Advances in Cognitive Systems (2019) 157–174 Submitted 6/2018; published 4/2019

Themes and Paradigms for Artificial Reasoning

Pat Langley PATRICK.W.LANGLEY@GMAIL.COM
Institute for the Study of Learning and Expertise, Palo Alto, California 94306 USA

Adrienne Raglin ADRIENNE.J.RAGLIN.CIV@MAIL.MIL
Stephen Russell STEPHEN.M.RUSSELL8.CIV@MAIL.MIL
Army Research Laboratory, 2800 Powder Mill Road, Adelphi, MD 20783 USA

Abstract

In this paper, we review research on computational mechanisms for complex reasoning. We start by defining the reasoning task in broad terms and discussing elements that arise when addressing it. Next we consider different types of multi-step reasoning problems and alternative paradigms that vary in their underlying assumptions. Finally, we discuss some complicating factors that make reasoning difficult, along with ways that controlled experiments can reveal strengths and limitations of alternative approaches. Our aim is to provide readers with a balanced, accessible introduction to the important area of artificial reasoning.

1. Introduction

One generally recognized facet of human intelligence is the ability to *reason* – to draw nontrivial conclusions from available information. People exhibit this capacity in many different contexts, including planning, design, language processing, and activity understanding. This makes the development of computational reasoning systems a core subfield of artificial intelligence. The pursuit is sometimes referred to in the literature as *automated reasoning*, but that term is often associated with particular approaches to the problem. In this paper, we will instead use *artificial reasoning* to denote all computational methods for addressing this broad class of tasks.

Research in this area has clear importance and great potential for aiding both commercial and military endeavors. Human decision makers are often confronted with complex reasoning tasks that challenge their cognitive abilities. These require people to access substantial knowledge, allocate many resources, trade off multiple objectives, monitor complicated situations, and respond to unexpected developments. Even highly trained personnel could benefit from computational aids that offload some of this information-processing effort. If we can automate key facets of the reasoning process, then the resulting software should improve both the quality and speed of human decision making across many different settings.

For example, suppose a military brigade is sent to capture a small city, occupied by both enemy forces and civilians, while minimizing casualties to its own troops and locals. The command staff must formulate a plan that achieves these objectives and complies with established doctrine. Once plan execution has been initiated, the command team must use reports from the field to track its progress, explain unexpected developments, and adapt the original course of action in response. These activities rely on complex reasoning that combines knowledge about military tactics, information on available troops and equipment, details about the target city, and data from the field. Civilian scenarios for providing disaster relief, managing energy grids, and overseeing logistics operations involve similar issues and comparable levels of cognitive complexity.

In the next section, we define more clearly the task of artificial reasoning, after which we discuss some basic issues that arise in its study. Next we examine some broad categories of problems in which such reasoning is necessary and seven paradigms that differ in their assumptions about representation and processing. We conclude by reviewing a number of complicating factors that can make reasoning tasks difficult, along with empirical approaches to evaluating artificial reasoning systems that test hypotheses about their behavior. We cannot be exhaustive in such a brief paper, but we attempt to provide a balanced overview that prepares readers for learning more about this important, challenging, and wide-ranging area.

2. The Task of Artificial Reasoning

We should start by clarifying what we mean by *artificial reasoning*. Of course, the adjective indicates some mechanical process that operates on a created artifact, such as a digital computer. Thus, it conveys much the same idea as *automated reasoning*, a phrase that is widely used in the AI literature (Robinson & Voronkov, 2001; Wos et al., 1984), but that has two unnecessary connotations. The first association is that reasoning involves the mechanical application of logic, which is overly restrictive, as other computational approaches to the task are possible. The second is that the process operates in an entirely autonomous manner, rather than allowing for human intervention or interaction. These suggest the need for a new term, and here we adopt the phrase 'artificial reasoning' in direct imitation of the name 'artificial intelligence'.

A greater challenge is to define what we mean by 'reasoning'. As with any computational task, we can specify reasoning in terms of what it accepts as input and what it produces as output:

- *Given:* A set of knowledge elements that encode expertise;
- *Given:* A set of known or believed facts that describe a situation;
- *Given:* A query to be answered or goal to be achieved (optional);
- *Find:* One or more reasoning chains that connect these facts (and optionally the query) through ground versions of knowledge elements.[1]

This formulation is slightly misleading, as it suggests that reasoning involves 'batch' processing, but we can view incremental varieties as invoking this procedure repeatedly, with outputs from one step serving as inputs to the following one.

Note that our specification makes no reference to logic. Clearly, logical formulae offer one way to state knowledge elements and facts, while logical proofs provide one way to encode grounded reasoning chains. However, other representations of this content are also possible, and any computational process that maps the above inputs to the above outputs should count as reasoning. We

1. By 'ground', we mean instances of knowledge elements that refer to specific entities indicated by constants rather than generic ones denoted by variables. We are not referring to 'grounded' representations that define abstract symbols in terms of low-level perceptions and actions.

will see later that one can tackle the general task in many distinct ways. The only *defining* feature is that one finds a set of grounded or instantiated knowledge elements which connect the given facts. Typically, this set will involve more than one knowledge element, which distinguishes multi-step reasoning from the simpler task of *prediction* or *inference*. The latter conventionally involves applying a single stored knowledge structure, rather than composing elements dynamically.[2]

Some formulations of artificial reasoning place additional constraints on the relation between inputs and outputs. One common variant requires that any newly generated facts, such as answers to queries, be *correct* in that they follow deductively from input facts and available knowledge. Another is that the process be *complete* in that it produces all correct facts or answers, rather than a proper subset of them. Both constraints are typically associated with logical approaches to artificial reasoning. They certainly merit attention, but we will not include them in our definition of the task, as they seem more appropriate for evaluating the *quality* of reasoning processes. There are certainly examples of human reasoning that are incorrect or incomplete, but this does not mean that people do not reason, and we should judge our machine reasoners by the same rules.

3. Aspects of Artificial Reasoning

The computational study of reasoning has four distinct aspects. We will discuss these in a particular order, because choices about some facets depend on prior decisions about others. The most basic issue concerns *representation*, as we cannot discuss processes until we have considered the structures over which they operate. As noted earlier, we can distinguish among three types of content:

- *Knowledge* comprises long-term structures that serve as the building blocks of reasoning. These are typically generic, in that they describe classes of situations rather than individual instances.[3] They are modular and compositional, in that one can combine them to create connected chains.
- *Facts* are 'grounded' structures that describe particular entities or relations among them. These are typically short-term elements that change across reasoning problems and even during their solution, although they may also encode static items that hold indefinitely.
- *Queries* or *goals* specify desired entities or relations among them. These may be grounded elements, but they are typically more generic, and they are usually associated with individual problems. Queries and goals may involve combinations of structures, such as conjunctions.

