract, learned representations of causal relations among events, and these representations allow children to make causal predictions and anticipate the effects of interventions [17, 18]. Nonegocentric means that maps are independent from the specific relation between the learner and the environment where he/she is situated (e.g., children spatial position with respect to the position of objects in the environment). They are similar to *cognitive maps* [13], but specialised for causal knowledge, constituting a halfway point between domain-specific and domain-general representations. In literature, Causal Graphics Model and Causal Bayesian Net/Reasoning have been adopted to formalise from a computational point of view the idea [41, 50].

Two key features of this modelling that are particularly relevant here are: *(i)* the capability to describe structured models that represent hypothesis about how the world works; *(ii)* the capability of describing probabilistic relations between these models and patterns of evidence in rigorous ways. As a consequence, this approach is capable of representing conceptual, knowledge-level structures that allow both for prediction and learning, in an integrated way.

Indeed in AI literature, causal reasoning and probabilistic models have been largely explored and successfully adopted as reference computational framework to design and implement intelligent agents capable of dealing with the complexities of real-world environments [40, 42]. In the DevL perspective proposed in this paper, it is interesting to explore causal learning and reasoning as the core mechanism of the domain-independent learning system in agent cognitive architectures (and models as well). More in particular, the insights from developmental learning could be useful to devise a learning causal system at the architectural level in which probabilistic models are used to describe the structured models that represent hypothesis about how the world works (causal maps) and probabilistic relations between these models and patterns of evidence. These mental models can be used by the cognitive agent to generate predictions, and, at the same time, can be constructed and revised as part of the learning process, from evidence and observations.

4 RETHINKING AGENT PROGRAMMING AND AGENT ENGINEERING PROCESS

The DevL perspective leads to rethink quite radically the concept of agent program and agent programming, and, more generally, the full spectrum of agent development/engineering process. From an agent developer, agent programming becomes more similar to a teaching/instructing activity, occurring in a learning environment properly designed by the developer—possibly involving both direct communication between the agent and the programmer (playing the role of instructor, teacher) as well as other agents (peer-to-peer/transfer learning), and environment-mediated interaction. In that view, agent programming languages are conceptually reframed as communication languages explicitly designed to effectively supporting learning from the agent perspective, or teaching/training from the developer perspective.

Current BDI agent programming languages [8, 9] can be considered a kind of extreme case, in which learning is reduced to dumb acquisition by the learner of a set of scripts/plans embedding a procedural knowledge that the learner is meant to strictly follow, without reasoning asking why. Actually, the same holds if we consider the typical training process in the case of RL agents, in that the agent would update its e.g. policies after receiving a reward, without reasoning or asking why about that reward. In the DevL perspective, the language and the learning environment supporting the development process should be effective in allowing the learner agent to incrementally construct and revise its theories about the world (its task environment), *understanding why* by exploiting its causal learning system [43].

Besides the programming stage, the DevL view leads to revisit also the full traditional engineering process, involving multiple stages, such as analysis, design, implementation, testing and validation. By assuming a learning process perspective, then:

- the analysis would include focusing the learning objective, i.e. what the agent is meant to learn—expected skill and knowledge;
- the design would include the design of the learning environment—learning methods, tools to be adopted;
- testing and validation would occur through different kinds of *assessments*, during the learning process.

As soon as the learning process as well as the assessment are completed, the agent is ready to be deployed in the real-word. In the operational stage (runtime), monitoring is important to have feedbacks about agent performances, identifying possible problems also related to possible differences between the actual operational field and the learning environment used to train the agent. This information could be valuable to update and refine the learning environment itself. Besides, at runtime a major role could be played by agent self-assessment, and the capability to go on learning to eventually improve performances.

4.1 The Learning Path

The agent development process accounts for setting up a proper learning path for the agent, from the design time to runtime, supporting an incremental progress "from novice to expert". In the human case, the careful design of this path is fundamental for a successful learning, and typically involves multiple stages, from an instructor-dependent learning stage to a more self-directed learning stage.

In the instructor-led learning stage, the developer's role is to provide strategic guidance to the learner agent in identifying "zones of development", so that the learning objective would be *proximal*, within reach. In psychology, these zones are called zones of proximal development (ZPDs), a key construct introduced by Lev Vygotsky [55]. Self-regulated learners agents generate outcomes by engaging intentionally and deliberately in productive learning activity – like in the case of *sapient agents* [21, 54] – that is, using tools, resources, artifacts [38] that are part of the learning environment, to mediate learning.

