Journal of Artificial General Intelligence 15(1) 3-61, 2025
DOI: 10.2478/jagi-2025-0002

Received: 2022-08-27
Accepted: 2024-01-22

In Search of Insight: My Life as an Architectural Explorer

Paul S. Rosenbloom ROSENBLOOM@USC.EDU
University of Southern California
941 West 37th Place
Los Angeles, CA 90089, USA

Editor: Christian Lebiere

Abstract

Looking back on my intellectual career of nearly a half century, from the 1970s to the 2020s, it is surprising—at least to me—that it was not until I began this memoir that I understood it was all about *architectural exploration*. Initially this focused on cognitive architectures for *integrated minds*, such as Soar and Sigma, but increasingly it has also involved architectures of *academic disciplines*—such as artificial (general) intelligence and computer/cognitive science—that study integrated minds. The theme of this memoir then emerged naturally from understanding this methodology—including a particular perspective on what it means to be an architecture and the continuing importance of exploration—and of its application to this pair of apparently disparate topics. This memoir thus develops this theme and traces it through my career.

Keywords: Mind, AI, AGI, cognitive science, computer science, architecture, exploration, mapping

1. Introduction

Throughout my life I have been a seeker, but of *intellectual insight* rather than spiritual enlightenment. By trade, I am an academic computer and cognitive scientist, with the core of my search having been the *nature of mind*, specifically in a manner that spans both artificial and natural variants, as studied in artificial intelligence (AI) and cognitive science. But later the search went meta, into the nature of the *disciplines that encompass (the study of) mind*—from AI and cognitive science most nearly, moving outward to computing science[1] within which both AI and the computational aspects of cognitive science find a home, and ultimately to the full scope of science and engineering, and in fact all intellectual endeavors. I have approached these various searches more as an *explorer* than a traditional theorist or experimentalist, with the aim of capturing the resulting insights as immutable aspects of integrated *architectures*. For example, the lion's share of my work has revolved around *cognitive architectures* that are to capture what is discovered about the nature of mind.

This memoir predominantly looks back at my life that was, but it also includes what is and helps to lay the groundwork for what may still come. It is a memoir rather than an autobiography in focusing on an overarching theme rather than on all of my life. The theme in this case is bipartite:

1. *Substantive*: Mind and its encompassing disciplines

[1] A broadening of the standard notion of computer science to include all of how computing is understood and applied (Rosenbloom, 2012a). More on this can be found in Section 6.

2. *Methodological*: Seeking insight via architectural exploration.

It is an intellectual memoir in that the story line concerns my research career rather than emotionally evocative episodes. It also includes speculations in philosophy of (computer and cognitive) science that proved essential in developing and applying the notion of architectural exploration, plus references (although not always as would be called for in a technical article). Although the focus is on the overall theme, it is accompanied by footnotes, figures, a pair of asides, and an interregnum[2] that cover key bits that don't fit the overall theme, whether personal or professional.[3]

The basics of my professional background comprise my education and subsequent academic positions. The former included a B.S. in mathematical sciences—a mixture of mathematics, computer science, operations research, and statistics—at Stanford University (1976),[4] an M.S. (1978) and Ph.D. (1983) in computer science (CS) at Carnegie Mellon University (CMU),[5] and a year (1978-1979) as a non-degree visiting graduate student in psychology at the University of California, San Diego (UCSD) that was sandwiched between the M.S. and any serious work toward the Ph.D. at CMU.[6] As a result, I was formally trained on both computing and the mind but remain little more than a fitfully educated amateur in philosophy of science despite it's becoming increasingly important in my later work.

I never planned on going academic, and with early phobias concerning both writing and speaking—with the latter due at least in part to a (self-diagnosed) position at the mild end of the autism spectrum—and no initial clue about research, I was clearly not a natural at it from the start. Yet, when I was finishing my dissertation, Allen Newell—my Ph.D. advisor—surprised me by suggesting I consider universities. Once I explored the options, I did end up taking a chance and choosing an academic position. Eventually I became semi-comfortable at teaching and leading groups, and quite comfortable with writing and research, enabling a challenging yet satisfying career in which I could largely determine my own path.

My major academic positions have been as a Research Computer Scientist[7] at CMU (1983-1984);[8] returning to Stanford as an Assistant Professor of Computer Science and Psychology (1984-1987); and then moving to what would become the Viterbi School of Engineering at the University of Southern California (USC), first as Research Staff (1987-1988), then as a Research Assistant Professor of Computer Science (1988-1990), followed by a tenured (Associate and then Full) Professor of Computer Science (1991-2022),

[2] Classically, an interregnum is a period between rulers, but here the term is used in the more general sense of a "pause in continuity," in particular among cognitive architectures.

[3] An older version of the "author's cut" of this memoir includes much more on personal matters and key aspects of my career that don't fit the overall theme. A newer variant of this author's cut can also be found in Rosenbloom (2025).

[4] Despite computer science having become such a central part of Stanford over the years, this was many years before the dot-com boom and Stanford did not even offer an undergraduate major in it at the time.

[5] I had not heard of CMU until late in my undergraduate career at Stanford, when I took a graduate survey course on AI taught by Cordell Green. The course included a video of Allen Newell talking about CMU's work on speech understanding. When I visited CMU as part of my decision-making process for graduate school, I was greatly impressed both with Allen's vision for AI and the warm and communal nature of its Computer Science Department. It was small, including only a Ph.D. program, and had a wonderful camaraderie among the students, and even with the faculty. So, although I had not heard of CMU until the year before, and it was in Pittsburgh of all places, I ended up choosing it over MIT and Stanford. Fortunately, Allen turned out to be a terrific Ph.D. advisor and CMU turned out to be a wonderful environment in which to pursue a CS Ph.D.

[6] This year at UCSD was initially motivated by personal considerations, but at a minimum it taught me how to think like a psychologist and to appreciate the literature in its own right, rather than always approaching it from an AI perspective. More about this year will be said in Section 3.

[7] This was a junior position on the *research faculty*; a nontenured faculty track where financial support came from external research contracts and grants rather than from teaching.

[8] Upon finishing my dissertation in 1983, I was just about to accept an offer from the University of California, Berkeley (UCB) when Allen Newell suggested I stay at CMU and team with him and John Laird on the Soar Project. After some hard thought, I agreed, ultimately setting the course for the rest of my career. When, within one more year, I felt an overwhelming need to get back to California, UCB was no longer a possibility, so I picked the other main option I had perseverated over the previous year and joined Stanford's faculty. John left for Xerox PARC at the same time, and he just reminded me at my retirement celebration that Allen was so high on the potential of Soar and so worried about the impact of our leaving on it that he told us it was in the national interest for us to stay in Pittsburgh.

before ultimately becoming a Professor Emeritus (2022-). During my time at USC, I also had a home at its Information Sciences Institute (ISI) (1987-2007)—where I actually started at USC before joining what is now the Thomas Lord Department of Computer Science—and its Institute for Creative Technologies (ICT) (2008-).

The overall inspiration for this memoir was a pair of talks given near the end of their careers by two other CMU computer scientists, Allen Newell, and Randy Pausch. Allen was my Ph.D. advisor, mentor, second father figure, colleague, and friend. He often had more confidence in me through the early years than I had in myself. His talk on *Desires and Diversions* discussed styles of research careers, with a particular emphasis on his own style—of focusing on a single *ultimate scientific question* (or *grand challenge*) throughout his career, which in his case was understanding the human mind—and the trajectory it yielded when supplemented by a number of significant diversions into other topics. I have shown a grainy old video of this talk to groups of students a number of times over the years to help them understand his, and to some extent my, career path and to help them think prospectively about their own careers.

Randy and I overlapped as graduate students, although he was several years behind. His talk, officially titled *Really Achieving your Childhood Dreams*, went viral as *The Last Lecture* before appearing in book form (Pausch, 2008). According to Wikipedia "This talk was modeled after an ongoing series of lectures where top academics are asked to think deeply about what matters to them, and then give a hypothetical "final talk", that is, "what wisdom would you try to impart to the world if you knew it was your last chance?""

My own memoir, which was begun at a critical juncture in my life—on the cusp of retirement and battling cancer—has been an attempt at an extended written version of such a lecture,[9] with much of the "wisdom" I would try to impart listed as maxims in Appendix A, and cited where appropriate throughout this work (for example, Maxim 13, which has applied here, of writing what you want to say before considering where to publish it). In the process, it has provided an opportunity to both review and frame my life and work, with the intention of including both the good and the bad; and both strengths and weaknesses to the extent that they seem relevant and that I understand them. One key outcome has been a better understanding of both me and my overall research approach/trajectory. It was not, for example, until this effort that I was able to articulate, or even to truly understand, either part of the theme that has ruled my career.[10] When later I was asked to give a *Remarkable Trajectory Lecture*—USC Viterbi School of Engineering's own "final talk" series—this memoir was the foundation for my presentation.

The methodological subtheme—of seeking insight via architectural exploration—is discussed in Section 2. Following introductory material that positions this approach with respect to more traditional scientific methodologies are subsections on architecture and exploration.[11] Subsection 2.1 examines the core notion of an architecture and how it relates to other similar concepts, such as theory, model, framework, and research programme. In the process, new ground is broken in exploring how to broaden the notion of an architecture beyond how it is commonly used in computing to include more abstract conceptualizations. Included in this are some of the key properties of such architectures.

Subsection 2.2 examines what it means to be an explorer rather than a theorist or an experimentalist, with an emphasis on venturing into and mapping complex phenomena to yield insight into their scope and structure rather than emphasizing firmly nailing down their nature via either formal proofs or systematic experimentation. Exploration is an essential part of any enterprise geared toward increasing understanding,

[9] The proximate trigger, however, was my sister Kate Rosenbloom Pohl's *Swan Song* as she retired from a career that included extensive work on the UCSC Genome Browser. It got me thinking about trying something myself at this point.

[10] As a memoir, this work is inherently self-centered, self-indulgent, and presumptuous, not traits greatly admired in academic writing. But writing it felt like the right thing to do under the circumstances, and the hope is that it will be of some use to others.

[11] This is a discussion that might more typically be found in an academic paper in philosophy of science than a memoir, but most of it has been worked out in the process of writing this memoir, it helps to understand the later material, and it provides a prime example of how I think at this point in my life, so it is included here.

but as it is not grounded in what is generally accepted as the scientific method, such work may often be considered as either prescientific or nonscientific.

The remainder of the story line focuses on the substantive subtheme, concerned with mind and its encompassing disciplines. Rather than following a chronological ordering, this material proceeds through four tiers of architectural exploration that together comprise my work on this subtheme (Figure 1). The innermost two tiers (1 & 2) focus on the mind. The outermost tier (4) focuses on computing, while also laying the groundwork to go significantly beyond this to span all intellectual endeavors. The intermediate tier (3) turns out to supplement both the inner tiers' focus on the mind and the outer tier's focus on disciplines, although with the latter narrowing to AI and cognitive science. Each tier is distinctively characterized by the nature of the architecture(s) involved.

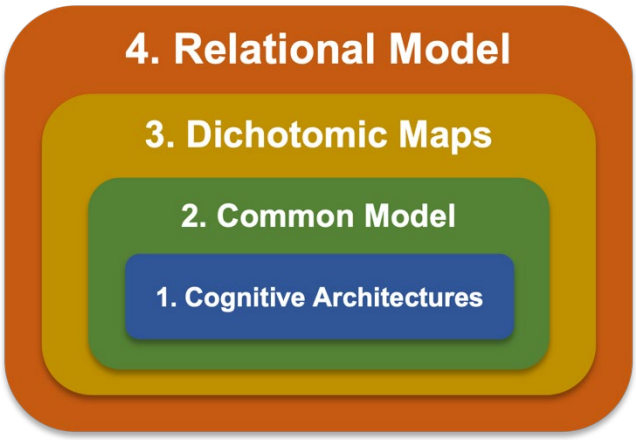

Figure 1: The four tiers of architectural exploration that comprise the substantive subtheme.

Section 3 covers Tier 1, concerning the grand challenge that has consumed most of my career (Maxim 15), of developing cognitive architectures that aid in understanding the human mind and in understanding and building artificial minds. In particular, a sequence of three cognitive architectures—XAPS (Rosenbloom, 2006) ➜ Soar (Laird, Newell, and Rosenbloom, 1987) ➜ Sigma (Rosenbloom, Demski, and Ustun, 2016a)—each of which perhaps might be better thought of as an *architectural lineage*,[12] will be examined, along with a bit of relevant work that occurred prior to this sequence. This work aligns naturally with Allen Newell's talk, given that much of the early segments were done in collaboration with him, and with a grand challenge akin to his ultimate scientific question.

Section 4 covers Tier 2, concerning the *Common Model of Cognition*, a more abstract architecture intended to capture a consensus over humanlike minds (Laird, Lebiere, and Rosenbloom, 2017).[13] Section 5 covers Tier 3, concerning *dichotomic maps* that aid in understanding the space of technologies underlying AI and cognitive science, and the germane combinations of such technologies for building cognitive architectures and other kinds of complex intelligent systems (Rosenbloom, 2019; Rosenbloom, Joshi, and Ustun, 2019; Rosenbloom, 2025). This is the newest body of work and still very much in progress. Section 6 covers Tier 4, concerning the *Relational Model* that provides an architecture that is to span all of computing (Rosenbloom, 2012a) and potentially beyond that to all intellectual activity. Until writing this memoir and formulating its substantive subtheme, I had considered Tier 4 as a diversion rather than as thematic; however, with the more recent development of Tier 3 and the relationship of its first half to Tier 4, my perspective on this has evolved.

Section 7 concludes with reflections on this memoir's theme. It also starts to consider what these four tiers and their interrelationships imply about my career so far and what it might make sense to continue pursuing into the future.

[12] A lineage of architectures, as the phrase is used here, is a sequence of architectures that typically starts with one simple yet concrete version and evolves over time—normally via human programming rather than natural variation and selection—into successively more advanced versions. All of the architectures within such a lineage typically share a single name, just with added modifiers or increasing version numbers over time.

[13] It is only recently that I have begun to think of the Common Model and the higher tiers as involving architectures, leading to the discussion of this topic to be found in Section 2.1.

2. Methodology: Seeking Insight via Architectural Exploration[14]

Methodologically, I am too often thrilled neither by new theorems and proofs, as would befit a theorist; nor by new experiments and their results, as would befit an experimentalist. Part of this is certainly that those that can be accomplished, which are largely on parts of architectures, so often do not get at the issues concerning architectures that I truly care about. However, part of my disinclination toward traditional theory may also be due to reaching an understanding as an undergraduate that I wasn't going to be a mathematician.[15] Another part may be due to too many attempts at formalization in AI over the years that appear to obscure rather than clarify the points being made. My disinclination toward traditional experiments may additionally be partly due to their inherent messiness.[16]

What turns out to really turn me on intellectually are new *insights—aha* moments that yield a new way of looking at things (Maxim 1). A recent article provides evidence for this as a third route to a good life, beyond the traditional two of pursuing *happiness* (*hedonics*) or *meaning* (*eudaimonics*), that instead seeks "*a psychologically rich life*—or a life characterized by a variety of interesting and perspective-changing experiences" (Oishi, and Westgate, 2021). In other words, it proposes the rewards of being an *aha junkie*. Meaning still does matter to me, coming from pursuing, via architectural exploration, impactful topics in which I have sufficient ability to make a difference (Maxim 14). Traditional notions of happiness have played a much lesser role in my intellectual life, although they have not been totally absent.

In search of insights, I have found myself repeatedly drawn to a research methodology that combines *exploring* complex phenomena with seeking to impose order on them by identifying a fixed conceptualization—an *architecture*—that both embodies any core insights and authorizes a space of more flexible *variations* that flesh out its capabilities. Together, an architecture and a particular set of variations yields a *system*, which can succinctly be semi-formalized as *system = architecture + variations*. Such systems may be conceptual, mathematical, computational, physical, etc. An *application* is then simply a system that yields value of some sort.

Simplistically, exploration seeks insights and architectures capture them.[17] A good example is the work discussed in Section 3.1 on modeling the *power law of practice.* From a regularity in human behavior an architectural learning mechanism was developed, called *chunking*, that could be used across a broad range of tasks. Yet, the full relationship between exploration and architecture is more multifarious than is indicated by this simple aphorism. Architectures can, for example, guide the exploration and interpretation processes that lead to new insights. They can also guide decisions about how and when the architecture

[14] A recent adaptation of this section for an A(G)I audience can be found in (Rosenbloom, 2022).

[15] I took math honors classes as a freshman at Stanford, and while I did fine grade-wise, I learned that, as opposed to what I would later find in computer science, it was not truly natural enough for me to consider it as a career.

[16] This aversion to messiness could be due to my OCD nature, as I was so diagnosed when young. This may also be a key factor in why I am driven to seek architectures that can capture any order lurking beneath the messiness found in complex phenomena, and why studying the brain rather than the mind has never had much appeal to me.

[17] Rosenbloom (2025) has articulated a methodology of *essential analysis* that, although it sounds distinct from architectural exploration, turns out to be more a different perspective on it that comes to the fore when the architecture is quite simple—such as comprising a minimal definition of a term—and so most of the exploration takes the form of analyzing the implications and means of decomposing the reformulated definition. The new definition of the term is intended to go beyond what many would find comfortable in order to identify its very essence. Essential analysis can also be seen as a variation on Maxim 21, of seeking similarities before considering differences. It is a bit ironic that I disliked my first philosophy course, on *ethics*, because it focused more on definitions of terms such as *ought* than on actual ethical systems, but now I spend a good deal of time attempting to (re)define terms such as *computing* (Rosenbloom, 2012a), *(cognitive) architecture* (Rosenbloom, 2022; Rosenbloom, 2025), *theory* (Rosenbloom, 2025), and *symbol* (Rosenbloom, 2023; Rosenbloom, 2025).

itself needs to be modified.[18] Allen Newell called this *listening to the architecture*.[19] I won't attempt a systematic exploration of this topic here, but a few additional examples may be helpful.

- Linnaean taxonomy (Linnaeus, 1735) has aided in interpreting what is uncovered during exploration of the natural world. Although the notion of a taxonomy provides too weak an architecture to yield much guidance in exploration, once a new species is discovered the taxonomy can help interpret it in terms of the hierarchy of known species.
- The understanding of *data chunking* in Soar (Section 3.2), as a particular interaction between search and chunking, arose from understanding major gaps in the architecture's ability to support declarative (fact) learning and memory capabilities. In this case, listening to the architecture led to an approach that surprisingly didn't require adding new architectural mechanisms. However, as will be seen, it was ultimately superseded in Soar by new declarative memories and learning mechanisms.
- A much less satisfactory example concerns how I have struggled for years to implement a new architectural mechanism for chunking in Sigma, attempting to use the architecture as it has evolved over this time to guide how best to do this. Unfortunately, it is an example for which the guidance has never been quite sufficient; or, perhaps, simply an example of where I have been too tone deaf to hear it properly.
- A much more positive example from Sigma is how two other capabilities that were already well studied in the field but that were new to Sigma—reinforcement learning (Rosenbloom, 2012b) and episodic memory and learning (Rosenbloom, 2014)—were deconstructed in terms of combinations of pre-existing mechanisms. As with data chunking, what were traditionally distinct modules were instead reinterpreted as combinations of more basic mechanisms. These novel reinterpretations yielded some of the strongest *aha* moments of my career, moments when it felt like a deeper understanding was achieved based on taking a new perspective.
- Another positive example from Sigma was how its basic architectural mechanisms were extended to handle feedforward neural networks, without adding a distinct neural module (Section 3.4).
- A recent push in Sigma to explore emotion, motivation, and personality exemplifies how understanding a combination of the major gaps in an architecture's coverage—in this case concerning "personal" aspects—and what topics it may be most ready to handle (or be extended to handle) can lead to appropriate selection of topics to explore and corresponding architectural modifications to implement.
- The *Common Model of Cognition* (Section 4) guided new analyses of functional brain scans that have been used to compare it to more traditional approaches from neuroscience for modeling the high-level architecture of the human brain. Fortunately for the Common Model, it turned out to dominate the traditional approaches.

Beyond simply adding mechanisms or capabilities to existing architectures, it is also possible to talk explicitly about exploring the space of architectures. This can amount to, for example, searching for a computer architecture that enables secure, energy-efficient, high-speed computation; or iteratively developing a cognitive architecture that yields a good model of the fixed structures and processes underlying the human mind. A lineage of architectures can thus be considered as a search within a relatively compact region of the space of all possible architectures, with each distinct lineage searching its own particular region of the overall space.

[18] Maxims 16 and 20-22—milking the existing architecture/hypothesis for all it is worth while focusing on: simplicity in terms of the number of architectural mechanisms, similarity between new and old capabilities, and creating explanations in terms of (combinations of) existing mechanisms rather than immediately jumping to add new mechanisms that are disjoint from existing ones—provide further perspective on such decisions.

[19] The painter Helen Frankenthaler said something similar about art: "Let the picture lead you where it must go."

In understanding and accepting this self-architecture—as an architectural explorer—late in life, I have freed myself to work on the problems I most care about via the methodology with which I am most comfortable (Maxim 19). There are, however, downsides to such an approach that will be discussed in the following two subsections on *architecture* and *exploration*. The philosophy of science that appears in these two sections arose from reflections on my research career, of what I did and didn't do, and even more so of what gave me joy rather than stemming largely from duty, obligation, or necessity.

More particularly, this material came out of an attempt to bring coherence, in terms of a common theme, to these results. In effect, this effort has itself been an example of architectural exploration, with exploratory reflections on my life, work, and history accompanied by an attempt to discover whether there is any kind of architectural fixity that could be articulated. However, as with any such post hoc rationalization, it should be viewed with an extra degree of skepticism as to its yielding any sort of general rule. Still, given that such (re)definitional efforts often reflect new perspectives—and thus new insights—they may still prove useful irrespective of their original motivations.

2.1 Architecture: Capturing Insights (and More)

The notion of an architecture as used here derives from that of a *cognitive architecture*, which provides a hypothesis concerning the fixed structures that jointly yield a mind when combined with the skills and knowledge that the architecture acquires and processes (Langley, Laird, and Rogers, 2009; Kotseruba, and Tsotsos, 2020). This notion is most familiar in cognitive science, where the architecture is intended to embody a theory of the human mind. However, it can also be found in AI and artificial general intelligence (AGI)—although sometimes under the variant names of an agent, AI, or AGI architecture—where it amounts to a fixed framework that supports the construction of intelligent systems.

The idea of a cognitive architecture was itself derived from the prior notion of a *computer architecture* (Bell, and Newell, 1971; Hennessey, and Patterson, 2012). From Wikipedia:

> In computer engineering, computer architecture is a set of rules and methods that describe the functionality, organization, and implementation of computer systems.
>
> Some definitions of architecture define it as describing the capabilities and programming model of a computer but not a particular implementation. In other definitions computer architecture involves instruction set architecture design, microarchitecture design, logic design, and implementation.

As I understand it, the aspects of this that most directly impacted the development of the notion of a cognitive architecture—given that Allen Newell was foundational in both—are "the capabilities and programming model," with the "instruction set architecture design"—that is, the language within which programming can occur—being a key aspect of the "programming model" that delimits the kinds of programs and data that can be processed.

Presumably the idea of computer architecture was itself derived from the older notion of *building architecture*, which has a broad meaning that includes the profession, the process, the design, the unifying concepts in the design, the results (that is, the buildings), and the relevant art and science. In this memoir the focus is on a generalization of the idea of architecture from the form it takes in studying computers and minds to one that is closest to "unifying concepts" in building architecture but that is useful in understanding a wider range of complex topics. The essence is the notion of a *fixed framework*[20] that enables and delimits a *range of variations*.

[20] Herbert Simon—a Nobel laureate in economics, Turing Award winner, and Allen Newell's Ph.D. advisor (and collaborator for many years), and thus my academic grandfather—discussed the importance of identifying *invariants* because of "their power to strip away the complexity and diversity of a whole range of phenomena and to reveal the simplicity and order underneath" (Simon,

By providing some level of fixity while delimiting the range of variations, an architecture inherently provides constraint on a body of phenomena, whether the goal is to model the phenomena or to produce it. Architectures, and thus the constraint they provide, can be minimal. For example, the concept of modules and connections can by itself comprise an architecture, as can the concept of a taxonomic hierarchy. Both provide a fairly minimal amount of fixed structure, but each does still ensure that certain kinds of properties will be met by any system based on it, such as, respectively, the ability to decompose and hide complexity within modules and to limit the number of children at each node in a hierarchy.

More basically, the constraint yielded by the simple act of decomposing systems into an architecture plus variations can bring benefits that are often ascribed instead to particular forms of architecture. For example, arguments have been made for both modules and hierarchies as enabling more robust systems that are easier to learn, develop, evolve, and engineer (see, for example, Simon, 1969). However, any architecture already provides at least an initial decomposition into fixed versus variable structure, and any substantive architecture will impose restrictions on the variable structure that further reduce its flexibility and complexity.

Another example comes from the world of cognitive architectures, where the argument has been made that the fixed structure of an architecture enables the possibility of one-time validation of the architecture followed by unlimited reuse with arbitrary variations (Ustun et al., 2018). As with the previous arguments, this one also holds for all architectures. It clearly, for example, applies to the use of architectures as theories, where validating the theory once may enable an endless variety of uses, as well as to the use of architectures as fixed mechanisms at the core of complex engineered systems.