Knowledge, facts, and queries can take many different forms, but they are always encoded as symbol structures (Newell & Simon, 1976), which means simply that they are organized sets of persistent patterns. Symbols and symbol structures can *denote* entities, relations, or processes in the environment, but they can also refer to other internal structures. These constructs may also be *interpreted*, as when a system runs a stored program to carry out some complex activity. Symbol structures can take many forms, with logical statements, production rules, and semantic networks being classic examples, but even neural networks fall under this broad umbrella.

2. In cognitive psychology, 'reasoning' is associated with conscious or controlled processing, while 'inference' connotes unconscious or automatized processing. Thus, inference is used to describe the unconscious mechanisms of language understanding, even though it often involves chains of reasoning. Here we use the two terms in different senses.
3. Later, we will see that analogical approaches instead assume grounded structures that are grouped into cases. Other frameworks, such as *scripts* and *frames*, also assume larger-scale organizations of knowledge.

The second aspect of reasoning concerns *performance* – the process of using knowledge and facts to produce new facts, which may in turn answer queries, support explanations, or achieve goals. This involves two complementary mechanisms. The first *instantiates* knowledge elements using existing facts, usually by matching or unifying the former with the latter, to produce grounded versions of these elements. The second *composes* two or more instantiated structures, typically through a *chaining* process, to generate new facts that, hopefully, will satisfy queries or goals. Reasoning is inherently compositional, which distinguishes it from simpler mechanisms for prediction and inference. Composition is the means by which one produces chains of reasoning that move beyond individual elements. Because many such chains are possible, this process requires *search* through a space of alternatives and, because exhaustive techniques are often intractable, many systems rely on heuristics to focus attention on promising candidates.

The third facet is *learning*, which is responsible for improving behavior on some performance task. This comes in two primary forms that have different implications for reasoning. One involves acquiring generative knowledge elements that enable new reasoning chains that were not possible before or that reach the same conclusion in fewer steps. Introduction of such knowledge can improve a system's completeness, but it can sometimes reduce efficiency by creating more options (Iba, 1989). A second variety of learning instead acquires constraints or heuristics that guide the search inherent in any compositional system (Sleeman, Langley, & Mitchell, 1982). This can improve a system's efficiency by reducing the number of candidates considered, but it can also improve correctness by eliminating inappropriate inferences. Learning is always defined with respect to some performance element that it aims to improve. Recent efforts on statistical induction have focused primarily on single-step inference tasks, but there is a long history of research on learning for multi-step reasoning problems, as reviewed in Langley (1996).

The final perspective is *transfer*, which refers to improvement in learning on some task as a function of prior training on earlier problems that are related in some manner. The classic explanation of transfer in humans is the reuse of knowledge elements acquired in previous settings. Thus, if the new performance task depends on cognitive structures learned earlier, then there is no need to acquire them again, so learning is more rapid. A special case involves representation change, where the introduction of high-level features or predicates makes learning easier. Another concerns compositional learning of new structures which make direct connections that previously required multiple steps. Although transfer is usually viewed as a way to improve learning, in some cases *negative* transfer can slow it by leading the system astray rather than offering aid. Transfer is always defined in the context of both performance and learning mechanisms, on which it builds. Recent years have seen substantial work on transfer for one-step inference tasks, but earlier efforts focused on its role in multi-step reasoning and problem solving.

We should also discuss briefly the classic distinction among three types of reasoning – *deduction*, *abduction*, and *induction*. In this view, deduction combines a set of generic rules with specific facts to derive new facts. Such reasoning is seen as 'justified' because, if the rules and initial facts are true, then the derived facts are also true. Abduction also starts with a set of generic rules and specific facts, but instead postulates new facts that, if true, would let one deduce the initial facts. In contrast, induction works from a set of specific facts (and possibly rules) to posit new rules that generalize over those facts. Both abduction and induction are seen as 'unjustified', since the results cannot be derived from the givens, but they may still be very plausible. One problem with these characterizations is that there are ways to reformulate one type of reasoning into another. For instance, some researchers have recast induction in terms of abduction, others have treated abduction as a variety of induction, and both have been modeled as variants on deduction. Thus, we will not rely heavily on this taxonomy in the pages that follow.

As we will see in the next section, many reasoning tasks focus on drawing factual or plausible *conclusions* from a set of initial facts and knowledge (Levesque & Brachmann, 1987). However, other varieties of problems, such as planning or knowledge-guided execution, instead emphasize *actions* that alter the agent's environment to achieve or maintain its *goals*. We can view these goals as types of queries that arise in action-oriented settings, because they play the same role of letting one know when reasoning has achieved its desired ends. The distinction is further blurred because we can use frameworks designed for conclusion-drawing tasks for action-oriented ones, and we can apply methods devised for the latter to address the former. These different framings are worth keeping in mind when reading the literature, as most papers adopt a particular perspective, but it is also important to remember the many central assumptions that they share. Neither are the two views always competitors, as some frameworks propose complementary roles for drawing conclusions and selecting actions (e.g., Choi & Langley, 2018; Laird, 2012).

4. Applications of Artificial Reasoning

We noted earlier that artificial reasoning differs from prediction and inference in that it relies centrally on composition or chaining of knowledge elements to draw conclusions or select actions. This distinction is most easily clarified by examining varieties of problems in which such multi-step processing arises. We will see that, despite many differences, each of these cognitive tasks relies on the combination of modular knowledge elements to produce coherent mental structures.

One classic use of artificial reasoning revolves around proving theorems. Here the knowledge elements are rules for mathematical or logical inference, the facts correspond to given assumptions, and the query is some conjecture or theorem we want to prove. A familiar example is proving a theorem in geometry (Gelernter et al., 1963). For instance, given a diagram with two triangles, A and B, given that certain angles and sides are equal, we may want to show that A and B are congruent. The result is a proof that uses geometric rules to derive this relation, step by step, from features of the diagram. Theorem proving has a long history in AI; the first implemented system, the Logic Theorist (Newell et al., 1957), proved theorems in propositional logic, and there has been considerable progress in the past 60 years. Applications of this technology include automated verification that software and hardware designs behave according to specifications, which is widely used in the computer industry.