When it comes to achieving long-term goals, the developer must consider how to create successive learning experiences, each building on the prior, to move the learner agent closer to achieving the distal goal. In the human case, these successive experiences are cumulative zones of proximal development [55], and, as a whole, comprise a zone of distal development (ZDD). The ZDD requires the learner and educator to be metacognitive – that is: aware of how they are planning, engaging and contemplating the acquisition of tools for teaching and learning, involving the strategic choice of methods of instruction and learning. In the design of the intelligent agents, the concept of ZPDs and ZDD appear interesting from a methodological point of view, allowing for shaping the development process in an incremental way, integrating both the design time and runtime, towards the engineering of lifelong learner agents.

5 RESEARCH DIRECTIONS

We complete the paper by identifying and discussing some main research directions that appear relevant to develop the DevL idea.

Cognitive Architectures. The idea calls for devising cognitive architectures for intelligent agents eventually integrating, on the one hand, the core abstractions and capabilities about practical reasoning and problem solving as displayed by existing model/architectures (e.g. BDI, SOAR), and, on the other hand, a proper learning causal systems, essential to support developmental learning. In the BDI case, for instance, this calls for extending the model to integrate at the foundational level practical reasoning and practical learning. Following the insights proposed in this paper, a main point in this direction is about exploring extensions of the BDI model/architecture in which causal maps and causal learning arefi rst-class concepts, to be proper understood and related to the other BDI key concepts such as beliefs, goals, intentions. To this purpose, existing works proposing an integration of Bayesian networks and Probabilistic Graphic Models in BDI [11, 14] could provide relevant insights, as well as work on probabilistic programming [19].

Developmental Learning and Friends. The research investigation about concrete computational models for DevL can benefit from results in literature about other kinds of learning that have strong affinities with the idea discussed in this paper. A main one is continual or lifelong learning [39, 53], which is about the ability to continually learn over time by accommodating new knowledge while retaining previously learned experiences. Another one is curriculum learning [34], which can be useful to shape the learning path discussed in Section 4.1, defining curricula of progressively harder tasks. DevL calls for exploring and understanding their value from an engineering perspective, as a way to systematically support the agent development process.

Learning environments. The design of proper learning environments is a key aspect of the idea, playing an essential role in the agent development process. Accordingly, a main issue is about the definition of methods and tools to ease the design and development of proper learning environments, as well as for defining a well-defined metamodel for them, capturing key conceptual aspects.

Generally speaking a learning environment could be a rich system, including e.g. domain-specific simulated (task) environments and assessment test-beds. Actually, to support the learning path devised in Section 4.1, the design process of an intelligent agent (the learning process) may call for developing multiple learning environments, featuring different levels of complexity and abstractions. Furthermore, a learning environment could concern contexts involving different forms of social interactions and communication, as well as social/institutional contexts characterised by different kinds of social laws and norms. Accordingly, DevL research may benefit from existing literature about learning e.g. in normative and institutional environments [6], to be reframed here in an agent development perspective, as part of the agent learning path.

From Individual Agents to MAS and Organisations. The focus of this paper has been on the design and development of individual intelligent agents. Nevertheless, from an engineering point of view, it is interesting to consider a broader MAS perspective [23], in which the agent dimension can be fruitfully integrated with other orthogonal dimensions, including agent organisations and environments [7]. As mentioned in previous section, social interaction and communication arefi rst-class dimensions in MAS, and are crucial elements in developmental learning of children too.

This triggers investigation about the meaning and value of developmental learning at the MAS level, that is: conceiving the design and development of e.g. organisation of agents as a *collective learning process*, which may concern not only a set of individual unrelated learning processes, but group-level strategies, that may take place in shared learning environments.

Platforms and Tools. Current agent-based platforms and IDEs (Integrated Development Environment) provide features that are quite similar to those found in platforms for mainstream programming paradigms—related to editing compiling code, running, debugging. The DevL idea calls for deeply rethinking the concept of IDEs, towards environments that foster a deeper integration between the design time and runtime, like in the case of live programming environments [33]. A recent work in that direction is [3].

Besides the integration between design and runtime, it is interesting here to explore IDEs proving afi rst-class support for the development process shaped as learning process, possibly involving the design and deployment of learning environments, in a learning path, and the design and execution of proper assessment stages, that correspond to testing and validation. This would have an impact also on the approaches to be used for important aspects such as agent debugging: in that perspective, debugging based on why questions [58] appears a natural approach to be considered here.