This doesn't rule out that any particular type of architecture might provide even more benefits, but it does provide a baseline benefit shared by all architectures. It also doesn't rule out enabling computational universality—that is, the ability to compute whatever is computable—in the variations, as is almost definitional for a computer architecture but is also typical of cognitive architectures. Yet, even in such cases it limits the nature of the programs and data directly usable in the former and the skills and knowledge directly usable in the latter.

In general, different kinds of architectures enable different forms of variations. As we have just seen, computer architectures enable programs and data while cognitive architectures enable skills and knowledge. In addition, modular architectures enable module definitions and API (that is, application programming interface) specifications; hierarchical and graphical architectures enable node and link definitions; and tables and maps enable entries at appropriate locations. In tightly specified theories or models, in which nearly the entire system is architecture, the variations may amount simply to parameter values. In Lakatosian research programmes (Lakatos, 1978), they amount to auxiliary hypotheses.

Architectures represent complex phenomena when used in understanding the phenomena of interest, as is typical in the sciences. When used instead to generate complex phenomena, as is typical in engineering, architectures may also represent something but need not do so. A computer architecture, for example, is central to engineering all kinds of computational systems, yet its foremost purpose is to aid in the development of such systems rather than to represent or to help understand anything in particular. The architecture in such a case could conceivably be considered as a model of computation, as with a Turing machine (Turing, 1937), but this is deep in the background in most applications.

Two major classes of architectures have been central in my own work: (1) *computational architectures*; and (2) *theoretical architectures* (Figure 2). A computational architecture, in conjunction with appropriate variations, can compute, in the general sense of being able to transform information (Rosenbloom, 2010). Both computer and cognitive architectures are examples of computational architectures, whereas building architectures and many scientific theories are not. A theoretical architecture, with or without accompanying variations, represents—or "stands in for"—key aspects of a phenomenon or domain of interest.

1974). In this sense, an architecture can be viewed as a set of invariants. However, architectures also go beyond this to consider the interactions among the included invariants so as to yield a tightly integrated and coherent whole.

Scientific theories,[21] including many cognitive architectures, are examples of theoretical architectures—focused on *understanding* (Rosenbloom, 2012a)—whereas building and computer architectures typically are not. Atheoretical architectures such as these latter two are more naturally considered key components of a generalized notion of engineering that I have in the past referred to as *shaping* (Rosenbloom, 2012a) and that also includes other professions, such as law and medicine, that are about actively altering other aspects of the state of the world.

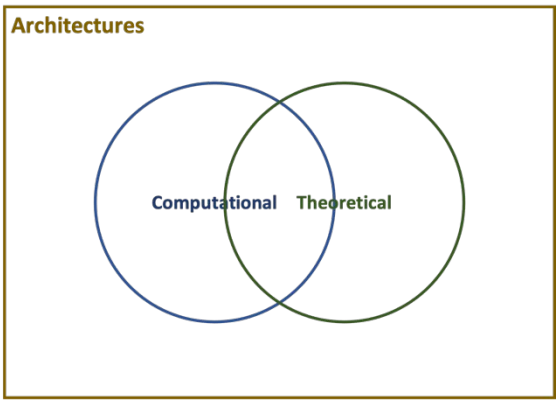

Figure 2: Space of architectures, highlighting overlapping classes of computational and theoretical architectures.

Cognitive architectures, according to this analysis, can be both computational and theoretical. They do, however, cede being computational when limited to a design on paper rather than an implemented and executable body of code. In the other direction, they cede being theoretical when they are purely engineering artifacts that are not directly about anything. Yet, even when not about human intelligence, they can potentially be about a more abstract domain of intelligent phenomena—perhaps considered as a Platonic ideal (Plato, ~375 BC)—and thus still be theoretical in this sense.

If intelligence is considered as one aspect of computation, the theoretical side of cognitive architectures could be considered as existing within the general area of *theory of computation*, the discipline within computer science concerned with understanding computing itself as a complex body of phenomena, including considering such questions as what can be computed and what can be computed tractably (including the million-dollar question of "P=NP?"). Here a sample question might then be "What can be intelligent?," although using cognitive architectures in answering this question may be a less formal approach than is the standard in theory of computation.

The neat picture provided by 2, with four clear regions spanning whether or not an architecture is computational and/or theoretical, can usefully be fuzzed up in multiple ways (Figure 3), but still without destroying the overall message. Consider, for example, scientific theories expressed as mathematical equations. Such theories directly provide theoretical architectures, but they typically aren't directly computational. Yet, it is often easy to convert them into a computational form that can directly transform measurements into predictions and simulations (Figure 3a). This contrasts with other theoretical architectures, perhaps expressed in English, or derived from the humanities, that may be far from computational architectures (Figure 3b).

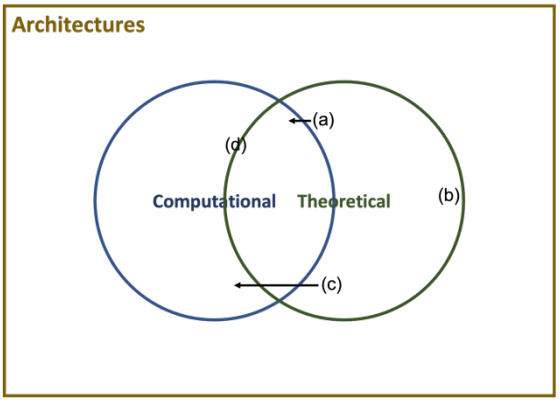

Figure 3: Some variations in the core architectural space: theoretical noncomputational architectures that can (a) easily or (b) not easily be converted into theoretical computational architectures, or (c) be converted into atheoretical computational architectures; and (d) computational architectures that embody implicit theories about their domains in the abstract.

In a different manner, a noncomputational theoretical architecture may be converted into an atheoretical computational architecture (Figure 3c), such as when the theory of evolution was converted into genetic programming (Koza, 1992) that, while inspired by the original theory, is intended to provide a useful tool rather than a model of the natural phenomena. Or, as discussed above, a computational

[21] I have attempted to extract from the literature in philosophy of science a clear distinction between theories and models but have instead been left with a muddle that includes such possibilities as degree of abstraction or approximation or locality or dynamics or accessibility. So, rather than attempting to draw a distinction here, these terms will be used interchangeably.

architecture that is primarily intended to provide practical value, such as a computer or A(G)I architecture, may also implicitly embody some amount of theory about its domain in the abstract (Figure 3d).

Both computational and theoretical architectures are amenable to being parts of layered stories. Consider first computational architectures. A computer architecture provides a fixed foundation that supports using programs to transform data. However, at a second level, programs may themselves be considered as computational architectures, particularly if they are fixed over some span of interest, with their associated data then providing the only source of variation. This data variation may in turn be very flexible—as for example with a cognitive architecture that supports a broad range of skills and knowledge within its data—or it may be severely limited. In the former case, of sufficiently flexible data, this process of implementing computational architectures within the variations that are enabled by other computational architectures can conceivably proceed to arbitrary levels. Sigma is designed in terms of two such layers—a *graphical* architecture and a *cognitive* architecture—with the former implementing the latter. In addition, further capabilities that are part of the architecture of the Common Model of Cognition are implemented with the aid of skills and knowledge that sit above Sigma's cognitive architecture.

Similarly, a theoretical architecture can be a theory itself or one part of a more elaborate theory that also includes appropriate choices among the variations the architecture enables. Kivunja (2018), for example, discusses the notion of a theoretical framework as the accepted wisdom from experts that serves as the background for a graduate student's own research. The former might perhaps also be considered a Kuhnian paradigm (Kuhn, 1962). This effectively partitions a student's contributed theory into a fixed, architectural framework or paradigm plus the student's own contributed variations. A Lakatosian research programme (Lakatos, 1978) also partitions in this manner. According to the Stanford Encyclopedia of Philosophy, "For Lakatos an individual theory within a research programme typically consists of two components: the (more or less) irrefutable hard core plus a set of auxiliary hypotheses."

A cognitive architecture can be considered as a theory of the fixed structures and processes found in a human mind, or of what is necessary and sufficient to yield an artificial mind capable of general intelligence. When combined with particular skills and knowledge, it can also serve as a more fleshed out model, for example, of human behavior in a particular task. While cognitive architectures are typically both computational and theoretical, only the more fleshed out models can actually be executed to yield behavioral data for comparison with human data, or applications that have value beyond their ability to model humans.

Computational architectures introduce an additional distinction, between the architecture and an implementation of it. This distinction can be relevant for all computational architectures, whether theoretical or not. For example, the earliest computers each had its own *instruction set*—a specification of the instructions the computer could execute—that was tied irremediably to the hardware it came with. However, it was not long before the architecture was separated from the implementation, with the architecturally defined instruction set specifying what was to be implemented, and multiple hardware implementations being developed to span sizes and generations of computers. The IBM System/360 provided a notable early example, where a single instruction set was implemented in a variety of ways that yielded different practical tradeoffs. A general language—Instruction Set Processor (ISP)—was even developed to enable specifying instruction sets across the field of computer architecture (Bell, and Newell, 1971).

For theoretical computational architectures, the question becomes how the theory, architecture and implementation are related. Versions of both the ACT-R (Anderson et al., 2004) and Soar cognitive architectures have been implemented in different programming languages—such as Lisp, C, Java, and Python—so there clearly can be a distinction between the architecture and the implementation. Nevertheless, there is typically no separate specification of the architectures beyond what is provided by their implementations, along with informal descriptions in the literature and even less formal ideas within their developers' minds. Although there were several attempts to specify Soar more formally (Milnes et

al., 1992; Cooper et al., 1996), with the first possibly having had an impact on the recoding of Soar from Lisp into C, this impact was at best transient.

So, the boundary between what is part of the architecture versus an implementation detail remains fuzzy for cognitive architectures (Cooper et al., 1996). This then also bleeds over into the question of what exactly the theory is in such cases. If the architecture is clearly separated from the implementation, the former would then typically embody the theory. However, when there is no clear distinction between architecture and implementation, the assumption is still that there must be a distinction between theory and implementation, given that the implementation typically needs to include more than just the theory in order to actually produce behavior. In such cases, only a subset of what is implemented should properly be part of the theory.

There are effectively two architectures in such situations: (1) the fixed code that is a mix of theoretical (core) and atheoretical (auxiliary) pieces and which combines with variations to yield behavior of interest, and (2) an abstract theoretical architecture that includes only the aspects of the implemented architecture that are intended to be part of the theory. In the first of these two architectures, the atheoretical component maintains an ambiguous state of being fixed implementation-wise with respect to the architecture's enabled variations but malleable in terms of being adjustable as necessary without affecting the core theory.

A classic slogan in AI, although it seems to have originated hundreds of years ago with Francis Bacon as *scientia potentia est*, is that *knowledge is power* (Feigenbaum, 1992).[22] We could also add that *applications are payoff*. In architectural terms, knowledge is a form of variation enabled by cognitive architectures, whereas applications are complete systems that result from combining appropriate architectures and variations. In analogy, we can now go a step further to say that *architecture is essence*.

Power for power's sake has rarely appealed to me, even if it implies the ability to leverage that power for desirable aims. Although I wouldn't mind being wealthy—one obvious form of applications payoff—I've also rarely found it attractive as a goal. Developing applications that matter to funders, and beyond to society as a whole, is a much worthier goal, and often essential for acquiring and sustaining research funding, but one I have typically found to be a duty rather than an interest. Instead, getting at the essence of complex phenomena, whether for understanding them or producing them, always has thrilled me.

Thus, by nature, I am an architecture person who has too often given less than their proper due to the variations or systems that are thus enabled (Maxims 19 and 25). At various points in my career, I have put effort into variations and systems, but they were often not the happiest of times, including one extended effort that likely contributed to a significant burnout in the late 1990s (see Section 3 for more on this). Yet, underemphasizing these two aspects has likely also held me back in significant ways, such as by losing the architectural insight that can result from a thorough study of the variations and applications it enables (or fails to enable),[23] reducing the availability of important forms of research funding, and denying the pathway of entrepreneurship that has been taken so productively by many of my colleagues. Such consequences are a major risk of focusing too exclusively on architectures.

Beyond simply being an architecture person, I have also always cared deeply about the simplicity, beauty, and elegance of the architectures I create, even when I did not consciously realize this. These are notions more typical of physics than CS/AI or cognitive science and are even further from what is the norm in the biological sciences, including neuroscience, where the notion of evolution as a tinkerer (François, 1977) can imply an expectation for an architecture that might informally be characterized in CS terms as "hacks upon hacks." Even in physics the idea can be controversial (Hossenfelder, 2018), but if you don't search for beauty before settling for less you will likely never find it. And such a search should be less

[22] Ed Feigenbaum was the main CS/AI person who recruited me to Stanford as faculty after my Ph.D., although I assume that the psychology part of my appointment owed a lot to Gordon Bower, who earlier had taught me mathematical psychology as an undergraduate and thereafter wrote me a recommendation letter for graduate school. My first attempt at a research project was with Gordon, based on the class, but it unfortunately failed to yield any insight.

[23] For example, one of the biggest misses in work on Sigma, despite having considered doing it for some time, has been the lack of any attempt at bringing it together with a major *knowledge graph*.

controversial when it is merely one path being followed within the overall community rather than the dominant culture that can make other paths appear less than respectable.

To a physicist, beauty may focus on the mathematical form of a theory. For me, the core question is whether it is possible to understand and/or produce a wide range of complex phenomena of interest via conceptually clean interactions among a small orthogonal set of fixed ideas or mechanisms plus appropriately limited variations (Maxim 19).[24]

In my explorations into the Sigma cognitive architecture, I've come to call this *functional elegance*. It is clearly related to *Occam's razor* but with a different emphasis. It also relates to Allen Newell's earlier maxim that one should listen to the architecture. An important part of this was exploring what the existing architecture implies about a new capability before designing a new mechanism from scratch for it (Maxims 16 and 21-22). In this way, it actively prioritizes mechanism reuse over mechanism addition, as was highlighted earlier in the discussion of deconstruction.[25] But functional elegance goes beyond just these incremental questions, of how an architecture can be extended to new capabilities, to concern the ultimate beauty of the architecture as a whole (Maxim 20), including its expression in terms of the most fundamental concepts or mechanisms that can be identified (Maxim 23).[26]

The extent to which I have achieved functional elegance across the various architectures I have developed is of course an independent question. The apparent lack of further avenues for such progress on the Soar cognitive architecture in the late 1990s was a second factor in the burnout just mentioned, and to my stepping away from not just Soar, but from cognitive architecture research as a whole for a decade (Maxim 17). I began work on Sigma in the late aughts because I thought I saw a path for going way beyond what could be done in this manner with Soar. Whether or not Sigma is ultimately the best way to approach understanding and building minds—for example, the appeal to beauty could be misleading in designing cognitive architectures and/or Sigma may not end up be the most beautiful approach to doing so—Sigma clearly has at least extended in fundamental ways the functionally elegant approach to cognitive architectures.

2.2 Exploration: Seeking Insights

Science conventionally plays off theory and experiment. Theory generates hypotheses and predictions based on what is known to date. Experiments test hypotheses and predictions, increasing what is known and leading to rejection or modification of theories. Proofs may also play a role when it is possible to formalize a theory to the extent that conjectures—a particular form of prediction—can be proven or disproven, although such activities may at times be considered more mathematics than science. Similarly, simulations may play a role when a theory can be made sufficiently computational[27] to enable implementing and executing it.

---

[24] More crassly, and somewhat less accurately, it can be thought of as how much bang for the buck you get for each part of the architecture via reuse and (re)combination. A recent article by an MIT physics Nobel laureate referred to this kind of thing generically as *productivity* or *exuberance* (Wilczek, 2015).

[25] This, perhaps surprisingly, also dovetails with the reuse of what already exists in the evolution of biological systems.

[26] There can be a slippery slope in general as to whether a framework should be considered an architecture versus a *toolkit*, that is, something that provides many options that can be leveraged as appropriate. A toolkit nominally provides fixity, yet the proliferation of options weakens this notion. For theoretical frameworks with many options, the variety of options is assumed necessary to match the scope of what is being modeled, and so there remains comfort in considering such as architectures. Whether a particular atheoretical framework is an architecture versus "just" a toolkit can, however, be contentious. When there are few options, "architecture" generally seems the appropriate label. When there are many, the result may be more a toolkit than an architecture. However, a functionally elegant approach that yields its options from interactions among a small set of common mechanisms yields what might be considered a hybrid, providing an architecture at one layer that engenders a toolkit at a higher layer (unless the higher layer itself has a theoretical stance).

[27] There can also be analog simulations that exploit a variety of physical phenomena to simulate other phenomena, but they are not of particular relevance here.

Exploration—that is, investigating unknown areas in search of new phenomena to understand—is often considered pre-scientific,[28] but the scientific enterprise would be terribly impoverished without it. Sometimes new phenomena are uncovered as serendipitous side effects of sanctioned scientific activities, such as an unexpected prediction from a theory or an unplanned outcome from an experiment. However, early natural scientists fanned out across the globe to deliberately seek out and make sense of novel plants, animals, and other natural phenomena. To this day, scientific disciplines such as paleontology, astronomy and particle physics intentionally explore the natural universe. The former seeks undiscovered caches of fossils, while the latter two build new instruments that enable sensing aspects of the universe not previously perceptible to us. There may be hypotheses and predictions that are intended from the beginning to be tested via such instruments, but part of the anticipation and excitement in any such enterprise must always be the possibility of uncovering what was not expected.

As indicated earlier, one of the things I have realized about myself over the span of my career is that I am attracted much more to thought provoking novelty than to rigorous methodology (Maxims 19 & 25). Perhaps this makes me inherently pre-scientific, or non-scientific, or even a bit of a Kuhnian revolutionary (Kuhn, 1962) in always longing for a paradigm shift in my own thinking, but I get little thrill from either formalization or precision. As mentioned in the introduction, my undergraduate degree was in mathematical sciences, and I have run a range of experiments over the decades, so it isn't that I don't understand these activities or can't be competent at them. But to me they are at most a means to an end rather than an end in themselves; and too often we seem to lose sight of the ends by overfocusing on such means, or metaphorically losing sight of the forest due to the trees. In combination with my natural inclinations, this has implied that although I deeply appreciate the role of formalizations and experiments in solidifying our understanding of the world, neither has ever been the center of my own endeavors.

The cognitive architectures I have developed over the years (Section 3) have partially been intended as expressions of, and means for testing, hypotheses. Often these hypotheses start out rather vague,[29] such as some notion of the potential benefits deriving from the combination of: rules and activation in XAPS (Section 3.1); rules, problem spaces and chunking in Soar (Section 3.2); and Soar-like cognitive architectures and probabilistic graphical models in Sigma (Section 3.4). Sometimes this develops into a somewhat more precise—and even at times grander—hypothesis, such as that chunking provides a general learning mechanism (Laird, Rosenbloom, and Newell, 1986a) or that Soar can support a *unified theory of cognition* (Newell, 1990). More often, a number of crisper small hypotheses are spun off, such as that chunking in Soar could be used not just to speed up performance but also to learn new things (Rosenbloom, Laird, and Newell, 1987), or that Sigma could straightforwardly be extended to incorporate neural networks (NNs) (Rosenbloom, Demski, and Ustun, 2017).

Still, most of the actual work is exploratory, using the architectures to guide the exploration of cognition and to interpret the results of these explorations; and searching the space of cognitive architectures. Controlled experiments provide only a small fraction of the criteria used to evaluate such work, to the point where at times they seem almost epiphonemal, in the sense of being afterthoughts performed more for publication purposes than as part of an original plan.[30] This is of course an exaggeration, as the experiments do have purpose and have been informative, but the bigger-picture criteria center on: (1) whether an architecture, or some part of it, actually functions as hoped; (2) how broadly the architecture models and/or produces the phenomena of interest; (3) how simple and elegant the architecture is; and (4) how much insight the architecture inspires and captures, whether into anticipated implications or wholly unanticipated ones.

---

[28] Modern descriptions can be found of "sanctioned" exploratory research—such as in Wikipedia—but it is restricted to specific methodologies to be used in preparation for a planned experiment and is thus considered much more of a niche activity.

[29] Sometime after writing this paragraph, I ran across an apt quote from Bertrand Russell—"I do not pretend to start with precise questions. I do not think you can start with anything precise. You have to achieve such precision as you can, as you go along." (Russell, 1918).

[30] I also sometimes wonder whether formalizations provided in many AI papers are similarly epiphenomenal.

Much of the evaluation activity around these four criteria centers around building new architectural components and exploring their interactions. It is as simple as that. Controlled experiments, in contrast, tend to focus on refining what can be said about criterion (1), in terms of how well the architecture models and/or produces phenomena of interest. Thus, such experiments clearly play a role, but not close to the dominant one.

Each of the other architectures I've worked on embodies some form of grand yet vague hypothesis, as I've always found it more satisfying to work on entire forests rather than on individual trees. The Common Model of Cognition (Section 4) is based on the hypothesis that there is useful commonality hiding behind the surface differences in many cognitive architectures. Dichotomic maps (Section 5) are based on the hypothesis that AI, cognitive science, and cognitive/A(G)I technologies and architectures can be understood in terms of the cross products among a small set of broadly applicable dichotomies. And the Relational Model is based on the hypothesis that computing as a great scientific domain can be understood in terms of a small set of relationships with itself and the three other such domains—the physical, life and social sciences. The same bigger-picture criteria apply here, but controlled experiments play an even smaller role. Instead, exploration is the dominant methodology.

To many of a more rigorous bent this likely seems like a fatal flaw (Maxim 11), and I have worried about this at various times over my career.[31] However, in understanding myself as an architectural explorer the point is to accept the resulting limitations and instead celebrate what can be yielded by such a methodology (Maxim 12). This doesn't rule out controlled experiments when doable and relevant, nor formalization—or semi-formalization, as found for example in the Relational Model—when it appears helpful and I can understand how to do it, but it shifts the focus from such activities to exploration and the bigger-picture criteria relevant to it.

Yet, it is also true that as a discipline matures, its architectures should as much as possible become more in the way of rigorous theories while the explorations themselves should morph increasingly into controlled experimentation. The relationship between architecture and exploration can be thought of as analogous to, and a precursor of, the relationship between theory and experiment. An architecture provides the framework that guides and summarizes exploration, while an exploration can help refine and flesh out the architecture. Early natural scientists, were, for example, accompanied in their explorations by simple architectures with accompanying variations, such as maps and taxonomic hierarchies. While these particular explorations largely focused on further populating the architectures with variations, such as new locations and species, they may also have helped to refine the architectures themselves.

My own focus, instead of being on the natural world, has been on the world of *information*, in terms of *computing* and its kissing cousin *thought*. Simon (1969) referred to these as *sciences of the artificial* that involve understanding what we create. But in my own attempts to understand the discipline of computing I ended up taking a more aggressive tack that acknowledges the existence of natural forms of computing (Denning, 2007) while arguing that all four great scientific domains—the physical, life, social, and computing sciences—inherently involve an inextricable intertwining of understanding and shaping, and thus both natural and artificial phenomena, with computing only being somewhat out ahead of the other three in terms of its ability to create instances of the phenomena it studies (Rosenbloom, 2012a).

Exploration was the name of the game in the early years of AI, when there were many more unexplored areas than researchers to look into them and it was more fruitful to move on to new topics once initial explorations of existing ones yielded sufficient low-hanging fruit than to work over each existing topic very carefully first. Now, however, exploration has fallen into relative disrepute as a method of investigating topics in AI—with a broader coverage of the space of topics presently under our belts, proofs and controlled experiments have become the coin of the realm (Maxim 11).

As discussed in (Rosenbloom, 2012a), the existence of stronger research methods—in terms of the level of veracity they can guarantee—within a discipline tends to reduce the acceptance of weaker ones (Maxim

[31] Driven by general community angst over this, there have been a number of workshops and papers over the years concerned with how to rigorously evaluate work on cognitive architectures, yet none of them have truly succeeded.

11). This is completely appropriate when both strong and weak methods apply to the same problem, but it tends to carry over even to those areas of the discipline in which the stronger methods don't (yet) apply (Maxim 12). Erecting new disciplinary boundaries that wall off the latter areas can sometimes protect them from thus being ignored due to no strong methods being available for them, but there is also then a concomitant risk of the walled off areas becoming insular and irrelevant.

One of the reasons I have particularly enjoyed attending conferences on AGI over recent years is that the spirit of exploration is still very strong in this relatively new community that has returned to AI's original grand challenge of building systems capable of general human-level intelligence (or beyond). Much of the work that goes on in that community strays from what is considered academically respectable, but my mind is often more engaged by the thought-provoking work that is presented at such fora—and the potential for new insights often seems higher—than at many traditional AI or cognitive science conferences. It is worth noting that by relabeling themselves as AGI, rather than thinking of themselves as part of AI, they have achieved some of the necessary separation mentioned in the previous paragraph, but also accompanied by the dangers inherent to such balkanization. I have dealt with this threat personally by spanning the divide; that is, by attending both AGI and more traditional academic conferences. I approach both with the goal of unearthing what insight might be available, while always trying to keep in mind how well the various possibilities are actually supported.