Some other applications are closely related to theorem proving, in that they involve connecting a query with facts through some form of derivation. For instance, most efforts on relational database retrieval emphasize direct access to stored facts, but some methods can also derive answers through multi-step reasoning chains that connect high-level queries with low-level database facts. Similarly, systems for question answering, once they have transformed sentences into an internal representation, combine a number of knowledge elements to generate results. Research on automated design, especially for software, often starts from an abstract specification (similar to a query) that is iteratively refined to produce an operational artifact (e.g., Green & Barstow, 1975). The resulting derivations are similar to proof trees, although a key difference is that many of the 'facts' that serve as terminal nodes are not provided as inputs but must be introduced during the solution process.

Planning is another key application area that requires compositional reasoning, in this case to connect facts in an initial state to goals one desires to achieve (Ghallab et al., 2004). This differs from work on theorem proving and design, which typically involve monotonic reasoning steps that only add new facts, because planning systems usually reason over action-oriented rules that can both add and remove facts.[4] This has led the AI planning community to emphasize different methods for finding reasoning chains, but the end product remains a set of grounded knowledge elements (plans) that connect facts (the initial state) to a query (the goal description). Research on qualitative simulation has addressed similar issues to generate trajectories of situations that can follow from an initial situation (Forbus, 1984), although this typically occurs in the absence of goals, as it deals not with agent actions but with natural phenomena. Both planning and qualitative simulation have clear applications to military and civilian problems like logistics, which requires reasoning about many entities and activities that manipulate them.

Another important application area is understanding sentences in natural language, which relies on another variety of multi-step reasoning. Classical approaches from computational linguistics use grammatical knowledge, often stated as rewrite rules, to construct parse trees of sentences (Winograd, 1982). More recent work on statistical techniques often focuses on one-step inference for component tasks like part-of-speech tagging, but advanced systems combine these and other results into complex structures (e.g., Song et al., 2015). Another common class of methods, recurrent neural networks, operate in cycles and, in some sense, chain the results of successive steps (Elman, 1991). A different tradition aims not to generate parse trees but rather to extract meaning descriptions from sentences (e.g., Wilensky, 1978); these engage even more clearly in complex reasoning, as sentences often suggest information without stating it explicitly. Dialogue systems offer opportunities for substantial reasoning about other participants' beliefs and goals (e.g., Langley et al., 2014), even though recent research on the topic is dominated by 'chatbots' that rely mainly on shallow processing. Each of these has clear potential for military and civilian uses, from automated summarization to automated translation to conversational user interfaces.

A related topic concerns understanding stories and explaining observed activities. These are similar to language processing in that the aim is to interpret a set of events in terms of existing knowledge. The event descriptions may be encoded manually, but they can also be generated from sentence or video analysis. Typical explanations require the composition of a number of knowledge elements, with alternative accounts being evaluated in terms of plausibility. The resulting explanations may take the form of proof structures, but an essential difference is that they may include default assumptions which are neither present in the event description nor derived from other facts. This means that both story understanding and activity explanation involve abductive reasoning (e.g., Gordon, 2016; Meadows et al., 2014) rather than deduction. Explanation generation has important applications to assessing situations that involve many entities that change over time, including managing battlefield activities and monitoring power grids.

4. The situation calculus (McCarthy & Hayes, 1969) differs by transforming plan generation into a monotonic reasoning task, but this approach has not been widely adopted by planning researchers.

5. Paradigms for Artificial Reasoning

The AI community has developed a variety of approaches to artificial reasoning, as we have defined the field. In this section we review a number of such frameworks that differ in the specific reasoning task they address, their representations of knowledge and facts, or the mechanisms that operate over these structures.[5] We will refer to these alternatives as *paradigms* because they make different underlying assumptions, although it will become clear, as we discuss these approaches, that they also have many deep similarities.

The first paradigm, often known as *logic programming* (Lloyd, 1984), encodes knowledge as rules, each of which links a set of antecedents or conditions under which one can infer a consequent. Each antecedent or consequent is a relational literal, typically with variables, such as *on(X, Y)* or *believes(P, wants(Q, R))*. Facts take the same form, but usually are ground literals with only constants as arguments, such as *on(a, b)* or *believes(ann, wants(bob, on(c, d))*. Queries may be individual literals or conjunctions of them, but they often include variables, as in *believes(X, wants(Y, on(Z, d))*. Given such a query, a logic programming system attempts to find reasoning chains that connect it to given facts, with each organized as a proof tree that propagates variable bindings to an answer. If no such proof trees are found, then no answers are returned. Typical logic programming systems reason backward from the query using depth-first search, but this is not required. A more important assumption is that each proof must be deductively valid. PROLOG (Clocksin & Mellish, 1981) is a widely used environment for this variety of artificial reasoning.

A second paradigm, closely related to the first, is *answer set programming* (Baral, 2003). This encodes knowledge in a similar rule format, but it may also include rules with no consequent that serve as constraints on acceptable solutions by specifying a set of antecedents that may not cooccur. Facts take the same form as in logic programming, but queries are absent, as the aim here is not to answer directed questions but rather to find the set of possible worlds, or *answer sets*, that are consistent with the given facts. Each alternative world assigns a truth value to every possible fact that involves known predicates. Many facts that hold in a candidate world will share arguments, giving the semblance of a reasoning chain without an explicit proof tree that shows how some elements follow from others. A common technique for generating answer sets is depth-first search through the space of worlds with informed backtracking. CLASP (Gebser et al., 2007) is one widely used environment for answer set programming.

Research on *abductive reasoning* has similarities to both of the previous paradigms. Knowledge is typically stated as relational rules with a single consequent and multiple antecedents, while facts are encoded as ground literals.[6] Queries are typically absent, as the standard formulation involves constructing explanations of observed facts in terms of available knowledge. A common approach frames each explanation as a set of connected proof trees, with one root node for each fact and with terminal nodes corresponding to other facts or default assumptions. In another formulation, observed facts and default assumptions serve as terminal nodes, with each root node being unobserved but derivable from them. In both cases, explanations take the same form as proofs in logic

5. Our treatment focuses on approaches that operate over relational representations. Thus, it ignores some well-known frameworks like *constraint satisfaction* (Tsang, 1993) that usually assume more restricted notations.
6. We will not cover approaches to abduction like Bayesian networks that assume attribute-value or featural representations, although later we discuss their extension to relational notations.

programming, in that they would be deductively valid if their terminal elements were known to hold. A typical search technique adapts the backward chaining used in logic programming (e.g., Gordon, 2016), but forward chaining is also possible (e.g., Meadows, Langley, & Emery, 2014), and the possible worlds framework of answer set programming offers another path to abduction.