Besides IDEs, an important role is played also by the tools part of the runtime platform useful for monitoring agent (and MAS) performances, once deployed in the real-world, out of the learning environment. Besides the classic monitoring capabilities provided by tools in the mainstream (e.g. for microservices [22]), the monitoring tools at the DevL level could analyse the feedbacks coming from the real-world about agent performances, along with the historical data about the learning process engaged by the agent and the learning environments used. This could be interesting, for instance, to predict and intercept problems that could be raised in situations not properly considered in the learning environment, during the learning process.

6 CONCLUDING REMARKS

Psychology has been a source of inspiration for several important concepts that became a reference in the research community interested to agent/MAS programming and engineering. A main example is given by the BDI model and architecture. In this paper we propose to "go to the children", meaning to be inspired by studies in psychology about theory of learning and cognitive development, to envision a developmental learning approach (DevL) for developing intelligent agents.

In that approach, agent development is rethought as a learning process in which agents incrementally acquire practical knowledge and skills, by interacting in a proper learning environment. Causal learning and reasoning is discussed an essential core at the agent architecture level, to support developmental learning.

REFERENCES

[1] Stephane Airiau, Lin Padgham, Sebastian Sardina, and Sandip Sen. 2009. Enhancing the Adaptation of BDI Agents Using Learning Techniques. *Int. J. Agent Technol. Syst.* 1, 2 (April 2009), 1–18.

[2] Eduardo Alonso, Mark D'Inverno, Daniel Kudenko, Michael Luck, and Jason Noble. 2001. Learning in multi-agent systems. *The Knowledge Engineering Review* 16, 3 (2001), 277–284.

[3] Cleber Jorge Amaral and Jomi Fred Hübner. 2020. Jacamo-Web is on the Fly: An Interactive Multi-Agent System IDE. In *Engineering Multi-Agent Systems*, Louise A. Dennis, Rafael H. Bordini, and Yves Lespérance (Eds.). Springer International Publishing, Cham, 246–255.

[4] John Anderson. 2007. *How Can the Human Mind Occur in the Physical Universe?*

[5] Volodymyr Mnih at al. 2015. Human-level control through deep reinforcement learning. *Nature* 518 (2015), 529–533.

[6] Tina Balke, Célia da Costa Pereira, Frank Dignum, Emiliano Lorini, Antonino Rotolo, Wamberto Vasconcelos, and Serena Villata. 2013. Norms in MAS: Definitions and Related Concepts. In *Normative Multi-Agent Systems*. Dagstuhl Follow-Ups, Vol. 4. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany, 1–31.

[7] Olivier Boissier, Rafael H. Bordini, Jomi F. Hübner, and Alessandro Ricci. 2019. Dimensions in programming multi-agent systems. *The Knowledge Engineering Review* 34 (2019).

[8] Rafael H. Bordini, Mehdi Dastani, Jürgen Dix, and Amal El Fallah-Seghrouchni (Eds.). 2005. *Multi-Agent Programming: Languages, Platforms and Applications*. Springer.

[9] Rafael H. Bordini, Mehdi Dastani, Jürgen Dix, and Amal El Fallah-Seghrouchni (Eds.). 2009. *Multi-Agent Programming, Languages, Tools and Applications*. Springer.

[10] Rafael H Bordini, Amal El Fallah Seghrouchni, Koen Hindriks, Brian Logan, and Alessandro Ricci. 2020. Agent programming in the cognitive era. *Autonomous Agents and Multi-Agent Systems* 34 (2020), 1–31.

[11] Yingke Chen, Jun Hong, Weiru Liu, Lluis Godo, Carles Sierra, and Michael Loughlin. 2013. Incorporating PGMs into a BDI Architecture. In *Principles and Practice of Multi-Agent Systems (Lecture Notes in Computer Science, Vol. 8291)*. Springer, 54–69.

[12] Thomas G. Dietterich. 2004. Learning at the knowledge level. *Machine Learning* 1 (2004), 287–315.

[13] Russell Epstein, Eva Zita Patai, Joshua Julian, and Hugo Spiers. 2017. The cognitive map in humans: Spatial navigation and beyond. *Nature Neuroscience* 20 (10 2017), 1504–1513.

[14] Moser Fagundes, Rosa Vicari, and Helder Coelho. 2007. Deliberation Process in a BDI Model with Bayesian Networks. In *Proceedings of the 10th Pacific Rim International Conference on Multi-Agents, PRIMA 2007 (Lecture Notes, Vol. 5044)*. Springer, 207–218.