My ongoing dedication to exploration is another reason why funding has been more difficult during the latter part of my career (Maxim 12), while making publication more difficult as well. Finding some useful experiments to add to papers when considering publication helps with this latter issue to some extent, but in general my later papers have appeared more in niche venues rather than in leading ones. Even leading venues that eagerly published papers that were largely exploratory in the early days of AI no longer consider such work appropriate (Maxim 11).

3. Grand Challenge: Cognitive Architecture [Tier 1]

I have spent most of my career on the grand challenge of a unified understanding of thought in natural and artificial minds that enables modeling the former and building the latter (Maxims 13-15); that is, on cognitive architectures that span both A(G)I and cognitive science (Maxims 5, 7 & 10). We now refer to such architectures as *humanlike* (Laird, Lebiere, and Rosenbloom, 2017), as they span both human minds and artificial minds that are sufficiently similar to them. Similarity is a squishy concept here, but if you think of the space of all possible minds, humanlike minds should fall within a single compact region, or perhaps on a single hill if you consider hill climbing in the search for models of intelligent behavior. At times I have considered the even more ambitious grand challenge of understanding this whole space, which not only includes humanlike minds but also animal minds and all other possibilities, whether artificial, alien, or elsewhat. The work on dichotomic maps (Section 5) may be a start in this direction, as may be newer work on what I have taken to calling the *intelligence space* (Rosenbloom, 2025), but both only provide very early beginnings.

The original goal for AI was to build complete artificial minds, but most of the field has focused on only individual pieces over the years, with the recent piece being almost exclusively machine learning (ML).[32] In searching for a unified understanding of thought, the focus is metaphorically on the forest of the overall mind rather than the trees of the individual capabilities into which it might be partitioned; the trees are typically only attended to in terms of how they can be combined to yield a fully functioning forest. There are times when addressing the forest becomes fashionable (Maxim 18), but most of the time those of us who view it as a mission rather than a fad are considered eccentric at best. We are also considered methodologically suspect, both because it is harder to prove theorems or run experiments on the whole

[32] However, the recent emergence of *large language models* (e.g., Brown et al., 2020) that are trained over vast bodies of text on many subjects has begun to blur the distinction given the capabilities they can exhibit from experience with such bodies of text.

forest (Maxims 11-12) and because the focus on the forest yields neither as precise a picture of the individual trees nor narrow applications that are as rewarding.[33]

I have also long believed that there must be at least one path toward unified AI that follows the blueprint laid out by human minds, at some appropriate level of abstraction, so that pursuing both fields in tandem is not only possible but enables pooling insights from both toward a single melded result (Maxim 10). This is part of what has led to the focus on humanlike minds in the Common Model of Cognition (Section 4).

Working on a grand challenge that seeks to unify across both capabilities and disciplines requires delving into many subareas of each of the disciplines, yielding a broad background and perspective on them, while at the same time—and in multiple ways—making one feel like an outsider with respect to both disciplines. Much of what is necessary to succeed in such an effort can appear irrelevant to the majority who either don't pursue a grand challenge at all or who pursue a very different one. Seeking a unified understanding of thought in terms of cognitive architectures also implies a focus on the minority approach of *general* rather than *narrow* AI (and cognitive science), and the associated approach of trading off local optimization of individual capabilities in favor of global optimization for general humanlike intelligence. Moreover, approaching natural and artificial minds as two sides of the same coin leads to push back from much of both fields with respect to not sufficiently optimizing the results for their respective goals of matching human behavior versus producing effective computational behavior.

I began my graduate career at Carnegie Mellon University (CMU) in 1976, working on a project with Doug Lenat that eventually became Eurisko (Lenat, 1983), the predecessor to Cyc (Lenat, and Guha, 1989). Eurisko would become an architecture for automated exploration—or *discovery*—within the space of concepts and heuristics. Although little progress was made in the year I was with the project,[34] it did provide an impetus to read Pólya's (1945) classic work on heuristics. It was also my first involvement with something like a cognitive architecture. Eurisko was principally intended to be a computational architecture, although it could also be considered theoretical with respect to the abstract capability of exploration or discovery.

The next year I switched advisors to Allen Newell[35] and joined the Instructable Production System (IPS) project. The goal here was to develop an AI architecture based on *productions*—or *rules*—that combine context-sensitive conditions with actions to be executed in the specified contexts. Such systems can be very reactive, in terms of their ability to respond appropriately to whatever the current context is. They differ from traditional stimulus-response (S-R) systems primarily in including an internal *working memory* as part of the overall context.

The idea driving work on IPS was that you shouldn't have to program such a system, instead it should learn automatically how to perform tasks in a limited domain—in this case, job-shop scheduling—from instructions provided to it in English. Production systems seemed particularly appropriate for such a project because each production provides an independently learnable, local bit of knowledge. They turn out also to have roots in both theory of computation (Post, 1943) and the modeling of human cognition (Newell, and Simon, 1972).

My contribution to IPS, other than participating in group meetings, was to help program a very small part of one of the early OPS rule-based systems that provided the architecture for IPS. Rule-based systems thus became the first lineage of cognitive architectures—albeit relatively minimal ones—with which I had any hands-on involvement.

The goal of the project unfortunately turned out to be terribly overambitious for the time, and ultimately premature by a number of decades. As Allen put it in *Desires and Diversions*, the IPS project was a classic failure, being described via a prospective paper (Rychener, and Newell, 1978) and a retrospective paper (Rychener, 1983), but without any intervening papers containing actual results. The project did, however, develop rule systems, such as OPS5 and its underlying Rete algorithm (Forgy, 1982), that would be used

[33] Although this too may be changing with the advent of large language models.
[34] Lenat would later acknowledge of Eurisko that "For the first five years, nothing good came out of it".
[35] I would have started working with Allen immediately, but he was on sabbatical my first year at CMU.

as architectures for a number of expert systems during the 1980s boom. IPS would also articulate many of the questions concerning the relationship between knowledge and problem solving that would later be addressed with the development of the Soar cognitive architecture. OPS5 also played a central role in the implementation of Soar's long-term procedural memory. Much later, goals similar to those of the IPS project were revisited in the form of Interactive Task Learning (Gluck, and Laird, 2019). Large language models can to some extent also be seen as a reincarnation of this idea.

Following these two largely unproductive research years, I spent a year (1978-79) as a visiting graduate student in psychology at the University of California, San Diego (UCSD), working with Don Norman. Although the initial inspiration for this move was largely personal, it did enable me to truly learn about experimental and cognitive psychology—a long-term desire that previous attempts had failed to satisfy—and to embed myself in both the topic and the mindset. Serendipitously, it also turned out to be the year that people like Jay McClelland, David Rumelhart, and Geoff Hinton[36] began the Parallel Distributed Processing (PDP) journey at UCSD that reinvigorated work on neural networks in cognitive science and AI. It was also the year that the first annual conference of the Cognitive Science Society was held at UCSD.

3.1 XAPS

While at UCSD, inspired by what was going on in the psychology department there that year and attempting to fuse this with what I had learned the previous year from the IPS project, I began exploring the first of three eXperimental Activation-based Production System (XAPS) architectures; the first lineage of cognitive architectures in which I took a leadership role.

The core idea behind XAPS was that rules would not only generate new symbol structures conditionally based on existing ones, but also guide the flow of neural-like activation associated with these symbol structures. In the terms of that era, these were local connectionist models rather than the distributed vector models that have since dominated the field. Still, they provided an initial experience with exploring what happens when an architecture includes symbols and associated quantitative values, an idea that reappears in subsequent explorations into cognitive architectures, and which becomes fully embraced in both Sigma and the Common Model of Cognition.

One of the things I learned from this effort was that unconstrained activation processing is a morass within which it is sometimes possible to produce what appears to be magic, but which most often leads to an uncontrollable mess. Eliminating the morass requires an approach such as constraining the processing to match the semantics of probability theory, as in probabilistic graphical models (Koller, and Friedman, 2009), or taming it by leveraging a learning algorithm that molds it to align with experience, as in the later development of the backpropagation algorithm (Rumelhart, Hinton, and Williams, 1986) that is central to deep learning. Ad hoc programming can work wonders with symbolic systems but won't do much of use in such activation-based systems beyond enabling a few toy examples.

When I returned to CMU in 1979 from my year at UCSD, I continued with XAPS. John Laird—another student of Allen Newell's who had also participated in the IPS project—and I started to bat around ideas for how to create a General Cognitive Architecture (GCA).[37] But more pragmatically, by this point I had very little sense of a useful direction in which to head with XAPS. Allen helpfully reoriented me when he asked for assistance with a project concerned with the power law nature of human practice curves—simplistically, $T=BN^{-\alpha}$, where T is the time to perform a task, N is the number of times it has been performed, and α and B are constants. The full power law we worked with though was $T=A + B(N + E)^{-\alpha}$, where A is the asymptote of performance and E is the effective number of practice trials that occurred prior to the experiment being modeled.

[36] Geoff was a visitor at UCSD that year like me, although post-Ph.D. He later served on my thesis committee at CMU, before going on to win a Turing Award for co-leading the deep learning revolution and becoming known as a/the "Godfather of AI."
[37] John and I went on to be close collaborators and best of friends over the years. More on John can be found later, in Section 3.2.

The goal here was first to establish whether human practice did indeed show this regularity across all of the tasks for which we could either find or generate practice curves, and if so to then use this invariant from human behavior as inspiration for building an architectural learning mechanism that would inevitably yield such curves when applied to comparable tasks. This could be viewed as a classic instance of computational cognitive modeling, but it was also inherently architectural by focusing on invariants and fixed mechanisms. In this form, it ultimately led to a pragmatically useful learning mechanism for AI systems (Maxim 10).

The first step on this journey (Newell, and Rosenbloom, 1981) turned out to be rather impactful, with over 3000 citations according to Google Scholar, even though it was published as an invited book chapter—as were many of my early papers—rather than as a rigorously reviewed conference paper or journal article. Invited articles were a particularly amenable venue for exploratory work, although this paper did include many statistical analyses that examined the comparative accuracy of different functional forms—such as power laws, exponentials, and hyperbolics—as models for practice curves. It might have made a good journal article had it not been written in response to a specific invitation to Allen. However, journal articles were not so much the focus in AI in those days. Publication in books and conferences was much more the thing. There was always so much new to explore that taking the time to carefully produce and go through the comment-and-revision cycles typically required for journal articles was the exception rather than the rule. Journal articles were more the standard in cognitive psychology. But, without a focus on new experiments, this work might not have been a good candidate there, even with the impact it has since had in psychology and cognitive science.

As is clear from the number of citations, there was much follow up on this work. Much of it simply used the article to support the power law nature of practice curves, but others challenged whether practice curves were truly power law or whether our explanation for why they were power law was the correct one. Still, the citation numbers keep growing. I was invited to write a paper (Rosenbloom, 2006) for a special issue of *Tutorials in Quantitative Methods for Psychology*[38] in celebration of the 25th anniversary of this paper's "seminal" publication (Lacroix, and Cousineau, 2006).

I must admit, though, that just about all of the intellectual horsepower reflected in the original paper was Allen's, with me very much in a junior support role. Most of my effort involved developing and applying a statistical analysis package—which didn't exist in any standard form at the time—that would search the parameter spaces of power law and other curves to find the best fits to the data from the various experiments we were modeling.

The explanation for the power law of practice that we included in this paper was based on the existing notion of *chunking* in psychology, of combining multiple distinct memory elements into a new compound one (Miller, 1956). We adapted this to combine sequences of rule firings into new rules that in one step accomplished what originally required multiple steps. This was ultimately developed into a domain-independent architectural mechanism in Xaps3, an architecture without activation despite still having "a" in its name.[39] But the process of getting there was one of the most challenging and frustrating periods in my research career. It seemed like there was at least a full year during which I made essentially no progress, spending much time pacing my office thinking, or walking around neighboring Schenley Park doing the same.

Not coincidentally, this was when computer Othello (Aside 1) went from being a hobby to something I poured a lot of passion and energy into, becoming what I would later refer to as "what I did while unable to make progress on my thesis." A normal person would have abandoned the chunking problem and moved on to greener pastures long before, and a normal thesis advisor would have lost patience with a

[38] This journal has since been renamed *The Quantitative Methods in Psychology*.

[39] Activation was eliminated in moving to Xaps3 due to the experience with chunking in Xaps2 that led to formulating the *crypto-information constraint* that bans using information in architectural decision making that is not available to the knowledge and skills on top of the architecture (Laird, Rosenbloom, and Newell, 1986b). Activation in Xaps2 was crypto information in this sense, although it is not so in either neural networks or probabilistic graphical models, and thus also not in Sigma.

student who could not make progress on their topic. However, I am stubborn—or inflexible or obsessed or blind—enough that the thought of abandoning it frankly did not cross my mind; and Allen was understanding enough to tolerate this. This is a trait that has served me well at a variety of times, but not so well at others.

1. Computer Othello

During my time at UCSD, in addition to taking courses, working on XAPS, and dabbling in psychology experiments, I began developing *Iago*, a program for the game of *Othello*—or, generically, *Reversi*—a simplified variant of *Go*. This was one of the few times in my career I was inspired by an application rather than an architecture. I was frustrated with the terribly simplistic program on the PDP-11 minicomputer in the LNR (Lindsay, Norman, and Rumelhart) lab, even though I was a complete novice at the game at the time.

This effort started with the simple notion that it couldn't be hard to do much better than that program, but it ultimately resulted after I returned to CMU in a world-championship-level program† written in the SAIL programming language, and my first journal article (Rosenbloom, 1982).‡ I remember Allen Newell shaking my hand after this article was accepted without modification, a form of acceptance I never afterwards repeated. The CS faculty surprised me by passing me on what was called the "area qual" based on this effort. Prior to this, I was supposed to write an extended review article, a useful but not terribly exciting task, for the area qual.

It was also during this effort that I took some of the worst advice I ever received as a student—this advice was not from Allen—and listed two fellow students who worked with me on Iago in the acknowledgements rather than as co-authors. The resulting sense of guilt led to Maxim 24, of being as generous as can legitimately be justified when assigning co-authorship. To follow up, I would like to apologize finally and formally to my fellow students at the time, Bruce Ladendorf and Charles Leiserson.

† Iago won the Santa Cruz Open Machine Othello Tournament (Frey, 1981). It ran at CMU on a "large" DECSYSTEM-2060 with 512K 36-bit words of primary memory running at <2 MIPS while being controlled remotely via a terminal in Santa Cruz by my sister Kate, who lived nearby.

‡ This required creating a new Othello font for both a very early laser printer—the Xerox Dover—of which CMU had one of the few ever built, and a digital typesetter, to illustrate particular board positions and entire games.

The easy part of this problem at the time was figuring out how to build a chunking process that would compile experience over time into new rules that could replicate within a single step the consequences of that experience for similar later situations,[40] and thus reduce the time required in these later situations. The hard part was figuring out how to integrate this in with the rest of the architecture, in a domain-independent manner, so that new rules of the right sort would be learned automatically at the right times and be reused automatically, as appropriate, in later situations in lieu of the original processing. Making chunking impasse-driven—that is, when gaps became apparent in the knowledge needed for decision making, chunks would be created based on the problem solving that filled the gaps—ultimately yielded a simple, elegant solution to this problem (Rosenbloom, and Newell, 1986). It also yielded the first major architectural *aha moment* of my career. This was a process that Allen guided but whose result was truly torn from my own soul. It also became central to my Ph.D. thesis (Rosenbloom, 1983).[41]

The less traumatic portion of my thesis concerned a psychological phenomenon known as *stimulus-response (S-R) compatibility* (Fitts and Seeger, 1953). Simplistically, think of an elevator that can go up and down, and two buttons that can be used to call it to take you up or down. A compatible arrangement is one where the upper button calls the elevator to go up, and the lower button calls it to go down. An incompatible arrangement would reverse this ordering. Compatible arrangements enable faster

[40] Although this did get considerably more complicated in Soar, inspiring many revisions over the years of how chunking worked there. Figuring out how to do this for chunking in Sigma proved even more difficult, ultimately becoming a major stumbling block.

[41] The work toward my thesis was done on a Vax-11/780—the later big brother of the PDP-11—running Unix plus a Xerox Alto, a pivotal precursor of the Apple Macintosh, that Xerox PARC had donated to CMU.

performance and fewer errors. Although the phenomena around S-R compatibility were well known for over two decades, there was still no domain-independent theory of it at this point.

Noticing that across many experiments on this topic the differential time to perform the tasks could be predicted by the number of steps taken by simple programs hypothesized to underlie human performance on them,[42] I ended up developing an algorithmic theory of S-R compatibility that in a goal-oriented reformulation could be combined with chunking in Xaps3 to yield a cognitive architecture capable of modeling both practice and S-R compatibility (Rosenbloom, and Newell, 1988). Without additional work it could also then model the interactions between these two phenomena, when compatibility conditions were repeated over time.

These results on S-R compatibility influenced work in human-computer interaction by Bonnie John, one of Allen's later graduate students (John, Rosenbloom, and Newell, 1985; John, and Newell, 1990), but did not have a long-term impact on my own path.

3.2 Soar

Transitioning out of the XAPS architectural lineage began with merging a reworking of the domain-independent version of chunking from Xaps3 into an evolution of the Soar[43] problem solving architecture[44] that was separately being developed by John and Allen (Figure 4). They had developed a follow on to Allen's earlier work with Herb Simon and Cliff Shaw on the General Problem Solver (GPS) (Newell, Shaw, and Simon, 1959), and the IPS project, that could use knowledge in the form of rules to define and control search in problem spaces, developing what they called a *universal weak method* (Laird, and Newell, 1983) that could exhibit many different search algorithms depending on the knowledge that was available. They then enhanced this with a form of impasse-driven reflection, called *universal subgoaling* (Laird, 1984).

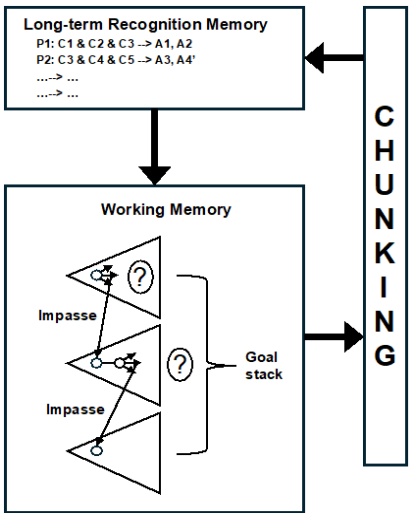

Figure 4: "Early" Soar, as a combination of a long-term recognition memory (rules); a short-term working memory in which a hierarchy of goals and problem spaces results from impasses; and a chunking mechanism that creates new rules based on traces of the rules that fire during impasses. (Figure redrawn with permission from Young, Ritter & Jones (1996)).)

Key to implementing chunking in Soar was that both Xaps3 and Soar were built around rules and impasses, although the form of the latter in Soar was much more general. Where the two architectures principally differed was that Xaps3 leveraged these capabilities to support learning by chunking, whereas Soar leveraged them to provide general problem solving. The combination of the two made a big splash in both AI and cognitive science, demonstrating a wide range of learning behaviors—not just simple practice learning—from the architecture plus two kinds of knowledge: domain-general knowledge about problem solving, and domain-specific knowledge about tasks to be performed.

John took the lead in implementing this combination, as the implementation of Soar was really his baby, but we both contributed conceptually to the result. The first journal article on Soar as a whole that included this learning mechanism (Laird, Newell, and Rosenbloom, 1987) has been cited over 4000 times, with another 900 citations for a concomitant paper that focused more exclusively on the generality of the

[42] These programs were written in a semi-formal language developed for this purpose called *Artless*.
[43] *Soar* was originally the acronym *SOAR*— State, Operator And Result—but it later morphed into a proper noun.
[44] Beyond the early technical article on Soar already cited, a "gentle" introduction to this version of Soar can be found in (Lehman, Laird, and Rosenbloom, 1998). There is also an early introductory video with me as the rather wooden presenter, plus more detailed recorded tutorials due to John. More up-to-date references, from after I left the project, can be found later in this section.

learning that resulted (Laird, Rosenbloom, and Newell, 1986a). These journal articles appeared, respectively, in the leading AI journal of the time and the first issue of the first journal dedicated to machine learning (ML). We even published our first book (Laird, Rosenbloom, and Newell, 1986b) by "stapling" together John's and my theses plus a short conference paper co-authored with Allen that amounted to the first publication on the combination (Laird, Rosenbloom, and Newell, 1984).

Soar became the second lineage of cognitive architectures in which I took a leadership role,[45] and remained my primary focus for a decade and a half. I was excited by multiple aspects of Soar. One was that it provided direct answers to a number of the questions concerning how to combine problem solving and knowledge (in the form of rules) that were raised earlier by the IPS project but which we had no clue how to answer at that time. The second was that it was conceptually a simple architecture that elegantly combined one representation for long-term knowledge (rules), one way to make decisions (based on symbolic preferences), one way to learn (chunking), and one way to reflect on its own behavior (impasse-driven subgoaling); yet it could exhibit a wide range of critical behaviors.

The work on Soar included a number of instances in which listening to the architecture led to new intelligent capabilities based on interactions among existing mechanisms rather than addition of mechanisms intended specifically for them (Maxims 16 and 20-22). John's and Allen's early work showed how many different problem-solving methods and search algorithms could arise from interactions among problem spaces, impasses, and knowledge. Adding chunking yielded many forms of learning (Steier et al., 1987) that might otherwise have required their own learning mechanisms, with data chunking (Rosenbloom, Laird, and Newell, 1987)—as mentioned in Section 1—perhaps the most unexpected such form. The same combination also yielded varieties of planning without explicit planning modules, with chunking storing plans piecemeal into rule memory (Laird, and Rosenbloom, 1990; Rosenbloom, Lee, and Unruh, 1990; Rosenbloom, Lee, and Unruh, 1993).

Overall, from this conjunction plus appropriate knowledge, we were able to demonstrate a wide range of humanlike intelligent behavior in Soar, including problem solving and planning, learning, knowledge-intensive behavior, natural language processing, and robotics. In 1987, Allen took advantage of an invitation to give the William James Lectures at Harvard to make a major statement about using cognitive architectures such as Soar as the basis for *Unified Theories of Cognition* (Newell, 1990).[46] By the early 1990s, under the continuing leadership of John, Allen and me (Figure 5), Soar formed the basis for two large books of collected papers, written by a community of researchers that grew to over one hundred individuals reaching out across disciplines, universities, and continents (Rosenbloom, Laird, and Newell, 1993).

Figure 5: John Laird, me, and Allen Newell (left to right) at the 6th Soar Workshop (1989) at the University of Michigan. In 2024, John hosted the 44th such workshop. In case you are puzzled by the numbering, workshops were held at an aggressive pace during the early years.

During this period, based on personal considerations that included a desire to be closer to family and friends and to be somewhere there was more sun, I moved from CMU to Stanford, where I was an Assistant

[45] The very first version of Soar was built around Xaps2, providing a second although more transient connection between the XAPS and Soar lineages, before Xaps2 was replaced by OPS5 as Soar's rule system.

[46] Mitch Waldrop, a Senior Writer at *Science* Magazine, followed up on this with two articles that explained Soar and the idea of a Unified Theory of Cognition to the broader scientific community (Waldrop 1988a,b). One unexpected result was a phone call I received one day from Francis Crick, co-discoverer of the structure of the DNA molecule, who had begun to think about consciousness and wanted to understand more about Soar.

Professor of Computer Science and Psychology for three years. This had seemed like an ideal position, back in California and at Stanford where I had thoroughly enjoyed my undergraduate years, and with a joint appointment in two top departments that would recognize and hopefully facilitate my desire to work on Soar across these two fields. Yet, despite the fact that I enjoyed being at Stanford and interacting with some very smart students while there, including three who completed a Ph.D. with me—Andy Golding (CS), Dirk Ruiz (Psychology), and Amy Unruh (CS)[47]—it proved to not be the best of fits. By my third-year renewal, it was clear that psychology was not interested in my continuing in that department, perhaps because my work did not seem closely enough tied to human cognition, with a complete lack of experiments on humans and only a rather abstract relationship to the modeling of human cognition. My fit within CS was somewhat better, but still somewhat out of paradigm for their AI group.

My position in CS at Stanford would have continued at least a few more years and I may have been able to create a longer-term niche for myself if I hadn't, for personal reasons, left in 1987 for the University of Southern California (USC) in Los Angeles. At USC, I started as a project leader at its Information Sciences Institute (ISI),[48] but then within a few years added the academic titles of Research Assistant Professor and then (tenured) Associate Professor in the Computer Science Department. I was later promoted to Full Professor. During my years at USC/ISI, I graduated four Ph.D. students working either on Soar or on a topic inspired by it—Bonghan Cho, Jihie Kim, Soowon Lee, and Benjamin Smith.[49]

Soar evolved in various ways during this time (Laird, and Rosenbloom, 1996), to a point where serious applications could be considered. When the opportunity—that is, the funding—arose to engage in a large effort focused on building humanlike automated forces for military simulation and training, we threw all of our energy into it. This started with modeling pilots of fixed-wing aircraft (Tambe et al., 1995). John Laird's group at the University of Michigan continued to explore this topic (Jones et al., 1999), while my group at ISI switched to rotary-wing aircraft (Tambe, Schwamb, and Rosenbloom, 1995)—that is, helicopters—and scaled up to groups of helicopters and their accompanying commanders (Hill et al., 1997).