A fourth paradigm, *analogical reasoning*, has much in common with abduction in that it focuses on drawing plausible conclusions rather than deductively valid ones. Like answer set programming, it uses background knowledge to interpret a set of relational facts. The primary differences are that this knowledge takes the form of ground literals rather than generic rules and that they are organized into *cases*, which are sets of facts connected through shared arguments. The result of analogical reasoning is one or more mappings or analogies between these new facts and the elements of a case. Mappings are typically partial, in that some case elements have no corresponding facts and vice versa. The former set, case elements not mapped onto facts, play the same role as default assumptions in abductive approaches, in that they can be plausibly inferred. Analogical reasoning can produce proof trees, but only if the stored cases take this form. Search for analogies often involves constrained heuristic search that eliminates inconsistent mappings and favors high-level relations over low-level ones. Perhaps the best-known software for analogical reasoning is the Structure-Mapping Engine (Falkenhainer, Forbus, & Gentner, 1989).

Another paradigm, *production systems*, has been widely used both in models of human cognition (Anderson& Lebiere, 1998; Klahr et al., 1987) and in expert systems (Waterman, 1986). This framework adopts an action-oriented metaphor. Knowledge comprises a set of productions, or condition-effect rules, where the conditions are a set of generic relational literals and the effects specify addition or deletion of matched elements. A dynamic working memory stores facts that may describe an agent's beliefs, goals, or queries. A production system operates in cycles, in each case finding rules whose conditions match against elements in working memory, selecting one of them to apply, and using their effects to alter working memory. OPS5 (Brownston et al., 1985) and CLIPS (Giarratano & Riley, 2018) are two well-known programming environments for production systems. AI systems for plan generation also encode knowledge in terms of condition-effect rules, although their details of operation differ substantially. Recurrent neural networks (Elman, 1991), which also operate in recognize-act cycles, are very similar in spirit, although they operate over finer-grained representations and propagate changes by activations rather than additions and deletions.

Statistical relational processing constitutes a sixth reasoning paradigm (De Raedt et al., 2016). The framework associates conditional probabilities or similar weights with rules, logical formulae, or similar structures. Facts and optional queries take the standard form of relational literals. Statistical relational systems that accept queries return a set of answers, each with posterior probabilities and reasoning chains that connect them to given facts. Non-query approaches instead return a set of possible worlds, as in answer set programming, each with a posterior probability given the facts. Some systems adapt the backward-chaining search methods common in logic programming, calculating probabilities for each inferred literal upon deriving it. Other systems instead rely on gradient-descent search through the the space of possible worlds (i.e., truth assignments for each possible ground fact) to find high-probability candidates, using random restarts to avoid local optima. PROBLOG (De Raedt et al., 2007) and ALCHEMY (Domingos et al., 2016) are two well-known software environments for statistical relational processing.

A final paradigm operates over distributed encodings of relational content. Rather than using logical or logic-like formalisms to represent knowledge, facts, and queries, this framework denotes them with finer-grained notations. The most widely studied examples are *multi-layer neural networks*, which store knowledge as weighted links between nodes and facts or queries as activations on those nodes. An alternative approach, *word embeddings*, instead maps symbols and symbol structures into a low-dimensional vector space based on cooccurrence information. Research on multi-step reasoning is rare in this paradigm. Although neural networks propagate activation through many levels, they are typically used for classification and regression tasks, and thus are better viewed as inference systems. However, recent work has demonstrated that the framework can support complicated reasoning. For example, Hudson and Manning (2018) report a recurrent neural network that carries out a sequence of reasoning steps to answer questions about complex images. Summers-Stay (2017) describes a very different scheme that encodes logical rules, facts, and queries in a word embedding, then uses LASSO, a gradient-descent method for variable selection, to iteratively find a reasoning chain that connects queries to facts. Multi-step reasoning in this paradigm remains relatively unexplored, but these initial results suggest it has considerable potential.

In summary, researchers have developed and explored a variety of distinct paradigms for artificial reasoning. Closer analysis suggests that these differ along three main dimensions. One is whether they operate over generic knowledge structures (as in logic programming and production systems), encode knowledge with ground elements (as in analogy), or translate the former into the latter (as in many implementations of answer set programming and statistical relational processing). Another difference concerns granularity of representation. Every reasoning paradigm uses some form of distributed notation, but most frameworks adopt rule-like structures for knowledge and simple relational statements for facts, whereas others use finer-grained schemes like neural networks and word embeddings. A final dimension relates to the nature of the space searched to find reasoning chains. Most paradigms carry out search through a space of partial reasoning chains, with operators adding one reasoning step at a time; logic and answer set programming, production systems, and most methods for abductive reasoning take this approach. Other frameworks instead search through a space of complete candidates, with 'repair' operators transforming one possible solution into an alternative; analogical and statistical relational schemes take this approach, as does some work with distributed encodings.[7] Despite these differences, all paradigms tackle the same problem: finding reasoning chains that connect knowledge elements to facts in an organized manner.

6. Challenges for Artificial Reasoning

We have seen that systems for artificial reasoning must encode knowledge, facts, and reasoning chains, as well as provide mechanisms that use the former to construct the latter. However, many real-world settings introduce complicating factors that merit separate discussion. Many of these challenges revolve around issues of representation, although they also have processing implications.

The first challenge involves reasoning about *time*. Continuous mathematics addresses temporal phenomena with differential equations, but humans typically reason about time in qualitative terms. This has produced notations like the situation calculus (McCarthy & Hayes, 1969), which encodes

7. The AI literature sometimes refers to this distinction with the terms 'global search' and 'local search', but neither of these labels seems especially descriptive.

situations as conjunctions of ground facts, each with an identifier that may have successors and predecessors.[8] The event calculus (Kowalski, 1992) adopts a finer-grained approach that associates time stamps not with situations but with individual facts. Another approach (Allen, 1983) defines temporal relations among facts about events, like *before* and *during*, that assumes they hold for intervals without specifying particular start and end times. The SNARK environment (Stickel et al., 2000) adopts a third scheme that defines intervals in terms of start and end times, which it can associate with events. Of course, schedules also adopt an interval notation that specifies when each of a number of activities will begin and end. Given these representational extensions, one can use a variety of reasoning methods to draw nontrivial conclusions about temporal relations, higher-level events, and the like. These have obvious applications in many settings, from understanding complex temporal events, such as watching a baseball game, to generating plans that involve coordinated activities, say for military missions that require groups to move simultaneously.