[15] Fabio Falcini, Giuseppe Lami, and Alessandra Mitidieri Costanza. 2017. Deep Learning in Automotive Software. *IEEE Softw.* 34, 3 (may 2017), 56–63. https://doi.org/10.1109/MS.2017.79

[16] Alison Gopnik. 2011. The Theory Theory 2.0: Probabilistic Models and Cognitive Development. *Child Development Perspectives* 5 (08 2011), 161 – 163.

[17] Alison Gopnik, Clark Glymour, David Sobel, Laura Schulz, Tamar Kushnir, and David Danks. 2004. A Theory of Causal Learning in Children: Causal Maps and Bayes Nets. *Psychological review* 111 (02 2004), 3–32.

[18] Alison Gopnik and Henry M. Wellman. 2012. Reconstructing constructivism: causal models, Bayesian learning mechanisms, and the theory theory. *Psychological bulletin* 138, 6 (2012), 1085–1108.

[19] Andrew D. Gordon, Thomas A. Henzinger, Aditya V. Nori, and Sriram K. Rajamani. 2014. Probabilistic Programming. In *Future of Software Engineering Proceedings* (Hyderabad, India) *(FOSE 2014)*. ACM, New York, NY, USA, 167–181.

[20] Alejandro Guerra-Hernández, Amal El Fallah-Seghrouchni, and Henry Soldano. 2005. Learning in BDI Multi-agent Systems. In *Computational Logic in Multi-Agent Systems*. Springer-Verlag, Berlin, Heidelberg, 218–233.

[21] Alejandro Guerra-Hernández and Gustavo Ortiz Hernandez. 2008. *Toward BDI Sapient Agents: Learning Intentionally*. Springer London, 77–91.

[22] Robert Heinrich, André van Hoorn, Holger Knoche, Fei Li, Lucy Ellen Lwakatare, Claus Pahl, Stefan Schulte, and Johannes Wettinger. 2017. Performance Engineering for Microservices: Research Challenges and Directions. In *Proceedings of the 8th ACM/SPEC on International Conference on Performance Engineering Companion* (L'Aquila, Italy). ACM, New York, NY, USA, 223–226.

[23] Nicholas R. Jennings. 2001. An Agent-Based Approach for Building Complex Software Systems. *Commun. ACM* 44, 4 (apr 2001), 35–41.

[24] Iuliia Kotseruba and John K. Tsotsos. 2018. 40 years of cognitive architectures: core cognitive abilities and practical applications. *Artificial Intelligence Review* 53 (2018), 17–94.

[25] John Laird. 2008. Extending the Soar Cognitive Architecture. *Frontiers in Artificial Intelligence and Applications* 171, 224–235.

[26] John Laird, Paul Rosenbloom, and Allen Newell. 1986. Chunking in Soar: The Anatomy of a General Learning Mechanism. *Machine Learning* 1 (03 1986), 11–46.

[27] John E. Laird. 2012. *The Soar Cognitive Architecture*. The MIT Press.

[28] John E. Laird, Christian Lebiere, and Paul S. Rosenbloom. 2017. A Standard Model of the Mind: Toward a Common Computational Framework across Artificial Intelligence, Cognitive Science, Neuroscience, and Robotics. *AI Magazine* 38, 4 (Dec. 2017), 13–26.

[29] J. E. Laird, A. Newell, and P. S. Rosenbloom. 1987. SOAR: An Architecture for General Intelligence. *Artificial Intelligence* 33 (1987), 1–64.

[30] Pat Langley, John E. Laird, and Seth Rogers. 2009. Cognitive architectures: Research issues and challenges. *Cognitive Systems Research* 10, 2 (2009), 141–160.

[31] Antonio Lieto. 2021. *Cognitive Design for Artificial Minds*. Routledge.

[32] Soul Machines. 2022. BabyX project. https://www.soulmachines.com/resources/research/baby-x/

[33] Sean McDirmid. 2007. Living It up with a Live Programming Language. *SIGPLAN Not.* 42, 10 (oct 2007), 623–638.

[34] Sanmit Narvekar, Bei Peng, Matteo Leonetti, Jivko Sinapov, Matthew E. Taylor, and Peter Stone. 2020. Curriculum Learning for Reinforcement Learning Domains: A Framework and Survey. *J. Mach. Learn. Res.* 21 (2020), 181:1–181:50.

[35] Allen Newell. 1982. The knowledge level. *Art. Int.* 18, 1 (1982), 87–127.

[36] Allen Newell. 1990. *Unified Theories of Cognition*. Harvard University Press, USA.

[37] Emma Norling. 2004. Folk Psychology for Human Modelling: Extending the BDI Paradigm. In *Proc. of the Third Int. Joint Conf. on Autonomous Agents and Multiagent Systems (AAMAS '04)* (New York, New York) *(AAMAS '04)*. IEEE Computer Society, Washington, DC, USA, 202–209.