This application also led to extending Soar in such directions as real-time behavior and multiagent systems. By using an instance of Soar to separately model each pilot of each aircraft, as well as particular commanders, and modeling typical communication among them, we were able to go beyond the previous state of the art of regimented Semi-Automated Forces (SAFs) to Intelligent Automated Forces (IFORs) and Command Forces (CFORs), with both individual and group behavior that was more autonomous and more humanlike. Watching these vehicles operate in this fashion did give the impression to key observers that they were controlled individually by humans rather than by AI systems. Ultimate success was measured by participation in several large Defense Advanced Research Projects Agency (DARPA) demonstrations in the late 1990s—Synthetic Theater of War (STOW) '97 and '98—and by John Laird spinning off SoarTech to continue with such efforts, along with other applications of Soar and related technologies.

A huge part of the success of the ISI component of this effort was the three researchers I was fortunate enough to hire to work with me: Milind Tambe went on to become a leader in multiagent systems and a Chaired Professor at Harvard, Randy Hill went on to become the Executive Director of USC's Institute for Creative Technologies (ICT), and Jon Gratch went on to become a world leader at ICT in virtual humans and affective (that is, emotional) computing.

---

[47] My time as faculty at CMU was too short for the two Ph.D. students who I worked with there to continue on with me as their advisor, but I did end up later co-advising two CMU CS Ph.D. students—Milind Tambe and Bob Doorenbos—who had started working with Allen before he became ill. Milind's name will come up again very shortly.

[48] The Intelligent Systems Division (ISD) at ISI grew to be one of the largest AI groups in the world prior to the recent ML boom. The ISD name deliberately omitted "AI" because at the time of its founding AI was entering one of its down cycles—known as *AI winters*—where the field was out of fashion. This has, however, recently been rectified with a change in the name to the Artificial Intelligence Division (AID).

[49] Many graduate students were trained overall within the Soar project over the years, moving on to successful academic or industrial careers, but most not directly continuing to work with Soar.

Allen Newell died during this period, in 1992, a major loss for us all, and for the AI and cognitive science communities more broadly.[50] He was a major figure in founding both fields, as well as a winner of the Turing Award, a National Medal of Science, and much more.[51] Allen had a huge influence on my own development as a scientist, while serving as a model of what a scientist should be that I was never quite able to live up to. He was genial and generous, deep and creative, passionate about his work and worked incredibly hard, good at both talking and listening, and cared about both people and organizations.

Within the Soar project, his leadership of the CMU component was replaced for a few years by Jill Fain Lehman, who had previously been leading the work on natural language processing in Soar (Lehman, Lewis, and Newell, 1991), before that site wound down. John Laird had himself moved on from CMU to Xerox PARC and then to a faculty position at the University of Michigan, where he remained working on Soar until his retirement in 2022. He has now spun up the Center for Integrated Cognition as a venue for continuing work on Soar and the Common Model of Cognition.

By the end of the military simulation effort, I personally was rather a basket case, with ambivalence about the application and a moribund basic research program. I found myself dulling my mind by inappropriately playing solitaire on my computer for hours on end and engaging in other forms of escapism. What followed was a sabbatical during which I attempted to model emotions in Soar, inspired by my own emotional issues plus a sense that there was something important to be done with cognitive architectures in this area. However, except fittingly for a bit of a model of frustration, I failed to make any substantive progress on this topic. By this point I was having trouble in general finding any direction in which I could make further progress on Soar while retaining the constraint that it remains simple and elegant, and felt like I was at a dead end without the internal resources to understand how to get out of it and to start something new. This was the burnout that has already been mentioned several times.

John escaped what I felt to be a dead end by dropping the constraint that there be at most one module for each major capability, freeing himself to continue developing Soar to this day (Laird, 2012; Laird, 2022), with new memories, learning mechanisms, and other modules such as for mental imagery (Figure 6). Included in this were new declarative learning and memory modules that essentially replaced data chunking as Soar's story. It turned out that although Soar could do declarative learning and memory without new modules, it required interrupting other activities to do so, rather than enabling them to proceed in parallel, as they needed to do. We have remained good friends over the years—we started our research journeys together as new graduate students at CMU in 1976 and spoke remotely at each other's retirement celebrations on the same day in 2022, although our official retirement dates were different. We grew somewhat apart during my interregnum, but he helped me transition back to work on cognitive architectures when I started work on Sigma, and we have been working closely together again on the Common Model of Cognition.

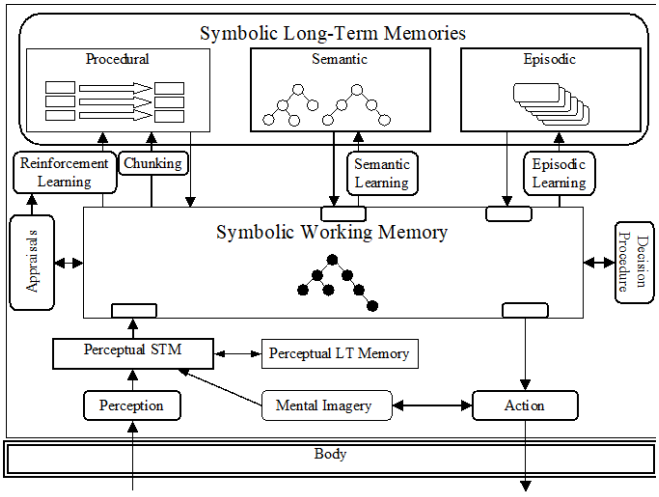

Figure 6: A later version of Soar with multiple memories, learning mechanisms, and more (From *The Soar Cognitive Architecture*: ©2012 MIT. All rights reserved.).

[50] I was in transit at Saint Louis's Lambert International Airport when I was paged and learned that Allen had died. I quickly rerouted to Pittsburgh to help deal with things there. I ended up taking on the job of calling many of his colleagues to let them know what had happened and stayed on to speak at his memorial service in the (now long gone) Skibo auditorium at CMU.

[51] He also played foundational roles in other areas of computer science, such as computer architecture (Bell, and Newell, 1971) and human computer interaction (Card, Moran, and Newell, 1983).

The Soar architecture has by now been under development for over four decades—and over the last twenty-five of those years led solely by John—making it one of the longest lasting, most thoroughly investigated, and well-known cognitive architectures.[52] The other architecture in this class is ACT-R (Anderson et al., 2004). Both are computational and theoretical architectures that play in both AI and cognitive science, although Soar is slanted more toward the computational/AI side and ACT-R more toward the theoretical/cognitive-science side. John's work on applying Soar also led him to play a leading role in developing the area of AI and (video) games, and later to do the same thing for interactive task learning (ITL)—a return to the original motivation for IPS, but now with technology and ideas that could make a difference. He also found time to explore the use of Soar on a variety of robots.

3.3 An Interregnum

Fortunately, when I needed it most Herb Schorr, ISI's Executive Director at the time, suggested I take a break from direct research to work on, and eventually lead, ISI's new directions efforts. This was less business development than trying to understand where the field was going and to help ISI move in such directions. Fortunately, it was one of the best possible cures for my burnout and the associated sense of being at a dead end from which there was no exit, as it involved an explicit mandate to think broadly in terms of both topics and collaborators.

In Allen Newell's terms, this turned into a decade-long diversion from the core desire of understanding and developing humanlike cognitive architectures; or, in my terms, a period when my research life was not ruled by architectural exploration. So, leaving the ISI Soar group in Randy Hill's and Jon Gratch's very capable hands, I set off on a decade-long exploration of the future of computing and its related disciplines.

My most significant accomplishment during this period was to take the lead in writing a complex multi-school proposal to the US Army that established USC's Institute for Creative Technologies (ICT), with an initial vision of applying academic computer science and entertainment expertise to enhancing military simulation and training. As Dick Lindheim—a Paramount VP and later ICT's first Executive Director—would say, it was about bringing story and character to military simulation to achieve *verisimilitude* (the appearance of being real); or, in other words, trying to create the technology for a real-life *Star Trek Holodeck*. My position in this effort was based on the earlier work on IFORs, plus time I spent afterward as a consultant to Paramount, working with Dick Lindheim and others on an ICT precursor called the *StoryDrive Engine*.

The plans for ICT were much more ambitious than our earlier work in military simulation in terms of how humanlike the simulated entities were to be—not just behaving appropriately in battlefield scenarios in which the AIs were entirely embedded in vehicles but serving as *virtual humans* that could engage in face-to-face interactions with each other and with real people. USC offered me the opportunity to take a leadership role at ICT, but I declined partly because I wasn't interested in leaving ISI at that point—I had just begun work on new directions— partly due to not yet feeling ready to take on such a role,[53] and partly because my earlier work on military simulation and training was one of the factors in the burnout that I was just working my way out of. However, even without me, the ex-ISI researchers who moved to ICT—including Jon Gratch, Randy Hill, Stacy Marsella, Jeff Rickel, and Bill Swartout—still chose Soar as the basis for building their initial virtual humans (Swartout et al., 2006).

---

[52] Soar is mentioned in the introduction to the leading AI textbook (Russell, and Norvig, 2010)—"The work of Allen Newell, John Laird, and Paul Rosenbloom on SOAR (Newell, 1990; Laird et al., 1987) is the best-known example of a complete agent architecture."

[53] One of the things I've learned about myself over the years is that various aspects have matured at different rates (Maxim 26). It wasn't until my fifties and sixties that I felt confident in my ability to lead anything beyond my own research group. Before then I was hobbled by my perceived shortfalls with respect to leaders I admired, and by my awkwardness in dealing with people. As with the ICT experience, I thus typically shunned opportunities to lead, more commonly and quite willingly instead taking on a background role as a deputy of some sort.

Other topics I explored during this period included automated building construction; responding to the unexpected, in response to 9/11 (Arens, and Rosenbloom, 2002); virtual teams composed of robots, agents, and people—aka RAP teams (Scerri et al., 2003); technology and the arts (including helping to recruit to ISI a group from the Getty Information Institute when it was shut down); translational medicine (helping with the informatics side of the proposal that led to the Southern California Clinical and Translational Science Institute); and high-performance computing (where I played a role in setting up USC's High Performance Computing Center). A range of collaborations were also explored with other engineering disciplines (for example, biomedical, electrical, civil, astronautical, and industrial and systems), the sciences (for example, geosciences and marine biology), medicine (for example, biomedical imaging) and the arts.

Not surprisingly, some of the efforts I attempted to initiate during this decade never bore fruit, at least in the form intended. The most memorable was the beginnings of a proposal to the Department of Homeland Security for a center following up on the groundwork that Yigal Arens[54] and I had earlier laid on responding to the unexpected. The core idea here was to build integrated models of cities, including their infrastructure, buildings, and populations, for use in understanding threats and planning to avert or respond to them.[55] The idea aligned closely with the whole approach of architectural exploration, as well as with the integrated simulation approach that underlay both my 1990s work on military simulation and ICT's initial vision. USC did end up winning this solicitation, leading to the USC Center for Risk and Economic Analysis of Terrorism Events (CREATE), but under different leadership and with a different approach.

I spent a number of years as an Associate Director at ISI while leading new directions activities and progressively getting more involved in the overall administration of ISI. When Ron Ohlander retired as Deputy Director, I set my sights on replacing him, with the thought of eventually succeeding Herb Schorr as Executive Director,[56] should he at some point retire. I did become Deputy Director, but even with the many years I had worked with/for Herb on new directions and as an Associate Director, we quickly learned that he and I had wildly divergent views at this point about both the nature of the position and the future of ISI, so I ended up leaving ISI rather suddenly, before completing a full year in this position.

For me personally, what followed once I left ISI was a year full time in the CS Department, focusing largely on teaching introductory AI courses. I was feeling rather beaten down in general at this point and was no longer sure what I might contribute. With respect to research in particular, it had been a decade since I had led a project of my own—I was explicitly expected to avoid leading any projects of my own during this period—and during this recovery year away from ISI the most I could do was just start to think about a few new directions I might personally consider pursuing.

One of these new directions was to follow up on some sketchy, back-of-the-envelope-style explorations I pursued while still at ISI, trying to understand whether there was any fixed coherence—that is, an architecture of a sort—behind the wide range of largely interdisciplinary efforts I had been involved with while leading new directions activities. As part of this follow up, I taught a small graduate seminar on the topic during my year of full-time teaching. The other new direction was based on an idea that also came up during the interregnum, of merging the insights from symbolic cognitive architectures like Soar with what had been learned over the years about probabilistic reasoning, in the form of probabilistic graphical models (Koller, and Friedman, 2009) and statistical relational learning (Getoor, and Taskar, 2007), a particularly hot topic at the time.

---

[54] Yigal was the Director of ISI's Intelligent Systems Division at the time while also leading a big push in *digital government*, concerned with leveraging digital technologies broadly across government agencies and activities.

[55] This general approach now often goes by the name of *digital twins*.

[56] Considering trying for the position of Executive Director of ISI was a big step in maturing to the point where I thought that even with my various weaknesses, whether relative or absolute, I could still bring enough to the table to consider contributing significantly via a leadership role. Although nothing came of this, having gone through the process helped me agree some years later to go up for election as President of USC's Faculty.

With these two largely undeveloped ideas in hand, but no research funding, it proved ideal to spend the following year on sabbatical. After some informal discussions with Randy Hill, who was by now ICT's Executive Director, and Bill Swartout, who was the Director of ISI's Intelligent Systems Division—and one of the people who had originally attracted me to ISI—before becoming ICT's Director of Technology, about my plans, they offered to host my sabbatical at ICT and to supplement those funds with some unrestricted research funding. This proved to be by far the best sabbatical of my career, enabling me to initiate two new large-scale efforts.

The first idea developed into the Relational Model of Computing, and the notion that computing was a great scientific domain (Section 6). The second idea turned into the Sigma cognitive architecture (Section 3.4). This second idea also led to a long-term affiliation with ICT and one of the most productive decades of my career, supported largely by the all-too-rare boon of relatively stable basic research funding—from the US Army through ICT's core contract—that enabled exploring essentially whatever directions that I felt Sigma needed to go. So, although I did not move immediately to ICT when it was spun up, I did ultimately reap some of the benefits of my earlier work on the proposal when I moved there a decade later.

3.4 Sigma

For a number of years, Sigma went unnamed, as it wasn't clear whether the ideas being explored would yield a broad toolkit for building probabilistic cognitive architectures or an actual probabilistic cognitive architecture. It eventually was named[57] when it became sufficiently clear that the constraints implied by combining what was learned from Soar-like architectures and probabilistic reasoning led to a single lineage of architectures rather than a broad space of possibilities (Rosenbloom, Demski, and Ustun, 2016a).

Over the years, a set of four desiderata crystalized around this lineage: (1) *grand unification*, which goes beyond the focus on high-level, central, symbolic thought processes typical of traditional cognitive architectures such as Soar to include low-level, peripheral, numeric and perceptual-motor processes; (2) *generic cognition*, which spans both modeling natural human thought processes and creating artificial humanlike thought processes;[58] (3) *functional elegance*, which looks to generate a broad range of capabilities and behaviors from the interactions among a small set of general mechanisms; and (4) *sufficient efficiency*, which constrains systems to run fast enough for relevant modeling and applications purposes.

Some of these desiderata were inherited from Soar, whether they were explicitly stated there or merely implicit. Generic cognition and sufficient efficiency are both of this ilk. As previously mentioned, functional elegance is my personal take on Allen Newell's earlier maxim of listening to the architecture (Maxims 16 and 20-22). Grand unification represents the largest shift, going beyond Soar's goal of unifying high-level cognitive functionality to include the low-level phenomena that it couldn't handle without separate modules but that seemed within reach in a functionally elegant manner via Sigma.

The early days of exploring Sigma felt much like the early days of exploring Soar, in that so many problems that previously appeared to have no appropriate resolutions suddenly became clear, with simple and elegant solutions. This is not to underplay the amount of conceptual and implementational work that had to be done over a number of years to bring this about, including coming up to speed on probabilistic graphical models[59]—something I continued with over the years but never completely mastered—and

[57] The name *Sigma (Σ)* was inspired by a self-referential acronym for something like "*S*igma is an *I*ntegrated (and/or *I*ntelligent) *G*raphical *M*odel *A*rchitecture," but based on the Soar experience it was considered a proper noun rather than an acronym from the start. It was also intended to connote a *significant change* from what had come before plus Sigma's *integrative* aspect and the notion of *addition* that is central to the sum-product algorithm (Kschischang, Frey, and Loeliger, 2001) underlying Sigma's graphical models. Kim LeMasters, ICT's Creative Director at the time, helped choose the name based on a set of raw possibilities I had generated for consideration.

[58] In retrospect, based on experience with the Common Model (Section 4), this may have been better named *humanlike cognition*.

[59] My biggest breakthrough in understanding how to apply such models to building a cognitive architecture came from reading a paper on factor graphs (Kschischang, Frey, and Loeliger, 2001), a very general formalism whose breadth of applicability was made readily apparent in that paper. This was a true *aha* moment for me, although driven by the work of others—in this case from electrical engineering—rather than from my own explorations.

getting back into programming after a decade away from it.[60] But it was a very exciting time in which many of the previous barriers to progress—or at least the ones in my mind—came tumbling down and breathtaking vistas opened up.

In a significant sense, Sigma grew as a variant of the early flavors of Soar that had only a single long-term memory, albeit with Soar's rule-based memory replaced with an extension of statistical-relational factor graphs (Rosenbloom, Demski, and Ustun, 2016b); only a single learning mechanism, albeit with Soar's chunking mechanism replaced by gradient descent learning (Rosenbloom et al., 2013); and decision and reflection capabilities much like Soar's, albeit with numeric preferences rather than the symbolic ones with which Soar had begun. Sigma's single long-term memory turns out to be general enough that it can still support the kind of rule-based procedural memory that Soar started with, but it can in parallel also support the two kinds of long-term declarative memories later added to Soar—for semantic and episodic knowledge—plus probabilistic reasoning, neural networks, and forms of perception and motor control (Figure 7). As mentioned in Section 2.1, this all occurs over two architectural layers—one graphical and one cognitive—plus the use of particular skills and knowledge (Figure 8).

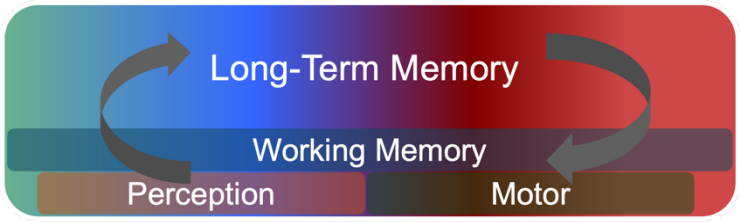

Figure 7: The Sigma cognitive architecture, emphasizing how key memory and processing components arise as variations on the underlying structure of extended factor graphs (Adapted from Rosenbloom, Demski & Ustun (2016a), p. 14, under Creative Commons Attribution-Noncommercial-Noderivatives 4.0 International license.).

Sigma's single learning mechanism can support various forms of symbolic, probabilistic, and neural learning (Rosenbloom et al., 2013; Rosenbloom, 2012b; Rosenbloom, 2014; Rosenbloom, Demski, and Ustun, 2017). What it does not yet support, however, is acquisition of new rules from experience, such as is provided by chunking in Soar.

As mentioned earlier, numerous attempts over the years to create a chunking mechanism in Sigma, whether a narrow version that only generates rules or a more general version that can produce all of the kinds of knowledge structures that Sigma encodes, have failed to pan out. This is another case of my extended stubbornness, but not one that has yet proven fruitful.

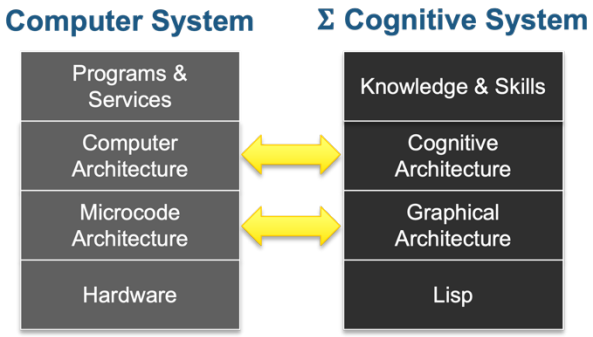

Figure 8: Sigma's layers, in analogy to those in standard computer systems (Adapted from Rosenbloom, Demski & Ustun (2016a), p. 16, under Creative Commons Attribution-Noncommercial-Noderivatives 4.0 International license, with two arrows removed).

The development of Sigma was pursued by a much smaller group of researchers, featuring in addition to me, my deputy and colleague Volkan Ustun, several Ph.D. students and programmers who made significant contributions—particularly Abram Demski, Himanshu Joshi, Sarah Kenney and Dinesh Rajpurohit—a number of helpful collaborators, and summer interns from a variety of universities and differing educational levels. But together, leveraging combinations of Sigma's existing mechanisms plus

[60] As a short cut, and to retain some level of comfort while changing so many other things, I went back to the use of Lisp here rather than investigating any of the newer languages that might have been more appropriate by this time. I always enjoyed Lisp because it naturally captured symbolic processing and using it was incremental and interactive. I learned Lisp as an undergraduate, from a course taught by John McCarthy, one of the four "fathers" of (symbolic) AI—along with Allen Newell, Marvin Minsky, and Herb Simon—who invented the language and had a surprisingly formal way of teaching it as a programming language, perhaps due to him also being the one to introduce logic into AI for automated reasoning. John later became a colleague when I joined Stanford's faculty.

limited extensions to the architecture and appropriate knowledge variations enabled by the architecture, the traditional symbolic capabilities of the early Soar versions have been extended without additional modules to include such capabilities as mental imagery (Rosenbloom, 2011a; Rosenbloom, 2012c), simultaneous localization and mapping (SLAM) (Chen et al., 2011), object and speech recognition (Joshi, Rosenbloom, and Ustun, 2014; Joshi, Rosenbloom, and Ustun, 2016), distributed vector and holographic representations (Ustun et al., 2014), and neural networks (Rosenbloom, Demski, and Ustun, 2016b) and deep learning (Rosenbloom, Demski, and Ustun, 2017).

Sigma has also been extended in recent years to incorporate aspects of Theory of Mind (Pynadath et al., 2013; Pynadath, Rosenbloom, and Marsella, 2014), attention and emotion (Rosenbloom, Gratch, and Ustun, 2015)—returning to where I last failed with Soar—and motivation (Rosenbloom, and Ustun, 2019). There was also to be an exploration of personality, but it was put aside when I became ill.

By this time, we were striving for *humanlike autonomous social cognitive (HASC) systems* (Ustun et al., 2018) able to:

- **Perceive**: Observing the environment and others in it
- **Reason**: Drawing conclusions from perception and knowledge
- **Act**: Choosing and executing appropriate actions
- **Interact**: Communicating naturally (verbally & nonverbally) with others
- **Use Theory of Mind**: Modeling one's own mind and that of others
- **Be Affective**: Understanding and exhibiting emotional behavior
- **Be Motivated**: Having desires and the ability to choose among them
- **Exhibit Personality**: Maintaining a distinct yet consistent character
- **Learn**: Adapting one's own behavior through experience

HASC systems were intended to support applications requiring virtual humans and other humanlike synthetic characters (Ustun, and Rosenbloom, 2015; Ustun et al., 2015; Ustun et al., 2018)—Sigma's most promising area for application—but this work later evolved under Volkan's leadership to use other approaches for learning basic character and team behavior, such as multiagent reinforcement learning, with the hope that Sigma's richness may still prove useful later (Ustun et al., 2021).

Whereas the implementation of Soar was very much John's baby in the early days, Sigma was mostly mine. That has been changing, however. Beyond limited early contributions by others in the group, there has been recent work to extend Sigma in languages beyond LISP and to take advantage of parallelism and special-purpose chips. Sarah Kenny, for example, explored leveraging parallelism in C for Sigma's inner loop. Himanshu Joshi has been exploring a fast reimplementation of a stripped-down variant of Sigma in C++. Jincheng Zhou has explored a new formulation of Sigma in Python for continuous data that may also ultimately support a version of chunking (Zhou, and Ustun, 2021).

Looking back, I started exploring cognitive architectural lineages based on symbolic rules and neural-like activation in the late 1970s and ended up with an architecture that still features such a combination. The intervening years involved architectures exhibiting increases in the generality and scope of the mechanisms embodied, the varieties of skills and knowledge that could be expressed and processed, and the kinds of applications that could be pursued. From this personal perspective on the evolution of cognitive architectures, a form of history of AI can be found in Aside 2.

My history with Sigma likely ends here, although not with a dead end as it did for me with Soar, as I still see significant untapped potential within its functionally elegant framework, including the possibility of serious applications. Although any such effort will need to be in the hands of others, my history with cognitive architecture research will hopefully continue on, but most likely in the more abstract forms found, for example, in the Common Model of Cognition (Section 4) and dichotomic maps (Section 5).