Another challenge concerns reasoning about *space*. This raises similar issues to those for time, but involves two or three dimensions rather than one. Again, humans often reason about space in qualitative terms, rather than with quantitative frameworks like partial differential equations. Some researchers (e.g., Cohn et al., 1997) have adapted temporal formalisms to encode relations among objects and regions with spatial extent, such as *disconnected from* and *overlapping*. Diagrams offer a visual notation that, when internalized, can also encode qualitative spatial relationships (e.g., Forbus et al., 1991). These frameworks support spatial reasoning that can make deductions, answer queries, and generate explanations. In contrast, most work in mobile robotics and geographic information systems uses a grid-based representation for spatial information that makes high-level reasoning more difficult. One exception is Kuipers' (2000) spatial semantic hierarchy, which connects abstract qualitative encoding of space with low-level robotic sense data. Applications of spatial reasoning also abound. Autonomous vehicles must take distances and terrain into account when selecting routes, while military planners must think about ways to divide or envelop enemy forces.

A third issue involves reasoning about *causal influences*, that is, the conditions under which some factors affect others. For instance, research on qualitative physics (e.g., Forbus, 1984) adopts qualitative versions of algebraic or differential equations that state how the values of some continuous variables increase or decrease with changes in others. Alternative paradigms like Boolean networks (Glass & Kauffman, 1973) and most Bayesian networks (Darwiche, 2009) instead assume that attributes are discrete[9] and specify mappings from combinations of some attributes' values to those for others. A third scheme, associated with planning, emphasizes relational rules that describe the conditions under which actions will have particular effects (Ghallab et al., 2004). Each approach encodes causal knowledge as a set of modular elements that one can compose to construct causal chains to simulate possible trajectories or 'envisionments' from some initial state; these may or may not involve intentional actions by goal-directed agents. They also support counterfactual reasoning about alternative worlds. Causal reasoning has a role in many AI applications, from self-driving cars to diagnostic assistants to logistics planners, all of which can benefit from considering the effects produced by different actions, whether for predicting the future or explaining the past.

8. Planning systems, finite-state machines, and partially observable Markov decision processes also place a total or partial order on actions that have implications for state descriptions.
9. In some cases, each symbolic value denotes a discretized range of some continuous variable. For instance, the predicates *to-left* and *to-right* might refer to whether the angular difference between objects is positive or negative.

Another complicating factor is *uncertainty*, and AI researchers have explored many approaches to encoding such content. We will not review them all here, but the most common framework assigns probabilities to facts and associates conditional probabilities with combinations of discrete values, as in Bayesian networks (Darwiche, 2009) or with rules (e.g., De Raedt et al., 2007). This numeric information serves as annotations on symbolic structures. Reasoning with uncertainty is challenging because it retains the combinatorial complexity of classical reasoning with an additional need to propagate probabilities or their analogs along reasoning chains. Exact methods for calculating posterior probabilities from givens are computationally intractable for complex structures, leading to widespread use of sampling techniques to estimate values. However, even these place a substantial burden in addition to the standard costs that arise when answering questions, constructing plans, or generating explanations. This holds especially for reasoning with uncertainty over relational representations, so that many efforts remain restricted to less powerful notations. Uncertainty arises in many applied settings, not only on perceptual tasks like image interpretation and speech recognition, but on problems like medical diagnosis, where knowledge is incomplete, and in physical control, where the results of actions can be unpredictable.

A fifth challenge concerns *normative* reasoning about which actions an agent should or should not take. This arises mainly in plan generation and activity understanding, both of which focus on goal-directed behavior, but they move beyond goals to address social norms, which include formal laws, informal customs, and even moral tenets. Normative reasoning, whether focused on generating an agent's activity or understanding another's behavior, requires that one represent such limits on behavior and take them into account during the reasoning process. Although norms may appear, at first glance, to constitute hard constraints, in practice they often conflict, which requires reasoning about relative priorities and tradeoffs. Research on legal reasoning (e.g., Branting, 2000) and moral cognition (e.g., Iba & Langley, 2011; Malle et al., 2015; Mikhail, 2007) has addressed some issues, but we need more work in the area, which has great relevance for self-driving vehicles, military robots, and autonomous systems in general.

A sixth issue revolves around reasoning about other agents. This is related to uncertainty because one cannot always predict reliably how others will behave, but probabilistic methods offer only one response. An earlier approach, still widely used in adversarial settings, draws on 'game trees' that alternate between considering a primary agent's possible actions, which are under its control, and alternative responses by another agent, which are not. The game-playing community has devised specialized search methods, such as alpha-beta pruning, to deal with such scenarios. Another challenge concerns representing and reasoning about the mental states of other agents. This arises not only in competitive scenarios like games but also in cooperative ones like human-robot interaction. Here the reasoner must encode ground facts like *believes(Abe, wants(Bob, holding(Abe, hammer)))*, as well as operators for communicative acts that alter mental states, which have obvious applications in dialogue systems. Such reasoning becomes especially challenging when one agent attempts to deceive another to achieve its objectives, but systems must also ensure that content from one context (e.g., Abe's beliefs about Bob) do not carry over to others (e.g., Bob's beliefs).

A final challenge deals not with representational issues but with the nature of processing. Most work on artificial reasoning assumes that all facts and knowledge elements are available at the outset. This means that one can construct explanations, generate plans, and the like only once, without needing to revisit the reasoning chains that support them. However, in some settings the content, especially facts, arrive in small increments over time, so that conclusions based on earlier, incomplete, information may now be incorrect, incomplete, or compare poorly to low-ranked alternatives. Methods for *belief revision* address this problem, typically by storing dependencies among steps in reasoning chains, identifying which ones cease to hold when new facts arrive, and revise mental structures in response. Truth-maintenance techniques (e.g., de Kleer, 1986; Doyle, 1979) are the best-known examples, but others include production systems (e.g., Forgy, 1982) and frameworks that monitor plan execution (e.g., Langley et al., 2017). These take into account new information and revise prior reasoning steps as needed, supporting what is often called *nonmonotonic reasoning* (Ginsberg et al., 1987). The ability to revise beliefs is important to any setting in which information arrives incrementally, from robots that integrate planning, execution, and monitoring, to computational aids for situation assessment that update their models as observations become available.