[38] Andrea Omicini, Alessandro Ricci, and Mirko Viroli. 2008. Artifacts in the A&A Meta-Model for Multi-Agent Systems. *Autonomous Agents and Multi-Agent Systems* 17, 3 (dec 2008), 432–456.

[39] German I. Parisi, Ronald Kemker, Jose L. Part, Christopher Kanan, and Stefan Wermter. 2019. Continual lifelong learning with neural networks: A review. *Neural Networks* 113 (2019), 54–71.

[40] Judea Pearl. 1988. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

[41] Judea Pearl. 2000. *Causality: Models, reasoning, and inference*. Vol. 29. Cambridge University Press.

[42] Judea Pearl. 2019. The Seven Tools of Causal Inference, with Reflections on Machine Learning. *Commun. ACM* 62, 3 (feb 2019), 54–60.

[43] Judea Pearl and Dana Mackenzie. 2018. *The Book of Why: The New Science of Cause and Effect* (1st ed.). Basic Books, Inc., USA.

[44] Anand S. Rao and Michael P. Georgeff. 1995. BDI Agents: From Theory to Practice. In *Proc. of the First Int. Conf. on Multi-Agent Systems (ICMAS-95*. 312–319.

[45] Paul S. Rosenbloom, Abram Demski, and Volkan Ustun. 2016. The Sigma Cognitive Architecture and System: Towards Functionally Elegant Grand Unification. *Journal of Artificial General Intelligence* 7 (2016), 1 – 103.

[46] David Silver, Satinder Singh, Doina Precup, and Richard S. Sutton. 2021. Reward is enough. *Artificial Intelligence* 299 (2021).

[47] Dhirendra Singh and Koen V. Hindriks. 2013. Learning to Improve Agent Behaviours in GOAL. In *Programming Multi-Agent Systems*, Mehdi Dastani, Jomi F. Hübner, and Brian Logan (Eds.). Springer, Berlin, Heidelberg, 158–173.

[48] Dhirendra Singh, Sebastian Sardina, Lin Padgham, and Geoff James. 2011. Integrating Learning into a BDI Agent for Environments with Changing Dynamics. In *Proc. of the Twenty-Second Int. Joint Conf. on Artificial Intelligence (IJCAI '11) (IJCAI'11)*. AAAI Press, 2525–2530.

[49] Javier Snaider, Ryan McCall, and Stan Franklin. 2011. The LIDA Framework as a General Tool for AGI. In *Artificial General Intelligence*. Springer, Berlin, Heidelberg, 133–142.

[50] Peter Spirtes, Clark Glymour, and Richard Scheines. 1993. *Causation, Prediction, and Search*. Vol. 81.

[51] Richard S. Sutton and Andrew G. Barto. 1998. *Introduction to Reinforcement Learning* (1st ed.). MIT Press, Cambridge, MA, USA.

[52] Ah-Hwee Tan, Yew-Soon Ong, and Akejariyawong Tapanuj. 2011. A Hybrid Agent Architecture Integrating Desire, Intention and Reinforcement Learning. *Expert Syst. Appl.* 38, 7 (July 2011), 8477–8487.

[53] Sebastian Thrun and Lorien Pratt (Eds.). 1998. *Learning to Learn*. Kluwer Academic Publishers, USA.

[54] M. van Otterlo, M. Wiering, M. Dastani, and J.-J. Meyer. 2003. A characterization of sapient agents. In *IEMC '03 Proceedings. Managing Technologically Driven Organizations: The Human Side of Innovation and Change*. 172–177.

[55] L. S. Vygotskiĭ and Michael Cole. 1978. Mind in society: the development of higher psychological processes.

[56] B. J. Wadsworth. 1996. *Piaget's theory of cognitive and affective development: Foundations of constructivism*. Longman Publishing.

[57] Gerhard Weiß. 1996. Adaptation and learning in multi-agent systems: Some remarks and a bibliography. In *Adaption and Learning in Multi-Agent Systems*, Gerhard Weiß and Sandip Sen (Eds.). Springer, Berlin, Heidelberg, 1–21.

[58] Michael Winikoff. 2017. Debugging Agent Programs with Why? Questions. In *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems* (São Paulo, Brazil) *(AAMAS '17)*. IFAAMAS, Richland, SC, 251–259.

[59] Robert E. Wray and Randolph M. Jones. 2005. *Considering Soar As An Agent Architecture*. Cambridge University Press, 53–78.