2. A Personal Cognitive Architectural Perspective on the History of AI

When I started as a graduate student at CMU in the mid-1970s, AI was still deeply in its foundational symbolic era, centered on such topics as AI languages, search, and knowledge representation and reasoning (KR&R). Cognitive architectures got their start in this era, as AI languages that not only integrated together capabilities such as search and KR&R but also added a theoretical aspect concerning how they combine to yield minds. The topic of learning was just beginning to have an impact late in this era, including in cognitive architectures.

The knowledge-intensive era, spearheaded by Ed Feigenbaum and others, built on the symbolic era, but centered on the knowledge required for effective task performance. The slogan was *knowledge is power*, which was backed up by the first successful wave of AI applications during the 1980s expert-systems boom. Work on cognitive architectures was primarily considered interesting during this era to the extent it did useful things with knowledge (such as in Rosenbloom et al., 1985).† This era ebbed as the limits of hand crafted, symbolic knowledge became apparent, leading to an *AI winter* during which interest in, and funding for, AI declined.

The next big era centered on probabilistic reasoning, which enabled shades of knowledge and the explicit representation of uncertainty in more principled ways. It also enabled statistical learning to automate some of the manual knowledge acquisition of the previous era. Judea Pearl was the pioneer of this approach (Pearl, 1988). Cognitive architectures were largely considered irrelevant during this era, as all that was assumed to be necessary were the individual learning and inference algorithms needed to convert probabilistic information into applications. As with the knowledge-intensive era, there were a number of successful applications. Although there was work on statistical learning in architectures (e.g., Langley et al., 1989), architectural researchers were slow to pick up on the deeper insights from this era. It was not until they were merged with the earlier ones from the symbolic era, in what was initially known as *statistical relational learning* (Getoor, and Taskar, 2007), could attempts such as Sigma appear in which cognitive architectures were grounded in these ideas.

The current era centers on artificial neural networks (ANNs) and machine learning (ML). Early work (McCulloch, and Pitts, 1943) predates AI's symbolic era, which is dated from the Dartmouth Conference in 1956. However, overinterpretations of a result (Minsky, and Papert, 1969) that proved limitations on what could be done with the one-layer ANNs that could be learned at that point (Rosenblatt, 1958) effectively suppressed the approach for a decade. The modern ANN explosion was enabled by the backpropagation algorithm for learning multi-layer ANNs (Rumelhart, Hinton, and Williams, 1986), but didn't truly take off until massive data sets, along with the massive computing necessary to train large ANNs on such data, became available.

For much of the ANN community, not only have architectures been irrelevant, but so also has been symbol processing, except as emergent from ANNs. In the other direction, there have been a number of attempts to incorporate aspects of ANNs into cognitive architectures, from my own primitive ones with XAPS and Soar (Cho, Rosenbloom, and Dolan, 1991) to the pervasive use of activation in architectures like ACT-R (Anderson et al., 2004) and explicitly hybrid architectures such as Clarion (Sun, 2016) and SAL (Jilk et al., 2008). Sigma has provided a new approach based on a tighter coupling between these capabilities. It is, however, possible to view recent work on ANNs as evidence that an increasing subcommunity is seeing the ultimate limitations of "pure" ANNs and thus the need to combine these ideas with more structured (Zhou et al., 2019), symbolic (Silver et al., 2018), and architectural (Eliasmith, 2013; LeCun, 2022) aspects.

The recent ascendance of *generative AI* (e.g., Brown et al., 2020), with their unique combination of awe-inspiring generality—yielding an era that combines *knowledge is power* with what can be termed *learning is generality*—and dangerous incompleteness, looks likely to move us into a new period of interest in cognitive architectures (e.g., Park et al., 2023). There is already much discussion as to the extent to which such systems yield AGI, and of where they fall short. Over the history of AI, shifts from the dominant focus on *narrow* intelligence that concerns individual capabilities to more *general* intelligence typically occurs when enough people get sufficiently frustrated with the limitations of the former to consider broader integrations.

† In this work, Ed particularly appreciated how chunking in Soar could automatically compile a shallow rule-based system from experience in running a deeper model—also in Soar—of how task problems could be solved.

4. Common Model of Cognition: Searching for Consensus [Tier 2]

The previous section focused on three related lineages of cognitive architectures. Soar is one of the two most iconic architectural lineages overall—the other being ACT-R—having been under development for decades and having a wide range of applications. XAPS was more of a flash in the pan, although embodying an early attempt at combining symbolic rules with numeric activations and other aspects such as chunking that had lasting impact. Sigma has been an attempt to retain what still seems right to me about Soar, and to a lesser extent ACT-R, but to replace its core memory and reasoning components with a more modern and general approach based on an extended form of graphical models.

In the research community more broadly, many different architectures have been investigated to one extent or another. A recent paper surveyed 84 different cognitive architectures, 49 of which are still under active development (Kotseruba, and Tsotsos, 2020). This paper also mentions the existence of 111 additional architectures, for a total of 195. Some of these bear a similarity to the lineages already mentioned, but others differ in striking ways, particularly when considering AI and AGI architectures that make no attempt to align with human cognition. Understanding, evaluating, and comparing these architectures can be challenging, as each is typically a complex software system that further requires variations in the form of skills and knowledge to be added into the mix before it can actually behave.

The Common Model of Cognition is an attempt to abstract away from individual cognitive architectures to find a community consensus that may be incomplete—it is not expected to be an implemented architecture itself—but that captures what is agreed upon in a common language (Laird, Lebiere, and Rosenbloom, 2017; Rosenbloom, Lebiere, and Laird, 2022). The Common Model focuses on humanlike cognitive architectures; that is, architectures intended either as models of human cognition or A(G)I architectures that are similar enough to models of human cognition as to be considered in the same class (Figure 9). This acknowledges that there may be multiple distinct ways in general of achieving human-level, or even higher levels of, cognition—making the existence of a Common Model over all such approaches questionable—while explicitly assuming that there must be substantive commonality for humanlike minds.

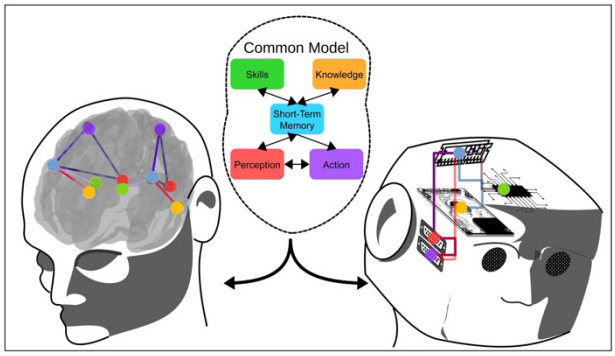

Figure 9: The Common Model of Cognition as an abstract consensus over natural human minds and artificial humanlike minds. (Drawn by Andrea Stocco. From Rosenbloom, Lebiere & Laird (2022), under Creative Commons Attribution-NoDerivatives 4.0 International license.

If successful, the Common Model should provide an evolving statement of where the field of (humanlike) cognitive architectures considers itself to be, while improving communication among researchers within this field and with others outside of the field. It should also help in comparing and contrasting distinct architectures by providing an interlingua to which they can all be related. It could also help guide the development of new architectures and of applications that either require models of human minds or working instances of artificial minds.

Work on the Common Model has involved exploration of both scientific and social architectures. On the scientific side, the Common Model itself is a theoretical architecture, concerning the nature of humanlike minds. It can also be considered a meta-architecture that is about the space of humanlike architectures. In being unimplemented, it is not directly computational, but it falls within the class characterized in Figure 3a, as a theoretical architecture that is not far from being computational.

As shown in Figure 9, the Common Model includes a central, short-term working memory, long-term procedural (skills) and declarative (knowledge) memories, and perception and action modules. All communication goes through working memory except for the direct connection between perception and action. Accompanying this overall structure are sixteen assumptions that underlie it, such as that there is: parallelism both within and across the modules; sequential behavior that arises from a cognitive cycle for

action selection that operates at ~50 msec in humans; a hybrid of symbolic data and quantitative metadata (that is, data about data); and learning that occurs incrementally as a side effect of performance.

The initial (meta-)theoretical work on the Common Model focused on generalizing across ACT-R, Soar, and Sigma, while taking other architectures and results from AI and cognitive science less directly into account. All three of these architectures have a presence in both AI and cognitive science—emphasizing, respectively, their computational and theoretical aspects—although with ACT-R more focused on the latter and Soar and Sigma more on the former. At this point, all three can all be considered as variant implementations of the Common Model. Each largely agrees with the Common Model's tenets but fails to include parts of it and may either include parts not (yet?) in it or that differ from it, and thus from the current consensus. The analyses show how ACT-R and Soar have converged over the decades, at least at this level of abstraction, with Sigma both more incomplete—not surprising given its relative newness—and more in disagreement with the consensus, particularly in terms of what it considers part of the architecture versus resulting from it plus variations.

By now, more than a dozen cognitive architectures have been mapped onto the Common Model or been described as Common Model architectures, with many suggestions having been made as to how it might be improved. The Common Model has also been used as a guide in several applications. Most excitingly, in an experiment I didn't have to perform personally, the Common Model has been tested as a model of high-level brain architecture. A team led by Andrea Stocco (University of Washington) has mapped the Common Model's components onto functional circuits in the brain and used brain scans to compare it with more standard hub-and-spoke and hierarchical proposals from neuroscience (Stocco et al., 2021). The result has been an overwhelming statistical dominance for the Common Model over these other approaches.

On the social side, the overall effort began with a final presentation that Christian Lebiere gave to sum up a symposium on *Integrated Cognition* that he and I co-organized in 2013.[61] In this presentation, he introduced the notion of a *Standard Model of the Mind* in analogy to the Standard Model of (Particle) Physics and mentioned a few aspects he believed from what he had seen at the symposium, and likely from his general background as well, should be in it. It frankly shocked us—in an extremely positive manner—that the participants in that symposium, who came from AI, AGI, cognitive science, robotics, and neural perspectives—and who worked on a wide range of architectures themselves—agreed with what he presented. This talk and the resulting reaction jointly inspired the broader effort that has turned into the Common Model of Cognition.

The first step was agreeing to an interuniversity collaboration among Christian (CMU), John Laird (University of Michigan), and me (USC), and thus implicitly among the three architectures we currently represent—ACT-R, Soar, and Sigma, respectively. While John was not a co-organizer of the 2013 symposium, he had been a participant and was soon included as a full partner in the overall enterprise. The social architecture here could remain fairly simple, but perhaps for that reason proved to be quite productive, resulting in a joint paper on the Standard Model that again surprised us in terms of how much could be agreed upon, at least among the three of us and our architectures (Laird, Lebiere, and Rosenbloom, 2017).

In one sense it should not be that surprising that some kind of consensus could be achieved over these researchers and architectures. We all received our doctorates in computer science at CMU, two (John and me) advised in parallel by Allen Newell, and the third (Christian) advised later by John Anderson. The three cognitive architectures are thus all part of a broad CMU architectural tradition. Soar, of course, owes a direct debt to Allen Newell's earlier thinking on cognitive architectures and their precursors (Newell, and Simon, 1956; Newell, Shaw, and Simon, 1959; Newell, and Simon, 1972; Newell, 1973) and to his personal involvement with its early development. Sigma is in many ways a descendant of Soar, although it does bring in a radically different technology at its core. ACT-R is descended from a line of earlier architectures,

[61] This symposium arose from an interaction Christian and I had at an earlier conference on cognitive modeling during which we lamented how, although there were many conferences that touched on particular aspects of cognitive architectures, there was not one that accepted work on all of the relevant threads and perspectives.

such as HAM (Anderson, and Bower, 1973),[62] ACT (Anderson, 1976), and ACT* (Anderson, 1983), that John Anderson began working on prior to joining CMU, but he has been at CMU since 1978.

Yet, it is notable how ACT-R and Soar—which started from rather different bases—turn out to be more alike now in how they map to the Common Model than either is to Sigma. The broader body of work considered in developing the consensus should further have helped prevent it from becoming too parochial. Still, much more was clearly required to reach a consensus across the broader range of relevant researchers and architectures in the field(s).

The next step on this path was holding a sequence of two new symposia (Flenner et al., 2018; Adler et al., 2019) and the creation of a community mailing list that last I checked had 270 members. At the first of these two symposia we were excited to find near unanimity that what we had proposed in the paper was an appropriate starting point for developing a Standard Model of the Mind. The biggest disagreement actually concerned its name—there was a significant sense that it was too aggressive, with worries about the possibility of the kind of top-down pressure that was seen with the *Standard Theory* in linguistics (Chomsky, 1965). So, we conducted a multi-stage community-wide poll that resulted in changing the name to the *Common Model of Cognition*.

We also started virtual working groups on eight topics for which we thought progress could be made with respect to the Common Model: declarative memory, emotion/mood/affect/motivation, higher-level knowledge, language processing, metacognition, neural/physiological underpinnings, perceptual/motor processing, and procedural memory. In an attempt to encourage a more distributed leadership to emerge, the three of us opted not to lead any of these working groups. Instead, we each served as resources in multiple of them.

Some of these groups got as far as producing reports on their topics for presentation at the second symposium. These were subsequently published in *Procedia Computer Science*. More recently an online journal—the *Common Model of Cognition Bulletin*—has been created with a direct focus on papers related to the Common Model. By now, however, all of the working groups have gone quiescent, likely due to the Common Model being only a side interest at best for most researchers. So, Christian, John, and I have recently begun rethinking what to do next.

We started by recruiting Andrea Stocco as a fourth member of our interuniversity collaboration. His background has ties to ACT-R, having done a post-doc with John Anderson at CMU, plus he brings in a cognitive neuroscience perspective that was largely lacking (although Christian does have some relevant background). The four of us now hold biweekly Zoom meetings to discuss how best to move forward in a manner that will maintain the community-consensus aspect while leveraging our joint commitment to the effort. From these conversations we have so far been able to organize the first of what will hopefully be a continuing sequence of specialized workshops on the Common Model; this one in June of 2022 on emotion and the Common Model,[63] which led to a proposal for a consensus on this topic (Rosenbloom et al., 2024) but not yet an actual consensus. We also were able to publish a short, general-interest piece in *The Conversation* on the Common Model and AGI (Rosenbloom, Lebiere, and Laird, 2022).

Still, as the old saying goes, "the proof of the pudding is in the eating." There is much more to be done in developing and using the Common Model, and in achieving consensus across broader communities. The ultimate success and worth of the Common Model will be determined by how all of this plays out and how useful it ends up being. However, I think there is a good chance that this effort will have a larger and more enduring impact than any of our individual cognitive architectures.

[62] Adding further connectivity, Gordon Bower was John Anderson's dissertation advisor at Stanford and John Anderson was the third member of my dissertation committee at CMU, after already counting Allen and Geoff Hinton (with Jaime Carbonell being the fourth).

[63] An earlier working group report on how emotion might relate to the Common Model can be found in Larue et al. (2018).

5. Dichotomic Maps: Scope, Similarities, Differences & Combinations [Tier 3]

A dichotomic map is a theoretical architecture that leverages the cross product of *n* dichotomies—that is, this-or-that distinctions—to structure and help to understand an *n*-dimensional space of possibilities (Rosenbloom, 2011b; Rosenbloom, 2019; Rosenbloom, Joshi, and Ustun, 2019; Rosenbloom, 2025). Consider the simple 2D map in Figure 10 that structures four abstract technologies relevant to both AI and cognitive science.

	Asymmetric	*Symmetric*
Uncontrolled	Reaction	Reasoning
Controlled	Deliberation	Reflection

Figure 10: A two-dimensional map based on (a)symmetric and (un)controlled dichotomies.

The two dichotomies in this map are asymmetric versus symmetric and uncontrolled versus controlled. For the first dichotomy consider, for example, rules versus logic. With rules, processing proceeds in only one direction during performance—forward over the structure that defines it—or, if there is processing in the reverse direction it is of a very different sort, whether abductive or in support of learning rather than performance. In contrast, deduction over logical statements can proceed in a uniform manner in any direction. For the second dichotomy consider, for example, parallel versus serial processing. The former typically proceeds in an uncontrolled manner with no decisions required in the process whereas the latter typically involves a controlled sequence of decisions.

Each location in the 2×2 space defined by Figure 10 receives a unique multi-label descriptor based on at which end of each dichotomy it sits. For example, reasoning (in its most straightforward form of monotonic logic) is symmetric (inference proceeds uniformly in any direction) and uncontrolled (decisions are not required during inference since every possible choice is equally valid), whereas deliberation (that is, decision making) is asymmetric (moving things forward in one direction) and controlled (with decisions required at each step). Reaction (independently triggering new thoughts or actions based on the current situation) and reflection (where you mentally stand back and consider what you are doing) are in their turn characterized by crossovers of the previous two descriptors.

Figure 10 may look more like a table than a map, but what makes it a (discrete) map is that the distance between the locations of two technologies defines how similar or different they are. In particular, for these maps a r*ectilinear*, *Manhattan*, or *L1 distance*—calculated simply as the sum of the horizontal and vertical steps between any two locations—is what matters. Thus, although it looks like a table, it is included here as a figure rather than as a table, as will all later maps in this memoir.

Whereas the Common Model is driven to identify the essential *similarities* among humanlike *cognitive architectures,* dichotomic maps—at least as used here—are instead driven to identify the key *differences* among *technologies* that provide core memory/knowledge, reasoning and learning capabilities more broadly across all of AI and cognitive science, including but not limited to those that underlie cognitive/A(G)I architectures. (As hinted at early in Section 3, they may also ultimately provide a path toward considering all possible forms of intelligence.) This exploration does not explicitly include technologies required for other cognitive capabilities, such as language processing, perception, and action, at least at present, but they are implicitly included when aligned with those found in core capabilities.

Consider Figure 11, which fleshes out the map from Figure 10 with a range of technologies that fall at each location. Two technologies are *identical with respect to such a map* if they fall at the same location in that map—that is, the distance between them is 0—even if they differ in other ways. For example, feedforward neural networks—that is, the classic multilayer perceptron (Rosenblatt, 1958) that has led to the modern efflorescence of deep learning (LeCun, Bengio, and Hinton, 2015) and generative AI (e.g., Brown et al., 2020)—and rules, at least in the parallel form found in Soar and Sigma, are identical with respect to this map because they are both asymmetric and uncontrolled. In contrast, two technologies are *opposite with respect to such a map* if every label differs between them—yielding a distance of 2 in this map—even if they may be similar in other ways. For example, with respect to this map, rules are opposite from heuristic search (which is symmetric and controlled).

	Asymmetric	*Symmetric*
Uncontrolled	Reaction – Arithmetic Circuits & Sum-Product Networks; DV Analogy; Functions; Feedforward Neural Networks & Backpropagation; Hidden Markov Models; Implicit Memory/Processing; Procedural Memory; Recurrent Neural Networks; Rules; Shallow Reasoning	Reasoning – Autoassociative Neural Networks; Constraint Propagation; Declarative Memory; Facts & Instances; Monotonic Logic; Probabilistic Graphical Models & Statistical Relational AI
Controlled	Deliberation – Decision Trees; Markov Decision Processes; Operators; Procedural Programs; Reinforcement Learning	Reflection – Analogy; Constraint Satisfaction; Heuristic & Probabilistic Search; Nonmonotonic Logic

Figure 11: Technologies mapping to the four locations.

More generally, anytime two technologies fall within the same row or column they bear a similarity, and anytime they fall within different rows or columns they bear a difference. For example, rules and monotonic logic are similar in that both are uncontrolled while they are different in that the former is asymmetric while the latter is symmetric (yielding a distance of 1). Similarly, both feedforward neural networks and autoassociative neural networks—such as Boltzmann machines (Ackley, Hinton, and Sejnowski, 1985)—are uncontrolled but differ in whether or not they are symmetric (also yielding a distance of 1). Thus, although the development of dichotomic maps is driven by differences, similarities still are relevant; they just show up at a smaller granularity than is seen with the consensus approach taken with the Common Model.

A map as a whole also goes beyond individual comparisons to provide a meaningful structure over a space of underlying technologies. Two kinds of dichotomic maps, corresponding to two kinds of spaces, have been explored so far. A *disciplinary map* considers the technologies underlying a discipline, such as AI or cognitive science. Figure 11 is, for example, a disciplinary map that reaches across significant portions of both AI and cognitive science. Figure 10 can also be seen as a disciplinary map, albeit one that provides a single abstract characterization of the technologies at each location. In this form, dichotomic maps are components of *metascience* (Rosenbloom, 2012a)—that is, science about science—that can be used to model entire disciplines.

Whereas a disciplinary map focuses on the overall structure of the corresponding discipline(s), along with the similarities and differences among its underlying technologies, an *architectural map* focuses on the particular combination of technologies integrated together within a specific architecture. The concern here therefore is how the technologies provide a set of complementary capabilities that jointly support complete systems. In such a map there should ideally be at most a single entry at each location that provides the functionality unique to that location unless there is either redundancy in the capabilities provided or excess abstraction in the dimensions of the map. In this form, dichotomic maps serve as theoretical meta-architectures for cognitive and A(G)I architectures. In comparison to the Common Model, dichotomic maps are broader in coverage—not limited just to humanlike cognitive architectures—yet more abstract and even further from being computational.

Figure 10 can serve double duty as not only a disciplinary map but also an (abstract) architectural map for cognitive/A(G)I architectures that integrate together instances of some or all of its four capabilities. Figure 12, for example, shows the analogous but more concrete structure for Soar, with each location specialized to how Soar embodies that capability.[64] Figure 13 shows the analogous structure for a very different architecture—

	Asymmetric	*Symmetric*
Uncontrolled	Rules	Facts & Episodes
Controlled	Selection & Execution	Subgoaling

Figure 12: Architectural map of Soar (adapted from Rosenbloom, 2019).

[64] In Rosenbloom (2019), an earlier variant of the (un)controlled dichotomy—(non)monotonic—was used. In Figures 12-14 it is updated with the more recent formulation from Rosenbloom (2025).

AlphaZero, which learned to play games such as *Chess* and *Go* (Silver et al., 2018). Although very effective at what it did, it is clear in this map that from a general architectural perspective, AlphaZero lacks a reasoning (symmetric uncontrolled) capability. AlphaZero would typically be considered a neural system rather than a cognitive architecture, but these two maps show that it can be seen to be much like Soar in the key capabilities it provides, with the differences ultimately due to other dichotomies.

	Asymmetric	*Symmetric*
Uncontrolled	Feedforward NNs	
Controlled	Game Moves	Monte Carlo Tree Search

Figure 13: Architectural map of AlphaZero (adapted from Rosenbloom, 2019).

Figure 14 shows the Common Model at a level of abstraction between that found in Figures 10 and 12. Although it fills in AlphaZero's gap with a generic notion of a declarative memory, the Common Model itself is missing a reflective capability that can be found in both Soar and AlphaZero.

Such architectural maps complement the Common Model by providing a more abstract structured means of exploring and comparing architectural mechanisms. It is, for example, possible to consider building a cognitive/A(G)I architecture based on any combination of one technology from each of the four locations in Figure 11, yielding a result that is *complete with respect to such a map*. Such maps also provide a principled means of detecting gaps in particular architectures in terms of where they fall short of completeness. Still, it is worth noting that not all disciplinary maps will have options at every location—possibly indicating unexplored regions of the discipline—and even when they do it isn't a given that complete architectures must necessarily include a technology from each location. What is provided at a minimum, however, is a way of structuring such considerations.

	Asymmetric	*Symmetric*
Uncontrolled	Procedural Memory	Declarative Memory
Controlled	Selection & Execution	

Figure 14: Architectural map of the Common Model of Cognition (adapted from Rosenbloom, 2019).

The creation of both disciplinary and architectural maps requires identification of appropriate dichotomies to serve as the dimensions that define them. One key aspect of this particular architectural exploration has been to look broadly at the dichotomies that are already familiar across AI and cognitive science, whether mentioned explicitly in prior work or only implicit, to see if they could be distilled down to a small enough set to serve as the dimensions of a sufficiently comprehensive yet still comprehensible map. The result to date has been the identification of three core dichotomies, two of which have already been mentioned—*(a)symmetric* and *(un)controlled*.

The other one is *(meta)data*, which leverages the distinction from the Common Model as to whether variations are based on symbolic data versus quantitative metadata, and explicitly includes the possibility found in the Common Model of a hybrid of data and metadata. This dichotomy characterizes the differences between rules and neural nets that don't show up when the focus is limited to the (a)symmetric and (un)controlled dichotomies.

Multiple traditional dichotomies can usefully be mapped onto each of these core dichotomies. Thus, even before laying out and filling in particular maps, the hope is that simplifying the understanding of AI and cognitive science by distilling a wide variety of pre-existing dichotomies across the disciplines into just three core ones is a useful contribution in its own right. Table 1, for example, shows thirty-one pre-existing dichotomies—whether explicit or implicit in their respective fields—that have been mapped onto (a)symmetric. This figure emphasizes an essential similarity among all of these prior dichotomies, as well as a similarity among their respective poles within the same column.

There are, of course, also significant differences among the dichotomies in such a list. Some of these differences are largely captured by other dichotomies and would therefore be clarified in a full 3D map. For example, the most significant difference between the traditional dichotomies of rules vs. logic and heteroassociative vs. autoassociative networks is that the former is based on symbolic data and the latter on quantitative metadata. Still, there will almost certainly be residual differences that remain even once all three core dichotomies are considered, likely providing an interesting topic for future exploration.