7. Interactive Reasoning Systems

The vast majority of work on artificial reasoning has focused on automated methods, with humans only providing the inputs and interpreting the outputs. However, there are many settings where people desire aids rather than replacements, and other cases where they can help offset the limitations of artificial systems. Fortunately, the task at hand involves finding *chains* of knowledge elements rather than isolated inferences, and this stepwise chaining process lends itself to interactive approaches. Moreover, the steps often map onto conscious decisions in human reasoning, in which case they are reasonably accessible and interpretable. People sometimes engage in unconscious multi-step inference, especially during language use, but this does not negate the many contexts in which they are aware of their reasoning steps. Designing interactive systems for reasoning is far more natural and straightforward than for low-level tasks like computer vision or robot control.

Because multi-step reasoning involves search through a space of proofs, plans, or explanations, a natural approach to interaction is to let people influence each step of this process. Such search depends on both *generation* of reasoning steps and *selection* of which one to pursue. Thus, one interaction design would have the system generate candidate steps (e.g., rule instances) and have a human choose among them, avoiding the need for automated selection criteria. In another approach, the person would propose a number of possible reasoning steps, sidestepping the need for a knowledge base, while the system decides which one to apply. These are two extreme examples, but more plausible designs would have the artificial system generate and evaluate choices, but also let a human add or remove the alternatives or reorder the candidates. However, each of these schemes assumes fine-grained cooperation with the system, which can be effortful and tedious for users.

An alternative framework would let people interact with artificial reasoners at a more abstract level. For example, in many cases the evaluation metric used to rank candidate solutions can be decomposed into a number of weighted criteria. Once the computational system has found a number of solutions, the user can inspect them and then ask it to find one that is better along one of the dimensions. Upon seeing the result, he may request another improvement, and so forth, providing high-level guidance while letting the system handle the details of search. Rogers et al. (1999) have reported an interactive route planner that operates in this manner, while Gervasio et al. (1998) have taken a similar approach to interactive scheduling. Cox and Veloso (1998) have described a different interaction design in which the user influences an automated planner's behavior by altering the top-level goals it attempts to achieve. Such high-level inputs seem likely to be more effective in cases with complex reasoning chains, because giving users detailed control would be impractical.

Another promising approach would let users critique solutions that an artificial reasoner has generated and propose revisions, with the system checking these proposals to ensure they are correct. Humans might also insert missing reasoning steps to repair errors of omission that are due to incomplete system knowledge. Both types of interaction would benefit if the system could *explain* its reasons for considering, favoring, or rejecting solutions when asked.[10] Langley (2019) has argued that this requires an episodic memory which stores decisions made during search, including the choices considered and the criteria used to select among them. To answer questions about its reasoning, the system must index these decisions in memory, retrieve relevant content on request, and convey this content to users in terms they can understand. Despite some research on explainable reasoning (e.g., Fox et al., 2017; Johnson, 1994; Swartout & Moore, 1993; van Lent et al., 2004), this topic deserves far more attention.

Although there has been substantial work on interactive decision aids, the great majority of efforts to date have dealt with simple choice or ranking tasks like those associated with recommender systems. Here the only 'search' involved is finding high-scoring items in a database, so the combinatorial explosion usually associated with problems that require multi-step composition of knowledge elements does not arise. Successful applications on these simpler tasks suggest the field of artificial reasoning would benefit from devoting more resources to the design and creation of interactive approaches that support the natural synergies between humans and machines, building on the strengths of each one and alleviating their weaknesses.

8. Evaluating Artificial Reasoning Systems

The diversity of approaches to artificial reasoning makes it important to gain scientific insight into their behavior, including the strengths and limitations under different conditions. Formal analyses offer one path to understanding, and this has played an important role in the literatures on logic programming and answer set programming. However, Simon (1968) has argued that, for sufficiently complex artifacts, the only viable approach is to study them empirically, using the same methods that have revealed so many insights into the behavior of natural systems. To this end, we should discuss what types of variables arise in experiments on systems that exhibit artificial reasoning and what types of hypotheses they can let us test.

As noted in Section 2, promising dependent measures include the *correctness* and *completeness* of reasoning processes; these are related closely to metrics for *precision* and *recall* in information retrieval. These address a reasoning system's accuracy, but we may also be interested in the *quality* of solutions that it generates, such as the length of proofs, the cost of plans, or the coherence of explanations. Another natural dependent variable is a system's *efficiency*, as indicated by the time it takes to complete a reasoning task. These measures are useful not only in evaluating performance, but in learning and transfer studies, where it becomes important to record higher-order variables. For example, the *learning rate* describes how a system's performance, however we measure it, improves with the number of training problems encountered (Kibler & Langley, 1988).

10. This differs from explaining other agents' activities, where the system lacks direct access to its own reasoning traces.

Controlled experiments let us examine how varying independent factors influences dependent measures of system performance. The most common studies of this sort compare the behavior of complete systems on a standard set of test problems, but these seldom offer insights into the reasons for observed differences. *Lesion studies* are more informative, as they examine the effect of removing some of a system's components to clarify the sources of its power, as are *parametric experiments*, which examine sensitivity to a technique's parameter settings. Systematic studies also let us measure how an artifact responds to changes in problem and domain characteristics. For instance, we can vary the size of knowledge base, length of solutions, and complexity of queries to determine how an approach scales on these complicating factors. We can also control the resources, such as time and memory, allotted for finding solutions to examine how this degrades behavior.

Although researchers can run controlled experiments for exploratory purposes, they are best motivated by specific hypotheses. For instance, we might predict that adding a new heuristic to guide search will improve a system's scalability, in terms of solution completeness, to larger knowledge bases. Similarly, we might hypothesize that one system will degrade more gracefully than another, in terms of solution quality, with increased restrictions on processing time. Systematic studies of these sorts can answer interesting scientific questions about sources of power in artificial reasoning and tradeoffs among different approaches, which are important because it is unlikely that one method will always fare better than others. Experiments should also report average results over random samples, rather than individual numbers, to guard against unreliable findings, and they should look for *substantial* differences among experimental conditions, rather than ones that are significant statistically but nevertheless small in magnitude.