Table 1: Prior dichotomies that map onto the core (a)symmetric dichotomy.

	Asymmetric	*Symmetric*
1	Rules	Logic
2	Rule-Based Reasoning	First-Principles Reasoning
3	Shallow Reasoning	Deep Reasoning
4	Reaction	Reflection
5	Deliberation	Reflection
6	Greedy Search	Exhaustive Search
7	Policies	Planning
8	Arithmetic Circuits & Sum-Product Networks	Probabilistic Graphical Models
9	Function-Based Reasoning	Model-Based Reasoning
10	Heteroassociative Neural Networks	Autoassociative Neural Networks
11	Attention Networks	Self-Attention Networks
12	Procedural Memory	Declarative Memory
13	Implicit Memory	Explicit Memory
14	Action-Centered	Non-Action-Centered
15	Doing	Thinking
16	Skill	Knowledge
17	Procedural Learning	Declarative Learning
18	Skill Acquisition	Knowledge Acquisition
19	Macro/Composition Formation	Fact Storage
20	Backpropagation	Hebbian learning
21	Classification	Clustering
22	Supervised Learning	Unsupervised Learning
23	Discriminative Learning	Generative Learning
24	Genetic Programming	Inductive Logic Programming
25	Control (Reinforcement) Learning	Model Learning
26	Machine Learning	Knowledge Representation and Reasoning
27	Program	Data
28	Procedural Programming	Declarative Programming
29	Executable Code	Source Code
30	Primitive Emotions	Cognitively Elaborated Emotions
31	Causal Intervention	Causal Counterfactuals

The exploration of dichotomic maps has included one additional key piece—the identification of *criteria* that aid in understanding why it is useful to exploit one end of a dichotomy versus the other. In Rosenbloom (2025) a hierarchy of criteria is proposed that at the top-level spans accuracy, efficiency, generality, simplicity, improvability, usability, and operability. There I suggested, for example, that asymmetric technologies favor precision (one aspect of accuracy) and reactivity (one aspect of operability) whereas symmetric technologies favor correctness and robustness (two other aspects of accuracy) (Rosenbloom, 2025). Many dichotomies have arisen historically, and been argued over sometimes endlessly, in terms of one side versus the other being "correct," or at least "better." This work on dichotomic maps has instead typically assumed that it is better to span dichotomies to yield the best of both worlds rather than to force a choice between them. For example, the assumption is that a single cognitive/A(G)I architecture should attain all four of these useful properties under appropriate circumstances by spanning (a)symmetric.

The Common Model does exactly this for uncontrolled processing, by including both procedural and declarative memories (row 12), which are, respectively, assumed to be based on rules and facts (the most basic form of logic) to yield a variant of row 1. In contrast, one can imagine a variant that is based on neural networks that are heteroassociative (as with feedforward networks) and autoassociative (as with

Boltzmann machines), as in row 10. The Common Model, however, does not span the (a)symmetric dichotomy for controlled processing.

Taking this general idea further, combining criteria across the core dichotomies in a map yields a profile for each location that could help to understand why specific combinations of capabilities have been included, or which combinations should be included, in cognitive/A(G)I architectures. Such an analysis for (a)symmetric × (un)controlled has shown, for example, nuanced within criterial tradeoffs for efficiency and problem generality, and a strong across-criterial tradeoff along the main diagonal between reactivity and correctness (Rosenbloom, 2025).

This exploration into dichotomic maps grew out of a sequence of attempts to understand and explain how Sigma spanned such a broad range of capabilities from a small set of general mechanisms. This started more than a decade ago with a paper on how to build architectures for virtual humans that were *broad spectrum* (making creation of simple virtual humans simple while enabling extension to arbitrary sophistication as needed), *tightly integrated* (yielding much more than the sum of its parts), and *functionally elegant* (as already discussed for Sigma) (Rosenbloom, 2011b). The approach was to consider how a grab bag of dichotomies relevant to such architectures might be bridged, and how Sigma supported this.

Later follow up came in two pieces. The first considered how the structure of many cognitive architectures could be understood in terms of *(a)symmetry × (non)monotonicity* (Rosenbloom, 2019), an earlier take on (a)symmetric × (un)controlled. Figures 12-14 reflect this way of thinking. The second considered how the traditional but coarse-grained dichotomy of symbolic versus neural could be understood in terms of *(sub)symbolic × (a)symmetric × (non)combinatory* (Rosenbloom, Joshi, and Ustun, 2019), where the first dichotomy was an earlier take on (meta)data and the last was a different early take on (un)-controlled. The notion of opposite locations in a map is grounded generically in this work.

These two explorations were similar in nature and shared the (a)symmetric dichotomy, but it wasn't until later that I was able to figure out how to combine them into a single theoretical architecture. The biggest conceptual stumbling block was that the former focused on individual systems and the latter on whole disciplines. The solution manages to bring the four dichotomies together, into a revised set of three, while including the distinction between architectural and disciplinary maps that has been developed here.

The major work on this combination is in preparation (Rosenbloom, 2025), implying that the material in this section should still be considered preliminary. But it goes much deeper into the overall topic than was possible in a pair of conference papers, including covering: all four of the previous core dichotomies in a unified 3D manner, the preexisting dichotomies that map on to them, the criteria that can be leveraged in understanding the tradeoffs they embody, how the dichotomies interact (including a summary 3D map) and can be spanned, and the implications of each dichotomy for learning. I was following Maxim 13 here, of writing what you want to say and only then worrying about how it might be published, with the ultimate venue having turned out to be the middle of three major parts of Rosenbloom (2025).

6. Relational Model: Computing as a Great Scientific Domain [Tier 4]

As mentioned in Section 3.3, the relational model grew out of attempting to make sense of the diversity of topics, all at least loosely affiliated under a computing banner, explored during my interregnum. On first blush, there was no coherence across them other than that they all involved computing in some manner. However, as I explored different possibilities, a deeper congruity started to emerge in which the various topics exhibited characteristic relationships among particular scientific domains. This insight—it was an extended *aha* moment—first led to the development of the *Relational Model*, which provides a theoretical architecture delimiting the kinds of relationships that can appear between domains, and then the notion of *great scientific domains* as the primary slot fillers (that is, variations) in these relations, accompanied by the identification of computing as one of these domains (Rosenbloom, 2012a).

Like dichotomic maps, the Relational Model has a dyadic aspect and can be used in mapmaking. This yields a high-level commonality between the two architectures, but the details differ significantly. First,

the dyads in the former are built from pairs of technological traits whereas in the latter they are built from pairs of scientific domains or disciplines, typically what came to be called great scientific domains—the physical, life, social and computing sciences. Second, the relationship underlying the dyads in the former is effectively *opposite*, whereas in the latter it is either *implementation* or *interaction*. Third, rather than maps based on cross products of dyads, maps are built here by crossing relationships with domains.

Figure 15 shows a simple way to view the Relational Model over the three traditional great scientific domains. In it, the domains are abbreviated as their initials—P, L, and S—with the generic notion of a relationship denoted by +. The figure includes monadic regions in which each domain is considered in isolation (and its name is spelled out in full), dyadic regions that combine two domains, and one triadic region that combines all three. For example, physics, biology, and psychology fit respectively within the monadic regions of P, L, and S; climate change, cognitive neuroscience, and molecular biology fit respectively within the dyadic regions of P+S, S+L, and P+L; and ecology fits within the triadic region of P+S+L.

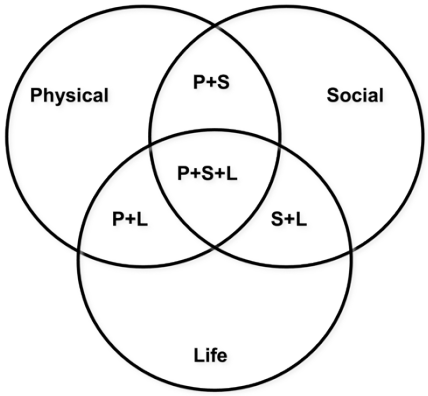

Figure 15: Overlaps among the traditional great scientific domains, yielding monadic, dyadic, and triadic regions (From *On Computing*: ©2012 MIT. All rights reserved.).

If the generic relationship denoted by + is refined into implementation (denoted by /) and interaction (denoted by ↔, →, or ←, depending on the direction(s) of influence involved in the interaction), a richer and more informative representation can result for the dyadic and triadic disciplines: climate change becomes P↔S, to represent both the human impact on climate change (P←S) and the impact of climate change back on humanity (P→S); cognitive neuroscience becomes S/L, to represent how the mind is implemented by the brain; molecular biology becomes L/P, to represent how life is based on molecules; and ecology becomes (P↔L)↔S, to represent the interaction among all three domains.

When computing is added to the picture, Figure 15 swells into Figure 16, embodying a range of possible overlaps, and thus relationships, with the other domains. Theory and algorithms are classical examples of monadic computing. The abundance of dyadic possibilities—involving computing plus one other domain, which may include itself—can be seen in Figure 17.[65] The one additional bit of representation here is Δ, which refers to any or all of the domains (P, L, S, and/or C).

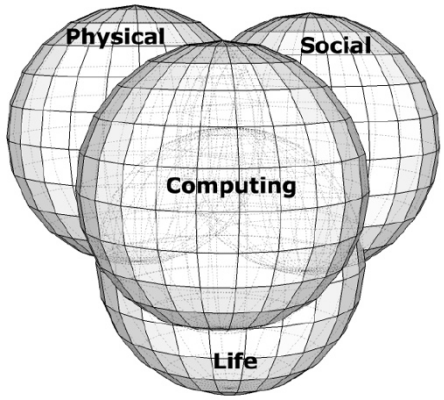

Figure 16: Three-dimensional representation of the four great scientific domains and their overlaps (From *On Computing*: ©2012 MIT. All rights reserved.).

It was the development of a very early version of this map, and how it was able to structure and relate so much of what was going on, or had gone on, in computing broadly that got me excited about the potential of the relational approach to help understand computing as a discipline. To highlight just two specific examples that are both based on relationships between the social and computing domains (S+C), but in very different ways, artificial intelligence implements the social domain by computers (S/C), whereas human-computer interaction involves a bidirectional interaction between the social and computing domains (S↔C). Both can thus be found in the same column but are distinguished by their rows.

[65] Although not metric in the same way as dichotomic maps, I have still found it useful to think of this as a map of the dyadic components of the great scientific domain of computing rather than simply as a table.

	C+P	C+L	C+S	C+C
C/Δ	mechanical, electromechanical, electronic, optical, quantum, chemical, precomputing, analog	phylogenetic, ontogenetic, immunological, neural, collective, biomolecular, biomimetic, evolutionary, autonomic, swarm, biomimetics	human cognition, mechanical Turks, Wizard of Oz, crowdsourcing, artificial intelligence	compilers, OS, emulation, reflection, abstractions, procedures, architectures, languages, databases
Δ/C	computational science, graphics, numerical analysis, virtual environments, digital physics	artificial life, systems biology, neural networks, virtual humans	artificial intelligence, cognitive modeling, computational sociology, econometrics	
C←Δ	sensors, scanners, computer vision, optical character recognition, localization	eye, gesture, expression and movement tracking, biosensors	IT industry (hardware and software), input devices, learning, authorization, speech understanding	networking, security, parallel and distributed computing, grids, clouds, multi-robot and multi-agent systems, automated programming and debugging, self-monitoring
C↔Δ	robots, informatics	brain-computer interfaces, assistive robotics, bio/medical informatics	computer and software engineering, human-computer interaction, full immersion, games, natural language, social robotics	
C→Δ	locomotion, fabrication, manipulation	bioeffectors, haptics, sensory immersion	presentation, screens, printers, graphics, speech generation, cognitive augmentation, computational thinking	

Figure 17: Two-dimensional map of dyadic computing. Each column denotes a relationship between computing and one other domain, which may be computing itself. Each row narrows in on one type of relationship and one direction within a type (From *On Computing*:).

Beyond dyadic relations, mapmaking gets more difficult, but the *Metascience Expression (ME)* language, which has at its core the notations that have already been introduced, enables specifying complex representations among multiple domains, as well as the possibility of individual domains playing multiple roles within a single expression. For example, one way to represent a robot is as l/(P↔C), where the lower-case "l" reflects a simulation of life rather than actual life. Building on this expression, a human enhanced with a prosthesis that contains an embedded computer can be denoted as L/(L↔l/(P↔C)), where the prosthetic—that is, a particular form of robot—interacts with the living body to implement a new form of life that is the combination of the two. We can also think of this new form of life as a *cyborg*.

The discussion of the Relational Model to this point has assumed that there are four great scientific domains, the three traditional ones—the physical, life, and social sciences—plus the computing sciences. Most people likely agree with the first three without needing to think about it, unless perhaps they are one of those who want to deny that the social sciences do science. One of the things that surprised me though

when I began this work was that, even with the long history of dividing up the sciences in this manner, there was no standard definition that explains why these three fit into this class while no other areas do. Using the Relational Model to understand the field of computing implicitly assumes that computing is of a kind with these other three, but many question whether computing is even a scientific discipline—as opposed to "merely" a form of engineering—not to mention a great scientific domain on a par with the other three. Making this claim required a further excursion into philosophy of science—or *metascience*, as discussed in Section 5 and as implied by the name of the ME language—to hypothesize the existence and nature of great scientific domains, and to understand whether the resulting definition applies to computing.

The definition I ultimately settled on for a great scientific domain was the *understanding and shaping of the interactions among a coherent, distinctive and extensive body of structures and processes*. As with the notion of architectural exploration in Section 2, this definition amounts to a post hoc rationalization, as the nature of the domain of computing was considered along with the three traditional domains, so an extra degree of skepticism is also in order here. But the broader hope is that it still provides a useful new perspective.

The combination of understanding and shaping implies that each domain—not just P and C, but also L and S—intertwines forms of science and engineering, cutting the overall intellectual pie at the top level by domain rather than by science versus engineering (Maxim 9). The domains are then distinguished by the characteristic structures and processes that they study and manipulate, with computing in particular concerning *informational structures* plus *processes that transform this information (*Rosenbloom, 2010). This yields a broad definition of computing as a whole that includes not only monadic C topics, but all topics in which C participates, whether science or engineering, academic or industrial, research versus applications, hardware or software, or based on static information or dynamic processing (Maxim 8). Both this paragraph and the previous one provide examples of Maxim 7, on breaking down walls rather than building them.

This way of cutting the pie provides, independent of the Relational Model, a novel theoretical architecture for science and engineering that extends to include a variety of disciplines not conventionally considered part of either, such as mathematics (part of the understanding side of C); the humanities, law, business, and education (all part of S); and the practice of medicine (part of the shaping side of L). When I gave a talk some years ago on the Relational Model at CMU, one of the strongest objections was to the idea of mathematics being part of computing. There are a range of very serious reasons for such an objection, but it is a direct consequence of the model based on how the domain of computing is defined broadly in terms of a focus on informational structures. Of course, alternative names could be considered for the domain, such as information sciences or mathematical sciences—coincidentally, the name of my undergraduate major—but I chose computing sciences because it emphasizes not just the structures, but also the transformational processes that are applied to these structures. Mathematics essentially becomes a static, formal component of the broad domain of computing.

At first, I considered the notion that computing was a great scientific domain as simply a *lemma*, or subsidiary point, that was necessary for consistency among the slot fillers in the Relational Model. It took Peter Denning, one of our leading thinkers about computing as a scientific discipline (Denning, 2005; Denning, 2013), to point out to me that this idea was important in its own right, and possibly even more important than the Relational Model itself. Since the four domains are in effect variations within the architecture defined by the Relational Model and the ME language, perhaps it is not surprising that with my architectural focus I missed this until it was pointed out to me. Once it was pointed out, however, it resulted in an inversion of how the book was written that summarized this exploration.

On Computing: The Fourth Great Scientific Domain required significant work spanning two sabbaticals, although with an unusually small-time gap between them, plus much of the time in between. I started with some initial ideas on how to proceed from notes I had jotted down while at ISI, plus an earlier

article on this topic (Rosenbloom, 2004)[66] and the graduate seminar I taught in 2007, but it still ended up as one of those efforts in which much of the requisite research occurred during the writing.

This book captured a number of insights that led me to reorient my thinking in multiple ways as I explored them, such as: the nature of great scientific domains, including their essential intertwining of understanding (that is, science) and shaping (that is, engineering); computing sciences as the fourth such domain; how the relational model could provide an interdisciplinary understanding of the breadth and structure of the computing sciences; and the potential of the relational model to extend beyond this to understanding a wider range of human intellectual endeavors. The relational model itself is the architecture that captures these insights.

Yet unfortunately but perhaps not surprisingly, *On Computing* did not have the impact I had hoped on the broad computing community. It did get a very nice review in *Nature* (Gilbey, 2012), one of the top general science journals/magazines, and has been cited 81 times on top of the citations garnered by various short articles on this general topic, but I can't say that the field as a whole has paid much attention. Perhaps, when I am long gone, the field will find some of these ideas useful as it reflects more on itself. At the moment, much of the attention is limited to members of the small subcommunity focused on understanding the nature of computing as a discipline, and a science, such as Peter Denning—who got interested in the work early on, encouraged and mentored me on it, and co-authored an article with me on it (Denning, and Rosenbloom, 2009)—Matti Tedre (2015), and Subrata Dasgupta (2016). This level of impact still matters, but my original ambitions were much greater.

In contrast, the book somewhat surprisingly had an impact on how we introduced computing to undergraduates at USC for some years. In 2013 the Chair of CS —Gaurav Sukhatme—suggested basing a new required freshman course on *On Computing* and offered to co-teach it with me. We did create and co-teach this course, pairing *On Computing* with *Computing for Ordinary Mortals* (Amant, 2013), with the goal of providing our freshmen a broad framework for understanding computing—as much more than just programming—that could carry them through the remainder of their education and beyond. The reaction from the students, however, was decidedly mixed, and after a few years of attempts at refining the general concept, the department downsized it to a more conventional focus on algorithmic/computational thinking.

Very early ideas that led to *On Computing* also inspired a restructuring of how we talked about the different research areas within the CS Department at USC—providing a novel, coherent way of characterizing it—but this too was abandoned after a few years because it was too eccentric to make sense to most people without a lot of associated explanation.

In a bit of a contrast, the notion of computing sciences from this work seems to have had made an impact more broadly on thinking about informatics education in Europe, with Denning and Rosenbloom (2009) cited as one of three complementary characterizations of *informatics*—the European term for a broader field than what is considered computer science in the United States—by The Committee on European Computing Education (2017).

One other interesting impact of the Relational Model has been in the area of *digital humanities*, which concerns the interactions between computing and the humanities (Nyhan, Terras, and Vanhoutte, 2013). I was introduced to this topic when a USC student, Max Feinstein, came to talk to me about applying the methodology of *critical code studies* (Marino, 2020), a part of digital humanities that treats computer programs like literature to be analyzed for cultural context and significance, to the code that implements Soar. After attending a workshop at USC on digital humanities to learn a bit more about the overall topic, I decided it would be fun to see what would happen if I applied the Relational Model to such a new and

[66] In this earlier article, I proposed restructuring computer science and engineering "rationally" so that the former actually focused on science and the latter on engineering, an idea that got me into hot water with colleagues in computer engineering—a discipline focused on computer hardware, architecture, and the software closest to the hardware—who were concerned I was suggesting either that they did no science or should do no science. With the newer focus on great scientific domains, this all shifted strongly toward the idea that it is best to intertwine science and engineering rather than creating hard-and-fast disciplinary boundaries between them.

relatively unformed area. So, I set out to explore in some detail the dyadic relationships possible between computing and the humanities, with a particular focus on understanding and structuring the existing topics within the field and on determining if there were any notable gaps in current work that might be worth proposing for future investigation.

It was indeed satisfying to discover that the Relational Model could bring some useful structure to the field, resulting in a journal article on the topic (Rosenbloom, 2012d).[67] This work attracted sufficient interest within the digital humanities community to be republished in the first reader aimed at helping to define the field (Nyhan, Terras, and Vanhoutte, 2013).

There is certainly much that can still be done with the Relational Model. Exploring its implications more broadly and deeply across computing comprises one aspect of this. But so too does exploring its aptness and utility across the rest of the intellectual world that does not include computing as a component, although I do not have the expertise myself in these other domains to take the lead in such an effort. A number of people have asked over the years as to whether the Relational Model can be used prescriptively—for example, to identify previously unconsidered areas of computing worth exploring—rather than just retrospectively as a way of understanding what already exists. Although the potential to do this was part of the original motivation for the model, and I have taken a few stabs at attempting to locate such examples—such as by systematically generating, and endeavoring to interpret, all relatively small expressions that include C, as well as by exploring what might be missing from the conventional scope of digital humanities—this mostly remains an intriguing area for future consideration.

7. Conclusion

Looking back on my career from today's perspective yielded the bipartite theme found in the introduction and which is repeated here:

1. *Substantive*: Mind and its encompassing disciplines
2. *Methodological*: Seeking insight via architectural exploration.

Although this pretty much sums it up in a neat package, my career was not particularly a planned one, except perhaps for an early long-term commitment to my grand challenge (Maxims 14-15), which in thematic terms becomes *seeking insight into mind* (in particular, complete humanlike minds). The remainder of the substantive subtheme, rather than deriving from a grand challenge, evolved late in my career more from frustration about the current state of things in areas in which I was working and a sense that deeper principles and forms of order would be discernable if I were to let myself explore the possibilities thoroughly.

The initial combination of frustration and a sense that I could do better was what earlier led to the work on Othello; yet, whereas it was clear throughout that effort that Othello would remain a diversion, the later work grew from what initially appeared to be another diversion—as it was in fact described in *On Computing*—into a far-reaching extension of the substantive subtheme. Overall, the choice of substantive topics was thus determined by a combination of curiousity—whether of the abstract intellectual sort or the sort induced by more concrete frustrations—plus local judgements concerning whether the problems were sufficiently important and amenable to my making significant contributions.

The choice of methodology was also largely not planned, instead typically emerging from a combination of the demands of the substantive subtheme with the biases induced by my personality. The specifics of the substantive subtheme were themselves biased by my personality, as well as by various

[67] The discussion of research methodology in this article on digital humanities—reflecting some of the amateur philosophy of science I needed to engage in while writing *On Computing*—has, it turns out, found its way back into the philosophy literature (Tallant, and Andow, 2020), perhaps entitling me now to claim a very small amount of professional status in that field.

environmental contingencies. This combination of influences has yielded an eccentric research path that only occasionally intersected with the mainstream of either AI or cognitive science. I would often try to understand the latest topic that was animating the mainstream, and to see what if any implications it might have for my own work—as with the incorporation of probabilistic graphical models at the core of Sigma based on a sense that they captured a much more general approach to many of the problems I had been dealing with, or failed to deal with, in Soar, and the subsequent extension of these graphical models to deep neural networks—but even then I would adapt the work for uses that were typically uninteresting to the mainstream. My research community, whether concerning mind or disciplines, was most often therefore rather small, when in fact such a community existed at all.

As was shown earlier in Figure 1, and as revisited in a slightly modified form in Figure 18, my research path ultimately yielded a career that can be modeled via a four-tiered architecture. At the core (Tier 1), as at the core of my career, was work on a sequence of three specific cognitive architectural lineages—XAPS ➜ Soar ➜ Sigma—that combined both computational and theoretical aspirations. The Common Model of Cognition (Tier 2) expands outwards to explore a consensus theoretical architecture for modeling humanlike minds. Dichotomic maps (Tier 3) expand further outwards to explore two kinds of theoretical architectures: architectural maps covering the structure of individual cognitive/A(G)I architectures (and the Common Model); and disciplinary maps covering the wider range of technologies found in AI and cognitive science. The Relational Model (Tier 4) expands all of the way out to not only include a theoretical architecture for the great scientific domain of computing, but also potentially for all intellectual endeavors.

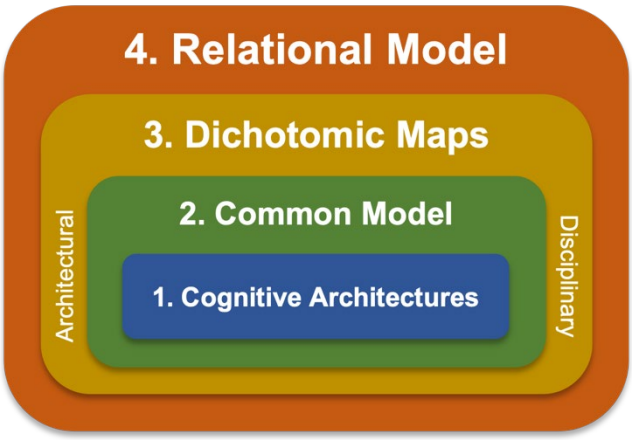

Figure 18: Reprise of substantive subtheme from Figure 6, with Tier 3 split into architectural and disciplinary maps.

At a coarse grain, Tiers 1 and 2 concern mind, Tier 4 concerns the discipline of computing (and beyond), and Tier 3 in some sense bridges these two rather distinct perspectives in providing both architectural maps of mind and disciplinary maps of AI and cognitive science. Digging even deeper, we can begin to explore what relationship, if any, exists between each adjoining pair of tiers. Starting with the innermost pair, there is a simple relationship given that the Common Model (Tier 2) is an abstract consensus over the humanlike architectures found in Tier 1. The relationship between Tiers 2 and 3 depends on using architectural maps for understanding the Common Model, and thus humanlike architectures more broadly. This has the potential to supplement the Common Model itself by explaining more fundamentally, in terms of dichotomies and criterial tradeoffs, why humanlike architectures combine the technologies they do, as well as to extend the consideration to beyond humanlike minds.

Considering Tiers 1 and 2 plus the architectural maps from Tier 3, the bulk of my "mind" career can be seen as comprising two large synthetic explorations of computational architectures—Soar and Sigma (with XAPS more minor)—and two large analytic explorations of theoretical architectures—the Common Model and architectural maps. Both Soar and Sigma clearly have theoretical aspects with respect to AI and cognitive science, but the computational aspect has been dominant for both.

Figure 19 was developed jointly with John Laird and shows the overall trajectory of my Tiers 1 and 2 in context with his work on the same two tiers and Allen Newell's (and Herb Simon's) work on Tier 1. Although Tier 3 doesn't currently fit into this picture, it still provides a succinct summary of the main body of my work on mental architecture.

Tiers 3 and 4, with Tier 3 now narrowed to disciplinary maps, can be brought together via an abstract expression from the ME language—Δ+Δ—representing two domains of any sort (Δ) connected by any

relation (+), assuming that the domains can, as necessary, be arbitrarily narrow. Applying the ME language to narrower subdomains was already anticipated in the work on digital humanities, where H was introduced as the humanities portion of S. It may then make sense to add an *opposite* relation to the Relational Model in order to represent dichotomies.

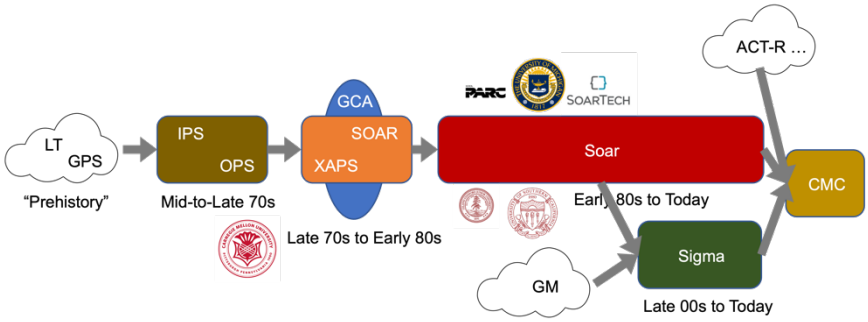

Figure 19: Research trajectory on cognitive architectures, developed jointly with John Laird. The affiliations below the Soar box are mine, whereas those above it are John's. LT (the Logic Theorist) (Newell, and Simon, 1956) and GPS (the General Problem Solver) (Newell, Shaw, and Simon, 1959) were critical to where we got our start but are prehistory from the perspective of our own trajectories.

It also turns out that, although I didn't notice it at the time, the two relations at the heart of the Relational Model—implementation and interaction—map rather well onto two key approaches to spanning dichotomies that were earlier referred to as *reduction* and *addition* (Rosenbloom, 2011b). Reduction concerns how elements at one level can be reduced to combinations of elements at the level below, while implementation takes the dual perspective of how a combination of elements at one level can yield elements at the level above. Addition adds more mechanisms that interact with existing ones to enable a wider range of capabilities overall. Given this mapping, rather than adding *opposite* as a relation type, given how different it is in nature from the other two, these two tiers might instead be brought together by applying the Relational Model to explain how dichotomies can be spanned.

Extending the cognitive-architecture research trajectory from Figure 19 to view my career as a whole on a chronological basis yields a triphasic structure spanning *early*, *middle*, and *late* phases. The early phase, roughly spanning the first two decades, largely aligned with Allen Newell's notion of desires and diversions. The substantive subtheme—that is, the desires—was laser focused on mind, with work in Tier 1 on XAPS and Soar via a methodological focus on cognitive architecture. During this phase, which was clearly the most impactful with respect to the broader research community, I could succinctly be considered a cognitive architecture researcher. This phase did include one major diversion—on Othello—plus the large-scale application of Soar to military simulation that in a sense blended desires and diversions. This early phase also coincided with the peak in my leadership with respect to the AI community.[68]

The middle phase was my interregnum—or *mid-life (research) crisis*—which lasted roughly a decade. This phase was athematic with respect to my research career as a whole, although more narrowly it did have a focus on initiating interdisciplinary new directions via a rather eclectic methodology.

The late phase, lasting roughly a decade and a half so far, has been a time of rejuvenation, continued growth, and surprising efflorescence. Research on mind continued in Tier 1, with Sigma, while also extending into Tiers 2 and 3. In addition, the substantive theme grew beyond a focus on mind itself to include mind-encompassing disciplines, with work in Tiers 3 and 4. As well, during this phase the methodological subtheme fully ripened and leadership with respect to the USC community peaked with a year as president of the university's faculty. Although the early phase was the most impactful one in a traditional academic sense, this late phase has in its own way been highly innovative—at least in my own judgement—and perhaps the most productive. It serves as a prime example of late maturation and of how not all of one's best ideas are limited to one's youth (Maxim 27).

[68] In alignment with Maxim 6, I, for example, served as Councilor and Conference Chair for the American Association for the Advancement of Artificial Intelligence (AAAI)—which is now the Association for the Advancement of Artificial Intelligence (AAAI)—and as Chair of the Association for Computing Machinery (ACM) Special Interest Group on Artificial Intelligence (which was SIGART then but is SIGAI now).

I have also realized and accepted quite late in writing this memoir that it was not until this last phase that I was truly mature enough intellectually, and perhaps embued with sufficient intellectual self-confidence, to set new, large-scale, research problems for myself (Maxim 26). The big early problems, such as chunking and Soar were set by Allen Newell, and for the latter John Laird as well, although I did contribute significantly to the pursuit of both while also devising many relatively smaller problems to which I could contribute. The work on Othello did originate with me, as did the extension of the work on practice to include stimulus-response compability and the interaction between these two, but these were more small-to-medium-sized efforts that did not start with any grand ambitions. And thus my career continued for a number of years while working on Soar and a range of small-to-medium ideas.

Yet, the more recent work on Sigma, dichtomies, and computing—plus more recent reflections that started with the methodology section in this memoir, continued with explicit thoughts on architectures (Rosenbloom, 2022; Rosenbloom, 2025), symbols (Rosenbloom, 2023; Rosenbloom, 2025), intelligence (Rosenbloom, 2025), and theories (Rosenbloom, 2025), and and may extend even further—are examples of problems I would not have been able to set for myself earlier, or for the most recent reflections even have had sufficient self-awareness to pull off. The earlier inability to set new large research problems for myself may in fact have been a large, previously-unacknowledged—even to myself—contribution to the sense of burnout discussed in Section 3.

This caveat notwithstanding, my research over the years reached a level of influence—with the work on Soar clearly being the most impactful—that I have been elected a fellow of three professional societies—the American Association for the Advancement of Science (AAAS), the Association for the Advancement of Artificial Intelligence (AAAI), and the Cognitive Science Society. I have also received several awards for work on Soar (the *Herbert A. Simon Prize for Advances in Cognitive Systems* in 2018, jointly with John Laird), Sigma (the *Kurzweil Award for Best AGI Idea* in 2011 and the *Kurzweil Award for Best AGI Paper* in 2012), and the Relational Model (*USC Phi Kappa Phi Faculty Recognition Award* in 2012-2013). The conference paper covering the work on symbols (Rosenbloom, 2023) also recently received the *Springer Prize for Best Paper at AGI-2023*.

Moving forward, I don't currently anticipate doing more serious programming, and so am unlikely to continue with Sigma, unless the temptation to give it a personality becomes too great. Fortunately, two students have investigated ideas for what might come after Sigma—Abram Demski pursued an approach in his dissertation that involved an even more general layer below Sigma's extended factor graphs, based on weighted logic programming (Eisner, and Filardo, 2011), that can implement them and much more (Demski, 2022); and Jincheng Zhou, a USC student at the time who started working with us as a sophomore, has investigated a generalization of Sigma that deals better with continuous data and holds out the possibility for a new approach to chunking (Zhou, and Ustun, 2021). These are both computational architectures, with implementations in Julia and Python respectively.

What I do anticipate continuing to work on are the non-computational theoretical cognitive architectures found in Tiers 2 and 3. The most immediate need is to finish the part of Rosenbloom (2025) concerned with Tier 3, which I have left to the last as it raised the thorniest issues. Further out is considering the question of whether such maps can aid not just in understanding, but also in shaping—for example, of how we build minds and how we structure much of A(G)I and cognitive science. Understanding better how the two kinds of maps—disciplinary and architectural—in Tier 3 relate to each other and to the other tiers remains of interest as well. Looking way, way out, there is a possibility of extending some combination of these two tiers, plus the thoughts on intelligence—and the intelligence space—that has made its way into Rosenbloom (2025), toward understanding the space of all possible minds.

Whatever the directions might end up being, the point will always be searching for insights with potential impact; and to do so via architectural explorations of topics I care about, and that leverage my strengths while working within or around my weaknesses and limitations.

Acknowledgements

The writing of this memoir has included time on long-term disability and a final sabbatical semester, both of which were funded by USC, so I would like to thank the university for this support. I have received helpful comments on various drafts of this memoir from Claire Jansa, Julia Kim, Paul Middlebrooks, Paul Allen Newell, Judy Pfeffer, and several JAGI reviewers. I would also like to thank anyone who has had any amount of positive impact on my career or life. Many of the most central people have already been mentioned by name in one context or another, and the remainder are too numerous to mention.

Appendix A. Maxims

This appendix distills some of what I have learned over my career into maxims plus minimal elaborations on them. Some of these were explicit early on while others remained implicit for much of my career until becoming clarified by later reflection. Not all of these maxims will make sense for everyone, nor do they all work for every career style (as Allen Newell also qualified the maxims in *Desires and Diversions*). Some may also be trite, but they have all mattered to me and my career. They are enumerated here, loosely grouped by topic, and with a final set that revisits some of Allen's maxims with respect to my career.

Insight and Impact

1. *It is the insight that matters.* Before an insight is applied it must be validated—whether theoretically, experimentally, or computationally—to an extent appropriate to the application, but it is paramount not to lose the centrality of the insight itself in the weeds of how it is validated.

2. *Struggling in the weeds can be essential for insight*. Whether the weeds are in early exploration or later validation, remaining above—or out of—them can mean losing a major source of insight. This may appear to contradict the previous maxim but is instead intended to complement it.

3. *Implementation is critical for understanding computational architectures*. This can be viewed as a special case of the previous maxim.

4. *Making an impact requires combining compelling visions with attention to detail*. This is likely trite, but essential nonetheless, and it applies in all phases of research.

Problems, Disciplines, and Venues

5. *Pledge your primary allegiance to problems rather than to disciplines*. Work to the extent you can in whatever disciplines, departments, institutes, or organizations facilitate your progress on your problems of choice. Although this risks making you neither fish nor fowl, and thus complicating your career path, it has its own rewards, and if you succeed, everything will turn out well.

6. *Service to your disciplines/organizations is an essential complement to problem-focused research*. This can be a win for both you and your disciplines/organizations and may in particular help you toward being both fish and fowl.

7. *Break down walls rather than build them*. This is my own variation on the now common phrase "build bridges rather than walls." It is particularly helpful when trying to survive and thrive while exploring interdisciplinary problems.

8. *Define disciplines broadly rather than narrowly*. This is related to the previous maxim but suggests constructing any walls that must be built as far as can be workable from a discipline's nominal center. In terms of the Relational Model, don't just include the discipline's monadic region within the walls, but also dyadic and polyadic regions. Accept that the result will be overlaps among disciplines, and that the walls themselves will therefore be rather weak, as they should in fact be.

9. *Great scientific domains include both science (understanding) and engineering (shaping)*. An effective intertwining of both should become more and more important as each continues to become more informative and essential for the other across all four domains.

10. *There is power in blending AI and cognitive science in exploring minds*. Although related to all three of the previous maxims, this one is much more specific, emphasizing the utility of judiciously combining the functional and experimental evidence yielded by these two fields, as well as both science and engineering, in exploring the space of potential cognitive architectures that apply to both fields. The use of *judicious* here implies not being a slave to the details of either field but leveraging what is most critical from each.

11. *Stronger research methods within a discipline push out weaker ones*. Strength here is the level of veracity that a method can guarantee. This maxim is true even in those areas of a discipline in which the stronger methods within that discipline don't (yet?) apply. Erecting new disciplinary boundaries that wall off the latter areas can sometimes help keep them active, but the walled off areas then risk becoming insular and irrelevant.

12. *Distinguish between absolute and relative method strength*. Use the strongest methods available for the problem at hand (that is, attend to relative strength) but do not shy from important problems just because there are not strong methods available, at least at present, for them (that is, disregard absolute strength). Absolute strength is critical when assessing the veracity, and considering the application, of any results, and may even rightly play a role in determining funding levels; however, it should not be a bar—or even lead to derogation of topics or researchers—where not attainable.

13. *Be idea driven in writing papers rather than venue driven*. In particular, figure out what you want to say and use that to drive where you try to publish the result rather than the other way around.

Problems, Challenges and Hypotheses

14. *Commit to solving a problem that is big enough to make a difference.* But be sure you care deeply about it, have the skills to contribute to it, and enjoy the day-to-day aspects of working on it.

15. *Dedication to a grand challenge can provide focus and long-term joy*. The times at which major advancements—and associated *aha* moments—become apparent can yield particularly heady experiences. However, they can be few and far between. So, if you are to pursue such an approach, it is best to be okay with delayed gratification (while, as in the previous maxim, finding sufficient short-term reward in the day-to-day aspects).

16. *Milk an ambitious hypothesis for all it is worth.* The more ambitious the hypothesis the more committed you need to be to it. It can take years, or even decades or more, to understand and derive implications from, and to validate or invalidate truly ambitious hypotheses. There is a deepening of understanding that occurs in the process, however, that is invaluable.

17. *Be willing to make radical changes when there are compelling reasons to do so*. This maxim complements the previous one. You have to be willing, and to give yourself permission, to abandon ambitious hypotheses when it is clear they no longer have legs or when the case for an alternative hypothesis becomes overwhelming.

18. *Focusing on a long-term grand challenge helps avoid being sucked into research fads*. But, as a result your work may, at best, go through cycles of being in and out of fashion, depending on how well it matches the current zeitgeist. When it is unfashionable, resources to pursue the work can be difficult to obtain and publication can also be difficult.

19. *Create your own niche*.[69] Given your particular skills, experience, expertise, history, knowledge, personality, and perspective (see Maxim 25) find—or more realistically create—and pursue problems and/or approaches that no-one else can, or even would consider, focusing on. This gives you a chance of making a unique contribution that only you could have made rather than constantly looking over your shoulder in fear of the competition getting to the finish line first. Such problems and/or approaches are moreover likely ones that you would particularly enjoy pursuing.[70]

Functional Elegance

20. *Always keep your eye out for elegance and beauty, and for what simplifies, unifies, and explains*. I think of this as the physicist's mindset on science, but for me it applies across the board. I seem to be wired in such a manner that without such a focus I feel like I'm stuck in a quagmire and get little satisfaction from what by other metrics might count as progress.

21. *Seek similarities before considering differences.* In other words, look for as much simplicity and elegance as possible before acknowledging that there must be messiness. There is also a relationship here to Occam's razor.

22. *Seek explanations before settling for descriptions*. Descriptions—even mathematically rigorous or statistically accurate ones—may be pragmatically useful but settling for them when deeper explanations may be within reach is a lost opportunity for true insight.

23. *Dig deeply into proposed -otomies*. Traditional dichotomies may conflate multiple distinct concepts that typically align but that can be better understood when teased apart. Higher-order -otomies, in their turn, may be better understood as cross products of simpler dichotomies.

More Personal

24. *Always be as generous as can legitimately be justified with co-authorship*. Don't let other considerations get in the way of this.

[69] This can be considered as part of the creed of the outsider, or even of the eccentric. It is a maxim I only identified quite late in the writing of this memoir, and which only truly took shape later in my life. It can be considered as my own personal twist on a Herb Simon maxim—of, as a scientist, finding your own "private" part of the universe to mine that no-one else is looking at—that I heard of indirectly through Allen Newell.

[70] Combining this maxim with others in this section should help ensure that what you do is important as well as uniquely personal. The risk though is that if your niche eventually proves to be unproductive, or even irrelevant, then you may not only work in obscurity during your lifetime, but your life's work may ultimately and correctly be consigned to oblivion. You may go down in history, if history remembers you at all, as a crackpot rather than as an eccentric or even as an incomparable.

25. *Understand your own strengths and limitations*. Leverage the former as much as you can, even when they are side effects of a broader "abnormal/negative condition." Although understanding and accepting your limitations can be quite trying, attempting to overcome or work around them can be extraordinarily valuable.

26. *Capabilities and interests mature at different rates both within and across individuals*. Some mature young, others take decades, and others may never mature. It isn't terribly productive to try to force them before their time.

27. *Not all of one's best ideas are limited to one's youth*. Continued maturation can lead to significant innovation even late in life. Some of what I consider my best ideas were developed late in my life.

28. *Family is the most important consideration in the big decisions*. However, how to prioritize with respect to the smaller decisions is a more subtle problem, particularly if you want to get a lot done.

29. *Early prejudices should be left to childhood.* I had, for example, early prejudices against USC (growing up in a UCLA/Stanford family) and the US military (growing up in the 60s) that I had to overcome—and I believe it was right to overcome—as my career developed.

Revisiting Several of Allen Newell's Maxims

Some of these are summarized from memory of what Allen Newell used to say and others are reproduced from (Laird & Rosenbloom, 1992). The point of having them here though is to discuss how they played out in my own career.

- *There will be diversions; make them count*. This applies most directly to my interregnum, which was a decade-long diversion from work on cognitive architectures, even if it has ultimately been cast in a non-diversionary manner in this memoir. I was able to make it count locally—that is, on its own—by developing the Relational Model and the notion that computing was a great scientific domain. But I was also able to make it count in returning back to cognitive architecture with a kernel of a new idea and the explicit understanding that the only way I felt I was making progress was when I was guided by functional elegance (Maxims 20-23). This was the core of what I tried to do with Sigma, providing much of the oomph behind its first three desiderata. It was also implicit in why I felt boxed in earlier while working on Soar rather than feeling free to take the road John Laird later did of adding more memories and learning mechanisms to the architecture. It is furthermore what I have tried to do with the three non-computational theoretical architectures.

- *To do interdisciplinary work you must become a professional in all of the relevant disciplines*. I did manage to uphold this maxim throughout much of my career, particularly with my work across AI and cognitive science. However, at various times, I did cut corners, including in working on speech understanding in Sigma and in engaging in philosophy of science. I did of course read on both topics but did not approach professional competence in either. This may very well have hurt both efforts as well as leading to a sense of failing to meet standards, whether my own and/or Allen's, but at this point I just need to accept this.

- *Listen to the architecture*. This was Allen's way of saying that you should treat the architecture as a source of knowledge, much like the phenomena of interest is already treated. To me, the part of

this that has always mattered most is the exhortation to explore what the architecture implies about how to implement a new functionality before jumping in to add a new mechanism for it. As such, it ties in with Maxims 20-22. More generally, my notion of functional elegance is a variant of this earlier maxim, although instead of being a heuristic for how to proceed it is a characterization of what you want to achieve. In this form it applies to a broader range of situations, including for example identifying a small set of dichotomies that structure AI and cognitive science.

- *Deep scientific ideas are exceedingly simple. Others usually see them as trivial.* This focus on simplicity can be seen across my career in both the desideratum of functional elegance that I have striven to satisfy in my work on cognitive architectures and in what I have tried to accomplish with the various analytical methods. The implied concern with triviality has also arisen a number of times, both in my own mind and in the minds of others.

- *Preserve the insight and deepen it. Deep ideas transform themselves beyond imagining.* This can be viewed as a forerunner to Maxim 16 above. It is reflected in my work on cognitive architectures; particularly Soar and Sigma, on which I spent more than a decade each attempting to deepen my/our understanding. It can also be seen in the work on the Relational Model, which went from back-of-the-envelope calculations to short papers to a full book-length treatment that required multiple years and sabbaticals to complete.

- *New things get started by evolution or chance, not design*. In writing this memoir, part of the intent was to lay out the circumstances that led to the choices made over my career. Much of the driving force behind these choices can be seen as evolutionary or as a product of chance. Still, I would add two more factors when these choices concern starting new things—the ability to discern what is new and of potential import when it appears and the confidence in it and in oneself to pursue it.

- *We are all flawed—We all carry our burden of incapacities*. Another part of the intent behind writing this memoir was to lay out a whole person, with both strengths and flaws, and to understand how their combination led to the trajectory followed by my career. This maxim can also be seen as relating to Maxims 25, 26, and 29 above.

References

Ackley, D. H.; Hinton, G. E.; and Sejnowski, T. J. 1985. A Learning Algorithm for Boltzmann Machines. *Cognitive Science*. 9: 147-169.

Adler, A.; Dasgupta, P.; Depalma, N.; Eslami, M.; Freedman, R. G.; Laird, J. E.; Lebiere, C.; Lohan, K.; Mead, R.; Roberts, M.; Rosenbloom, P. S.; Senft, E.; Stein, F.; Williams, T.; Wray, K. H.; Yaman, F.; and Zilberstein, S. 2019. Reports of the 2018 AAAI Fall Symposium. *AI Magazine*. 40: 66-72.

Amant, R. 2013. *Computing for Ordinary Mortals*. Oxford, UK: Oxford University Press.

Anderson, J. R. 1976. *Language, Memory and Thought*. Hillsdale, NJ: Erlbaum Associates.

Anderson, J. R. 1983. *The Architecture of Cognition*. Cambridge, MA: Harvard University Press.

Anderson, J. R.; Bothell, D.; Byrne, M. D.; Douglass, S.; Lebiere, C.; and Qin, Y. 2004. An Integrated Theory of the Mind. *Psychological Review*. 111: 1036-1060.

Anderson, J. R.; and Bower, G. H. 1973. *Human Associative Memory.* Hillsdale, NJ: Lawrence Erlbaum Associates.

Arens, Y.; and Rosenbloom, P. eds. 2002. *Responding to the Unexpected: Report of the Workshop Held in New York City, February 27-March 1, 2002*.

Bell, C. G.; and Newell, A. 1971. *Computer Structures: Readings and Examples*. New York, NY: McGraw-Hill.

Brown, T. B.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; Agarwal, S.; Herbert-Voss, A.; Krueger, G.; Henighan, T.; Child, R.; Ramesh, A.; Ziegler, D. M.; Wu, J.; Winter, C.; Hesse, C.; Chen, M.; Sigler, E.; Litwin, M.; Gray, S.; Chess, B.; Clark, J.; Berner, C.; McCandlish, S.; Radford, A.; Sutskever, I; and Amodei, D. 2020. Language Models are Few-Shot Learners. In *Proceedings of the Thirty-Fourth Conference on Neural Information Processing Systems*, 1877–1901.

Card, S.; Moran, T.; and Newell, A. 1983. *The Psychology of Human Computer Interaction*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Chen, J.; Demski, A.; Han, T.; Morency, L-P.; Pynadath, D.; Rafidi, N.; and Rosenbloom, P. S. 2011. Fusing Symbolic and Decision-Theoretic Problem Solving + Perception in a Graphical Cognitive Architecture. In *Proceedings of the Second International Conference on Biologically Inspired Cognitive Architectures*, 64-72.

Cho, B.; Rosenbloom, P. S.; and Dolan, C. P. 1991. Neuro-Soar: A Neural-Network Architecture for Goal-Oriented Behavior. In *Proceedings of the Thirteenth Annual Conference of the Cognitive Science Society*, 673-677.

Chomsky, N. 1965. *Aspects of the Theory of Syntax*. Cambridge, MA: MIT Press.

Cooper, R.; Fox, J.; Farringdom, J.; and Shallice, T. 1996. Towards a Systematic Methodology for Cognitive Modelling. *Artificial Intelligence*. 85: 3-44.

Dasgupta, S. 2016. *Computer Science: A Very Short Introduction*. Oxford, UK: Oxford University Press.

Demski, A. 2022. *Cognitive Equations: Fixed-Point Finding as an Intermediate Layer in Cognitive Architecture.* PhD Dissertation, University of Southern California.

Denning, P. J. 2005. Is Computer Science Science? *Communications of the ACM*. 48: 27-31.

Denning, P. J. 2007. Computing is a Natural Science. *Communications of the ACM*. 50: 13-18.

Denning, P. J. 2013. The Science in Computer Science. *Communications of the ACM*. 56: 35-38.

Denning, P. J.; and Rosenbloom, P. S. 2009. Computing: The Fourth Great Domain of Science. *Communications of the ACM*. 52: 27-29.

Eisner, J.; and Filardo, N. W. 2011. Dyna: Extending Datalog for Modern AI. In *Datalog Reloaded* ed. O. de Moor; G. Gottlob; T. Furche; and A. Sellers. Berlin, Germany: Springer-Verlag.