9. Concluding Remarks

In this essay, we examined artificial reasoning from a number of different perspectives. We defined the generic task in terms of inputs – knowledge, facts, and queries – and outputs – conclusions and reasoning chains that support them – that distinguish it from one-step inference problems. We also noted four layers that arise in the study of reasoning – representation, performance, learning, and transfer – with later facets building on earlier ones. Next we discussed applications of artificial reasoning, such as proving theorems, answering questions, designing artifacts, generating plans, simulating trajectories, understanding language, and explaining activities. We reviewed different paradigms for reasoning: logic and answer set programming, analogical and abductive reasoning, production systems, statistical relational processing, and reasoning over distributed encodings. After this, we considered factors that make reasoning difficult: space, time, causation, uncertainty, norms, other agents, and revisions. In closing, we discussed the benefits of interactive reasoning systems, along with methods for evaluating artificial reasoners through controlled experiments.

Artificial reasoning systems have great potential to aid decision making on both civilian and military tasks that require the generation of complex inference chains. Such systems have existed since the earliest days of artificial intelligence, but they have not yet seen as widespread use as computational techniques for classification and prediction for an obvious reason – they address more challenging problems. Thus, we need increased research on this important topic, especially work that is driven by realistic scenarios and that stretches the representational and processing capabilities of current methods. To this end, researchers should draw freely on each of the theoretical paradigms we have reviewed and combine elements from them as necessary. We should also devote substantially more resources to interactive approaches that combine the distinctive strengths of humans and machines. Together, these efforts should let us develop computational artifacts that exhibit the full potential of artificial reasoning systems.

Acknowledgements

Preparation of this essay was supported by funding from the Army Research Laboratory, which is not responsible for its contents. We thank Paul Bello and Stephanie Sage for useful feedback that helped refine both the ideas in this paper and their exposition.

References

Allen, J. F. (1983). Maintaining knowledge about temporal intervals. *Communications of the ACM*, *26*, November, 832–843.

Anderson, J. R., & Lebiere, C. (1998). *The atomic components of thought*. Mahwah, NJ: Lawrence Erlbaum.

Baral, C. (2003). *Knowledge representation, reasoning and declarative problem solving*. Cambridge, UK: Cambridge University Press.

Branting, L. K. (2000). *Reasoning with rules and precedents: A computational model of legal analysis*. Dordrecht, Netherlands: Kluwer.

Brownston, L., Farrell, R., Kant, E., & Martin, N. (1985). *Programming expert systems in OPS5*. Boston: Addison-Wesley.

Choi, D., & Langley, P. (2018). Evolution of the ICARUS cognitive architecture. *Cognitive Systems Research*, *48*, 25–38.

Clocksin, W. F. & Mellish, C. S. (1981). *Programming in Prolog*. Berlin, Germany: Springer-Verlag.

Cohn, A. G. Bennett, B., Gooday, J., & Gotts, N. (1997). RCC: A calculus for region based qualitative spatial reasoning. *GeoInformatica*, *1*, 275–316.

Cox, M. T. & Veloso, M. M. (1998). Goal transformations in continuous planning. *Proceedings of the 1998 AAAI Fall Symposium on Distributed Continual Planning* (pp. 23–30). Arlington, VA: AAAI Press.

Darwiche, A. (2009). *Modeling and reasoning with Bayesian networks*. New York, NY: Cambridge University Press.

de Kleer, J. (1986). An assumption-based TMS. *Artificial Intelligence*, *28*, 127–162.

De Raedt, L., Kersting, K., Natarajan, S., & Poole, D. (2016). *Statistical relational artificial intelligence: Logic, probability, and computation*. San Rafael, CA: Morgan & Claypool.

De Raedt, L., Kimmig, A., & Toivonen, H. (2007). PROBLOG: A probabilistic PROLOG and its application in link discovery. *Proceedings of the Twentieth International Joint Conference on Artificial Intelligence* (pp. 2462–2467). Hyderabad, India: Morgan Kaufmann.

Domingos, P., Lowd, D., Kok, S., Nath, A., Poon, H., Richardson, M., & Singla, P. (2016). Unifying logical and statistical AI. *Proceedings of the Thirty-First Annual ACM/IEEE Symposium on Logic in Computer Science* (pp. 1–11). New York, NY: ACM.

Doyle, J. (1979). A truth maintenance system. *Artificial Intelligence*, *12*, 251–272.

Elman, J. L. (1991). Distributed representations, simple recurrent networks, and grammatical structure. *Machine Learning*, *7*, 195–225.

Falkenhainer, B., Forbus, K., & Gentner, D. (1989). The Structure-Mapping Engine: Algorithm and examples. *Artificial Intelligence*, *20*, 1–63.

Forbus, K. D. (1984). Qualitative process theory. *Artificial Intelligence*, *24*, 85–168.

Forbus, K., Nielsen, P., & Faltings, B. (1991). Qualitative spatial reasoning: The CLOCK project. *Artificial Intelligence*, *51*, 417–471.

Forgy, C. (1982). Rete: A fast algorithm for the many pattern/many object pattern match problem. *Artificial Intelligence*, *19*, 17–37.

Fox, M., Long, D., & Magazzeni, D. (2017). Explainable planning. *Proceedings of the IJCAI-17 Workshop on Explainable AI* (pp. 24–30). Melbourne, Australia.

Gebser, M., Kaufmann, B., Neumann, A., & Schaub, T. (2007). CLASP: A conflict-driven answer set solver. *Proceedings of the Ninth International Conference on Logic Programming and Nonmonotonic Reasoning* (pp. 260–265). Tempe, AZ: Springer.

Gelernter, H., Hansen, J. R., & Loveland, D. W. (1963). Empirical explorations of the geometry-theorem proving machine. In E. A. Feigenbaum & J. Feldman (Eds.), *Computers and thought*. New York, NY: McGraw-Hill.

Gentner, D., & Forbus, K. (1991). MAC/FAC: A model of similarity-based retrieval. *Proceedings of the Thirteenth Annual Conference of the Cognitive Science Society* (pp. 504–509). Chicago, IL: Lawrence Erlbaum.

Gervasio, M. T., Iba, W., & Langley, P. (1999). Learning user evaluation functions for adaptive scheduling assistance. *Proceedings of the Sixteenth International Conference on Machine Learning* (pp. 152–161). Bled, Slovenia: Morgan Kaufmann.

Ghallab, M., Nau, D., Traverso, P. (2004). *Automated planning: Theory and practice*. San Francisco, CA: Morgan Kaufmann.

Giarratano, J. C., & Riley, G. D. (2018). *Expert systems: Principles and programming* (4th Ed.). Pacific Grove, CA: Brooks-Cole Publishing.

Ginsberg, M. L. (Ed.) (1987). *Readings in nonmonotonic reasoning*. San Francisco, CA: Morgan Kaufmann.