Eliasmith, C. 2013. *How to Build a Brain: An Architecture for Neurobiological Cognition*. New York, NY: Oxford University Press.

Feigenbaum, E. 1992. *Expert Systems: Principles and Practice*. Stanford KSL Tech. Report 91-79.

Fitts, P. M.; and Seeger, C. 1953. S-R Compatibility: Spatial Characteristics of Stimulus and Response Codes. *Journal of Experimental Psychology*. 46: 199–210.

Flenner, A.; Fraune, M. R.; Hiatt, L. M.; Kendall, T.; Laird, J. E.; Lebiere, C.; Rosenbloom, P. S.; Stein, F.; Topp, E. A.; Unhelkar, V. V.; and Zhao, Y. 2018. Reports of the AAAI 2017 Fall Symposium Series. *AI Magazine*. 39: 81-86.

Forgy, C. L. 1982. Rete: A Fast Algorithm for the Many Pattern/Many Object Pattern Match Problem. *Artificial Intelligence*. 19: 17-37.

François J. 1977. Evolution and Tinkering. *Science*. 196: 1161–1166.

Frey, P. W. 1981. The Santa Cruz Open Othello Tournament for Computers. *BYTE*. 6: 26-37.

Getoor, L.; and Taskar, B. 2007. *Introduction to Statistical Relational Learning*. Cambridge, MA: MIT Press.

Gilbey, J. 2012. Virtually There. *Nature*. 491: 331.

Gluck, K. A.; and Laird, J. E. eds. 2019. *Interactive Task Learning: Humans, Robots, and Agents Acquiring New Tasks through Natural Interactions*. Cambridge, MA: MIT Press.

Hennessy, J. L.; and Patterson, D. A. 2012. *Computer Architecture: A Quantitative Approach (5th Edition)*. Waltham, MA: Morgan Kaufmann.

Hill, R. W.; Chen, J.; Gratch, J.; Rosenbloom, P. S.; and Tambe, M. 1997. Intelligent Agents for the Synthetic Battlefield: A Company of Rotary Wing Aircraft. In *Proceedings, Ninth Conference on Innovative Applications of Artificial Intelligence*, 1006-1012.

Hossenfelder, S. 2018. *Lost in Math: How Beauty Leads Physics Astray*. New York, NY: Basic Books.

Jilk, D. J.; Lebiere, C.; O'Reilly, R. C.; and Anderson, J. R. 2008. SAL: An Explicitly Pluralistic Cognitive Architecture. *Journal of Experimental & Theoretical Artificial Intelligence*. 20: 197-218.

John, B. E.; and Newell, A. 1990. Toward an Engineering Model of Stimulus-Response Compatibility. In *Stimulus-Response Compatibility: An Integrated Perspective* eds. R. W. Proctor; and T. G. Reeve. Amsterdam: North-Holland.

John, B. E.; Rosenbloom, P. S.; and Newell, A. 1985. A Theory of Stimulus-Response Compatibility Applied to Human-Computer Interaction. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 212-219.

Jones, R. M.; Laird, J. E.; Nielsen, P. E.; Coulter, K. J.; Kenny, P.; and Koss, F. V. 1999. Automated intelligent pilots for combat flight simulation. *AI Magazine*. 20: 27-41.

Joshi, H.; Rosenbloom, P. S.; and Ustun, V. 2014. Isolated Word Recognition in the Sigma Cognitive Architecture. *Biologically Inspired Cognitive Architecture*. 10: 1-9.

Joshi, H.; Rosenbloom, P. S.; and Ustun, V. 2016. Continuous Phone Recognition in the Sigma Cognitive Architecture. *Biologically Inspired Cognitive Architectures*. 18: 23-32.

Kivunja, C. 2018. Distinguishing between Theory, Theoretical Framework, and Conceptual Framework: A Systematic Review of Lessons from the Field. *International Journal of Higher Education*. 7: 44-53.

Koller, D.; and Friedman, N. 2009. *Probabilistic Graphical Models: Principles and Techniques*. Cambridge, MA: MIT Press.

Kotseruba, I.; and Tsotsos, J. K. 2020. 40 years of Cognitive Architectures: Core Cognitive Abilities and Practical Applications. *Artificial Intelligence Review*. 53: 17-94.

Koza, J. R. 1992. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. Cambridge, MA: MIT Press.

Kschischang, F. R.; Frey, B. J.; and Loeliger, H. 2001. Factor Graphs and the Sum-Product Algorithm. *IEEE Transactions on Information Theory*. 47: 498-519.

Kuhn, T. 1962. *The Structure of Scientific Revolutions*. Chicago, IL: University of Chicago Press.

Lacroix, G. L.; and Cousineau, D. 2007. The Introduction to the Special Issue on "RT(N) = a + b N-c: The Power Law of Learning 25 Years Later." *Tutorials in Quantitative Methods for Psychology*. 2: 77-83.

Laird, J. E. 1984. *Universal Subgoaling*. Ph.D. Thesis, Carnegie-Mellon University.

Laird, J. E. 2012. *The Soar Cognitive Architecture*. Cambridge, MA: MIT Press.

Laird, J. E. 2022. Introduction to Soar. arXiv:2205.03854.

Laird, J. E.; Lebiere, C.; and Rosenbloom, P. S. 2017. A Standard Model of the Mind: Toward a Common Computational Framework across Artificial Intelligence, Cognitive Science, Neuroscience, and Robotics. *AI Magazine*. 38: 13-26.

Laird, J.E.; and Newell, A. 1983. A Universal Weak Method: Summary of Results. In *Proceedings of the International Joint Conference on Artificial Intelligence*, 771-773.

Laird, J. E.; Newell, A.; and Rosenbloom, P. S. 1987. Soar: An Architecture for General Intelligence. *Artificial Intelligence*. 33: 1-64.

Laird, J. E.; and Rosenbloom, P. S. 1990. Integrating Execution, Planning, and Learning in Soar for External Environments. In *Proceedings of the Eighth National Conference on Artificial Intelligence*, 1022-1029.

Laird, J. E.; and Rosenbloom, P. S. 1992. In Pursuit of Mind: The Research of Allen Newell. *AI Magazine*. 13: 17-45.

Laird, J. E.; and Rosenbloom, P. S. 1996. The Evolution of the Soar Cognitive Architecture. In *Mind Matters: A Tribute to Allen Newell* eds. D. M. Steier; and T. M. Mitchell. Mahwah, NJ: Lawrence Erlbaum Associates.

Laird, J. E.; Rosenbloom, P. S.; and Newell, A. 1984. Towards Chunking as a General Learning Mechanism. In *Proceedings of the National Conference on Artificial Intelligence*, 188-192.

Laird, J. E.; Rosenbloom, P. S.; and Newell, A. 1986a. Chunking in Soar: The Anatomy of a General Learning Mechanism. *Machine Learning*. 1: 11-46.

Laird, J. E.; Rosenbloom, P. S.; and Newell, A. 1986b. *Universal Subgoaling and Chunking: The Automatic Generation and Learning of Goal Hierarchies*. Hingham, MA: Kluwer Academic Publishers.

Lakatos, I. 1978. *The Methodology of Scientific Research Programmes: Philosophical Papers Volume 1*. Cambridge, UK: Cambridge University Press.

Langley, P.; Laird, J. E.; and Rogers, S. 2009. Cognitive Architectures: Research Issues and Challenges. *Cognitive Systems Research*. 10: 141-160.

Langley, P.; Thompson, K.; Iba, W.; Gennari, J.; and Allen, A. J. 1989. *An Integrated Cognitive Architecture for Autonomous Agents*. Technical Report 89-28 Department of Information & Computer Science, University of California, Irvine, CA.

Larue, O.; West, R.; Rosenbloom, P. S.; Dancy, C. L.; Samsonovich, A. V.; Petters, D.; and Juvina, I. 2018. Emotion in the Common Model of Cognition. *Procedia Computer Science*. 145: 730-739.

LeCun, Y. 2022. A Path Towards Autonomous Machine Intelligence, Version 0.9.2, 2022-06-27.

LeCun, Y.; Bengio, Y.; and Hinton, G. E. 2015. Deep Learning. *Nature*. 521: 436-444.

Lehman, J. F.; Laird, J. E.; and Rosenbloom, P. S. 1998. A Gentle Introduction to Soar, an Architecture for Human Cognition. In *An Invitation to Cognitive Science (Second Edition), Volume 4: Methods, Models and Conceptual Issues* eds. D. N. Osherson; S. Sternberg; and D. Scarborough. Cambridge, MA: MIT Press.

Lehman, J. F.; Lewis, R. L.; and Newell, A. 1991. Integrating Knowledge Sources in Language Cpmprehension. In *Proceedings of the Thirteenth Annual Conferences of the Cognitive Science Society* 461-466.

Lenat, D. B. 1983. Eurisko: A Program that Learns new Heuristics and Domain Concepts. *Artificial Intelligence*. 21: 61-98.

Lenat, D. B.; and Guha, R. V. 1989. *Building Large Knowledge-Based Systems; Representation and Inference in the Cyc Project*. Boston, MA: Addison-Wesley Longman.

Linnaeus, C. 1735. *Systema Naturæ*.

Marino, M. C. 2020. *Critical Code Studies*. Cambridge, MA: MIT Press.

McCulloch, W. S.; and Pitts, W. 1943. A Logical Calculus of the Ideas Immanent in Nervous Activity. *Bulletin of Mathematical Biophysics*. 5: 115-133.

Miller, G. A. 1956. The Magical Number Seven, Plus or Minus Two: Some Limits on our Capacity for Processing Information. *Psychological Review*. 63: 81-97.

Milnes, B. G.; Pelton, G.; Doorenbos, R.; Hucka, M.; Laird, J.; Rosenbloom, P.; and Newell, A 1992. *A Specification of the Soar Cognitive Architecture in Z*. Carnegie Mellon University, USA, Technical Report.

Minsky, M.; and Papert, S. 1969. *Perceptrons*. Cambridge, MA: MIT Press.

Newell, A. 1973. You Can't Play 20 Questions with Nature and Win: Projective Comments on the Papers of this Symposium. In *Visual Information Processing: Proceedings of the Eighth Annual Carnegie Symposium on Cognition* ed. W. G. Chase. MA: Academic Press.

Newell, A. 1990. *Unified Theories of Cognition*. Cambridge, MA: Harvard University Press.

Newell, A.; and Rosenbloom, P. S. 1981. Mechanisms of Skill Acquisition and the Law of Practice. In *Cognitive Skills and their Acquisition* ed. J. R. Anderson. Hillsdale, NJ: Erlbaum.

Newell, A.; Shaw, J.C.; and Simon, H.A. 1959. Report on a General Problem-Solving Program. In *Proceedings of the International Conference on Information Processing*, 256-264.

Newell, A.; and Simon, H. A. 1956. The Logic Theory Machine: A Complex Information Processing System. *IRE Transactions on Information Theory*. 2: 61-79.

Newell, A.; and Simon, H. A. 1972. *Human Problem Solving*. Englewood Cliffs, NJ: Prentice-Hall.

Nyhan, J.; Terras, M. M.; and Vanhoutte. E. eds. 2013, *Defining Digital Humanities: A Reader*. Farnham, UK: Ashgate.

Oishi, S.; and Westgate, E. C. 2022. A Psychologically Rich Life: Beyond Happiness and Meaning. *Psychological Review*. 129: 790–811.

Park, J. S.; O'Brien, J. C.; Cai, C. J.; Morris, M. R.; Liang, P; and Bernstein, M. S. (2023). Generative Agents: Interactive Simulacra of Human Behavior. arXiv:2304.03442.

Pausch, R. 2008. *The Last Lecture*. New York, NY: Hachette USA.

Pearl, J. 1988. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. San Francisco, CA: Morgan Kaufman.

Plato ~375 BC. *Republic*.

Pólya 1945. *How to Solve It*. Garden City, NY: Doubleday.

Post, E. 1943. Formal Reductions of the General Combinatorial Decision Problem. *American Journal of Mathematics*. 65: 197-215.

Pynadath, D. V.; Rosenbloom, P. S.; and Marsella, S. C. 2014. Reinforcement Learning for Adaptive Theory of Mind in the Sigma Cognitive Architecture. In *Proceedings of the Seventh Annual Conference on Artificial General Intelligence*, 143-154.

Pynadath, D. V.; Rosenbloom, P. S.; Marsella, S. C.; and Li, L. 2013. Modeling Two-Player Games in the Sigma Graphical Cognitive Architecture. In *Proceedings of the Sixth Conference on Artificial General Intelligence*, 98-108.

Rosenblatt, F. 1958. The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain. *Psychological Review*. 65: 386-408.

Rosenbloom, P. S. 1982. A World-Championship-Level Othello Program. *Artificial Intelligence*. 19: 279-320.

Rosenbloom, P. S. 1983. *The Chunking of Goal Hierarchies: A Model of Practice and Stimulus-Response Compatibility*. Ph.D. Thesis, Carnegie-Mellon University.

Rosenbloom, P. S. 2004. A New Framework for Computer Science and Engineering. *IEEE Computer*. 37: 23-28.

Rosenbloom, P. S. 2006. A Cognitive Odyssey: From the Power Law of Practice to a General Learning Mechanism and Beyond. *Tutorials in Quantitative Methods for Psychology*. 2: 43-51.

Rosenbloom, P. S. 2010. Computing and Computation. *ACM Ubiquity Symposium: 'What is Computation?'*.

Rosenbloom, P. S. 2011a. Mental Imagery in a Graphical Cognitive Architecture. In *Proceedings of the Second International Conference on Biologically Inspired Cognitive Architectures*, 314-323.

Rosenbloom, P. S. 2011b. Bridging Dichotomies in Cognitive Architectures for Virtual Humans. In *Proceedings of the AAAI Fall Symposium on Advances in Cognitive Systems*.

Rosenbloom, P. S. 2012a. *On Computing: The Fourth Great Scientific Domain*. Cambridge, MA: MIT Press.

Rosenbloom, P. S. 2012b. Deconstructing Reinforcement Learning in Sigma. In *Proceedings of the Fifth Conference on Artificial General Intelligence*, 62-271.

Rosenbloom, P. S. 2012c. Extending Mental Imagery in Sigma. In *Proceedings of the Fifth Conference on Artificial General Intelligence*, 272-281.

Rosenbloom, P. S. 2012d. Towards a Conceptual Framework for the Digital Humanities. *Digital Humanities Quarterly*. 6.

Rosenbloom, P. S. 2014. Deconstructing Episodic Learning and Memory in Sigma. In *Proceedings of the Thirty-Sixth Annual Conference of the Cognitive Science Society*, 1317-1322.

Rosenbloom, P. S. 2019. (A)Symmetry × (Non)Monotonicity: Towards a Deeper Understanding of Key Cognitive Di/Trichotomies and the Common Model of Cognition. In *Proceedings of the Seventeenth Annual Meeting of the International Conference on Cognitive Modeling*.

Rosenbloom, P. S. 2022. Thoughts on Architecture. In *Proceedings of the Fifteenth Conference on Artificial General Intelligence*, 364-373.

Rosenbloom, P. S. 2023. Rethinking the Physical Symbol Systems Hypothesis. In *Proceedings of the Sixteenth International Conference on Artificial General Intelligence*, 207-216.

Rosenbloom, P. S. 2025. *On Minds: Reflections of a Cognitive Architect*. Cambridge, MA: MIT Press. In Preparation.

Rosenbloom, P. S.; Demski, A.; Han, T.; and Ustun, V. 2013. Learning via Gradient Descent in Sigma. In *Proceedings of the Twelfth International Conference on Cognitive Modeling*, 35-40.

Rosenbloom, P. S.; Demski, A.; and Ustun, V. 2016a. The Sigma Cognitive Architecture and System: Towards Functionally Elegant Grand Unification. *Journal of Artificial General Intelligence*. 7: 1-103.

Rosenbloom, P. S.; Demski, A.; and Ustun, V. 2016b. Rethinking Sigma's Graphical Architecture: An Extension to Neural Networks. In *Proceedings of the Ninth Conference on Artificial General Intelligence*, 84-94.

Rosenbloom, P. S.; Demski, A.; and Ustun, V. 2017. Toward a Neural-Symbolic Sigma: Introducing Neural Network Learning. In *Proceedings of the Fifteenth Annual Meeting of the International Conference on Cognitive Modeling*.

Rosenbloom, P. S.; Gratch, J.; and Ustun, V. 2015. Towards Emotion in Sigma: From Appraisal to Attention. In *Proceedings of the Eighth Conference on Artificial General Intelligence*, 142-151.

Rosenbloom, P. S.; Joshi, H. ; and Ustun, V. 2019. (Sub)Symbolic × (A)Symmetric × (Non)Combinatory: A Map of AI Approaches Spanning Symbolic/Statistical to Neural/ML. *Advances in Cognitive Systems*. 8: 113-131.

Rosenbloom, P. S.; Laird, J. E.; Lebiere, C.; Stocco, A.; Granger, R. H.; and Huyck, C. 2024. A Proposal for Extending the Common Model of Cognition to Emotion. In *Proceedings of the Twenty-Second International Conference on Cognitive Modeling*.

Rosenbloom, P. S.; Laird, J. E.; McDermott, J.; Newell; A.; and Orciuch; E. 1985. R1-Soar: An Experiment in Knowledge-Intensive Programming in a Problem-Solving Architecture. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 7: 561-569.

Rosenbloom, P. S.; Laird, J. E.; and Newell, A. 1987. Knowledge Level Learning in Soar, In *Proceedings of the Sixth National Conference on Artificial Intelligence*, 499-504.

Rosenbloom, P. S.; Laird, J. E.; and Newell, A. eds. 1993. *The Soar Papers: Research on Integrated Intelligence*. Cambridge, MA: MIT Press.

Rosenbloom, P. S.; Lebiere, C.; and Laird, J. E. 2022. Cross-Pollination among Neuroscience, Psychology and AI Research Yields a Foundational Understanding of Thinking. *The Conversation*.

Rosenbloom, P. S.; Lee, S.; and Unruh, A. 1990. Responding to Impasses in Memory-Driven Behavior: A Framework for Planning, In *Proceedings of the Workshop on Innovative Approaches to Planning, Scheduling, and Control*, 181-191.

Rosenbloom, P. S.; Lee, S.; and Unruh, A. 1993. Bias in Planning and Explanation-Based Learning. In S. Minton (Ed.), *Machine Learning Methods for Planning*. San Mateo, CA: Morgan Kaufmann.

Rosenbloom, P. S.; and Newell, A. 1986. The Chunking of Goal Hierarchies: A Generalized Model of Practice. In *Machine Learning: An Artificial Intelligence Approach, Volume II* eds. R. S. Michalski; J. G. Carbonell; and T. M. Mitchell. Los Altos, CA: Morgan Kaufmann.

Rosenbloom, P. S.; and Newell, A. 1988. An Integrated Computational Model of Stimulus-Response Compatibility and Practice. In *The Psychology of Learning and Motivation, Volume 21* ed. G. H. Bower. San Diego, CA: Academic Press.

Rosenbloom, P. S.; and Ustun, V. 2019. An Architectural Integration of Temporal Motivation Theory for Decision Making. In *Proceedings of the Seventeenth Annual Meeting of the International Conference on Cognitive Modeling*.

Rumelhart, D. E.; Hinton, G. E.; and Williams, R. J. 1986. Learning Representations by Back-Propagating Errors. *Nature*. 323: 533-536.

Russell, B. 1918. *The Philosophy of Logical Atomism*.

Russell, S. J.; and Norvig, P. 2010. *Artificial Intelligence: A Modern Approach (Third Edition)*. Upper Saddle River, NJ: Pearson.

Rychener, M. D. 1983. The Instructable Production System: A Retrospective Analysis. In *Machine Learning: An Artificial Intelligence Approach* eds. R. S. Michalski; J. G. Carbonell; and T. M. Mitchell. Palo Alto, CA: Morgan Kaufmann.

Rychener, M. D.; and Newell, A. 1978. An Instructable Production System: Basic Design Issues. In *Pattern-Directed Inference Systems* eds. D. A. Waterman; and F. Hayes-Roth. Orlando, FL: Academic Press.

Scerri, P.; Pynadath, D. V.; Johnson, L.; Rosenbloom, P.; Schurr, N.; and Tambe, M. 2003. A Prototype Infrastructure for Distributed Robot-Agent-Person Teams. In *Proceedings of the Second International Joint Conference on Autonomous Agents & Multiagent Systems*, 433-440.

Silver, D.; Hubert, J.; Anonoglou, I.; Lai, M.; Guez, A.; Lanctot, M.; Sifre, L.; Kumaran, D.; Graepel, T.; Lillicrap, T.; Simonyan, K.; and Hassabis, D. 2018. A General Reinforcement Learning Algorithm that Masters Chess, Shogi, and Go through Self-Play. *Science*. 363: 1140-1144.

Simon, H. A. 1969. *The Sciences of the Artificial*. Cambridge, MA: MIT Press.

Simon, H. A. 1974. How Big Is a Chunk? *Science*. 183: 482-488.

Steier, D. M.; Laird, J. E.; Newell, A.; Rosenbloom, P. S.; Flynn, R.; Golding, A.; Polk, T. A.; Shivers, O. G.; Unruh, A.; and Yost, G. R. 1987. Varieties of Learning in Soar: 1987. In *Proceedings of the Fourth International Workshop on Machine Learning*, 300-311.

Stocco, A.; Steine-Hanson, Z.; Koh, N.; Laird, J. E.; Lebiere, C.; and Rosenbloom, P. S. 2021. Analysis of the Human Connectome Data Supports the Notion of a "Common Model of Cognition" for Human and Humanlike Intelligence. *Neuroimage*. 235: 118035.

Sun, R. 2016. *Anatomy of the Mind: Exploring Psychological Mechanisms and Processes with the Clarion Cognitive Architecture*. New York, NY: Oxford University Press.

Swartout, W. R., Gratch, J., Hill, R. W., Hovy, E. H., Marsella, S., Rickel, J.; and Traum, D. R. 2006. Toward Virtual Humans. *AI Magazine*. 27: 96-108.

Tallant, J.; and Andow, J. 2020. English Language and Philosophy. In *The Routledge Handbook of English Language and Digital Humanities* eds. S. Adolphs; and D. Knight. London, UK: Routledge.

Tambe, M.; Johnson, W. L.; Jones, R. M.; Koss, F.; Laird, J. E.; Rosenbloom, P. S.; and Schwamb, K. B. 1995. Intelligent Agents for Interactive Simulation Environments. *AI Magazine*. 16: 15-39.

Tambe, M.; Schwamb, K. B.; and Rosenbloom, P. S. 1995. Building Intelligent Pilots for Simulated Rotary Wing Aircraft. In *Proceedings of the Fifth Conference on Computer Generated Forces and Behavioral Representation*, 39-44.

Tedre, M. 2015. *The Science of Computing: Shaping a Discipline*. Boca Raton, FL: CRC Press.

The Committee on European Computing Education 2017. Informatics Education in Europe: Are We All In The Same Boat? New York, NY: Association for Computing Machinery.

Turing, A. M. 1937. On Computable Numbers, with an Application to the Entscheidungsproblem. *Proceedings of the London Mathematical Society*. 42: 230-65.

Ustun, V.; Kumar, R.; Reilly, A.; Sajjadi, S.; and Miller, A. 2021. Adaptive Synthetic Characters for Military Training. In *Proceedings of the Interservice/Industry Training, Simulation, and Education Conference (I/ITSEC) 2021*.

Ustun, V.; and Rosenbloom, P. S. 2015. Towards Adaptive, Interactive Virtual Humans in Sigma. *Proceedings of the Fifteenth International Conference on Intelligent Virtual Agents*, 98-108.

Ustun, V.; Rosenbloom, P. S.; Kim, J.; and Li, L. 2015. Building High Fidelity Human Behavior Models in the Sigma Cognitive Architecture. In *Proceedings of the 2015 Winter Simulation Conference*, 3124-3125.

Ustun, V.; Rosenbloom, P. S.; Sagae, K.; and Demski, A. 2014. Distributed Vector Representations of Words in the Sigma Cognitive Architecture. In *Proceedings of the Seventh Annual Conference on Artificial General Intelligence*, 196-207.

Ustun, V.; Rosenbloom, P. S.; Sajjadi, S.; and Nuttall, J. 2018. Controlling Synthetic Characters in Simulations: A Case for Cognitive Architectures and Sigma. In *Proceedings of the Interservice/Industry Training, Simulation, and Education Conference (I/ITSEC) 2018*.

Waldrop, M. M. 1988a. Toward a Unified Theory of Cognition. *Science*. 241: 27-29.

Waldrop, M. M. 1988b. Soar: A Unified Theory of Cognition? *Science*. 241: 296-298.

Wilczek, F. 2015. Why is Physics Beautiful? *World Economic Forum*.

Zhou, J.; Cui, G.; Zhang, Z.; Yang, C.; Liu, Z.; and Sun, M. 2019. Graph Neural Networks: A Review of Methods and Applications. arXiv:1812.08434.

Zhou, J.; and Ustun, V. 2021. PySigma: Towards Enhanced Grand Unification for the Sigma Cognitive Architecture. In *Proceedings of the Fourteenth Conference on Artificial General Intelligence*, 355–366.