Glass, L., & Kauffman, S. (1973). The logical analysis of continuous, nonlinear biochemical control networks. *Journal of Theoretical Biology*, *39*, 103–129.

Gordon, A. (2016). Commonsense interpretation of triangle behavior. *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence* (pp. 3719–3725). Phoenix, AZ: AAAI Press.

Green, C. C., & David R. Barstow, D. R. (1975). Some rules for the automatic synthesis of programs. *Proceedings of the Fourth International Joint Conference on Artificial Intelligence* (pp. 232–239). Tbilisi, Georgia.

Hudson, D. A. & Manning, C. D. (2018). Compositional attention networks and machine reasoning. *Proceedings of the Sixth International Conference on Learning Representations*. Vancouver, BC.

Iba, G. A. (1989). A heuristic approach to the discovery of macro-operators. *Machine Learning*, *3*, 285–317.

Iba, W. F., & Langley, P. (2011). Exploring moral reasoning in a cognitive architecture. *Proceedings of the Thirty-Third Annual Meeting of the Cognitive Science Society*. Boston, MA.

Johnson, W. (1994). Agents that learn to explain themselves. *Proceedings of the Twelfth National Conference on Artificial Intelligence* (pp. 1257–1263). Seattle, WA: AAAI Press.

Kibler, D., & Langley, P. (1988). Machine learning as an experimental science. *Proceedings of the Third European Working Session on Learning* (pp. 81–92). Glasgow, UK: Pittman.

Klahr, D., Langley, P., & Neches, R. (Eds.) (1987). *Production system models of learning and development*. Cambridge, MA: MIT Press.

Kowalski, R. (1992). Database updates in the event calculus. *Journal of Logic Programming*, *12*, 121–146.

Kuipers, B. (2000). The spatial semantic hierarchy. *Artificial Intelligence*, *119*, 191–233.

Laird, J. E. (2012). *The Soar cognitive architecture*. Cambridge, MA: MIT Press.

Langley, P. (1996). *Elements of machine learning*. San Francisco, CA: Morgan Kaufmann.

Langley, P. (2019). Explainable, normative, and justified agency. *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence*. Honolulu, HI: AAAI Press.

Langley, P., Choi, D., Barley, M., Meadows, B., & Katz, E. P. (2017). Generating, executing, and monitoring plans with goal-based utilities in continuous domains. *Proceedings of the Fifth Annual Conference on Cognitive Systems*. Troy, NY: Cognitive Systems Foundation.

Langley, P., Meadows, B., Gabaldon, A., & Heald, R. (2014). Abductive understanding of dialogues about joint activities. *Interaction Studies*, *15*, 426–454.

Levesque, H. J., & Brachmann, R. J. (1987). Expressiveness and tractability in knowledge representation and reasoning. *Computational Intelligence*, *3*, 78–93.

Lloyd, J. W. (1984). *Foundations of logic programming*. Berlin, Germany: Springer-Verlag.

Malle, B. F., Scheutz, M., & Austerweil, J. L. (2015). Networks of social and moral norms in human and robot agents. *Proceedings of the International Conference on Robot Ethics* (pp. 3–17). Lisbon, Portugal.

McCarthy, J., & Hayes, P. (1969). Some philosophical problems from the standpoint of artificial intelligence. In B. Meltzer & D. Michie (Eds.), *Machine intelligence* (Vol. 4). Edinburgh, UK: Edinburgh University Press.

Meadows, B., Langley, P., & Emery, M. (2014). An abductive approach to understanding social interactions. *Advances in Cognitive Systems*, *3*, 87–106.

Mikhail, J. (2007). Universal moral grammar: Theory, evidence and the future. *Trends in Cognitive Science*, *11*, 143–152.

Newell, A., & Simon, H. A. (1976). Computer science as empirical enquiry: Symbols and search. *Communications of the ACM*, *19*, 113–126.

Newell, A., Shaw, J. C., & Simon, H. A. (1957). Empirical explorations of the Logic Theory Machine. A case study in heuristic. *Proceedings of the Western Joint Computer Conference* (pp. 218–230) New York, NY: Institute of Radio Engineers.

Robinson, J. A., & Voronkov, A. (Eds.). (2001). *Handbook of automated reasoning* (Vols. 1 and 2). Cambridge, MA: MIT Press.

Rogers, S., Fiechter, C., & Langley, P. (1999). An adaptive interactive agent for route advice. *Proceedings of the Third International Conference on Autonomous Agents* (pp. 198–205). Seattle, WA: ACM Press.

Simon, H. A. (1968). *The sciences of the artificial*. Cambridge, MA: MIT Press.

Sleeman, D., Langley, P., & Mitchell, T. (1982). Learning from solution paths: An approach to the credit assignment problem. *AI Magazine*, *3*, 48–52.

Song, Y., Peng, H., Kordjamshidi, P., Sammons, M., & Roth, D. (2015). Improving a pipeline architecture for shallow discourse parsing. *Proceedings of the SIGNLL Conference on Computational Natural Language Learning*. Beijing, China: ACL.

Stickel, M. E., Waldinger, R. J., & Chaudhri, V. K. (2000). *A guide to SNARK* (Technical Report). AI Center, SRI International, Menlo Park, CA.

Summers-Stay, D. (2017). Deductive and analogical reasoning on a semantically embedded knowledge graph. *Proceedings of the Tenth International Conference on Artificial General Intelligence* (pp. 112–122). Melbourne, Australia.

Swartout, W. R., & Moore, J. D. (1993). Explanation in second generation expert systems. In J.-M. David, J.-P. Krivine, & R. Simmons (Eds.), *Second generation expert systems*. Berlin: Springer-Verlag.

Tsang, E. (1993). *Foundations of constraint satisfaction*. London, UK: Academic Press.

van Lent, M., Fisher, W., & Mancuso, M. (2004). An explainable artificial intelligence system for small-unit tactical behavior. *Proceedings of the Nineteenth National Conference on Artificial Intelligence* (pp. 900–907). San Jose, CA: AAAI Press.

Waterman, D. A. (1986). *A guide to expert systems*. Reading, MA: Addison-Wesley.

Wilensky, R. (1978). Why John married Mary: Understanding stories involving recurring goals. *Cognitive Science*, *2*, 235–266.

Winograd, T. (1982). *Language as a cognitive process: Syntax*. Boston, MA: Addison-Wesley.

Wos, L., Overbeek, R., Lusk, E., & Boyle, J. (1984). *Automated reasoning: Introduction and applications*. Englewood Cliffs, NJ: Prentice-Hall.