Reasoning system

From Wikipedia, the free encyclopedia

In information technology a reasoning system is a software system that generates conclusions from available knowledge using logical techniques such as deduction and induction. Reasoning systems play an important role in the implementation of artificial intelligence and knowledge-based systems.

By the everyday usage definition of the phrase, all computer systems are reasoning systems in that they all automate some type of logic or decision. In typical use in the Information Technology field however, the phrase is usually reserved for systems that perform more complex kinds of reasoning. For example, not for systems that do fairly straight forward types of reasoning such as calculating a sales tax or customer discount but making logical inferences about a medical diagnosis or mathematical theorem. Reasoning systems come in two modes: interactive and batch processing. Interactive systems interface with the user to ask clarifying questions or otherwise allow the user to guide the reasoning process. Batch systems take in all the available information at once and generate the best answer possible without user feedback or guidance.[1]

Reasoning systems have a wide field of application that includes scheduling, business rule processing, problem solving, complex event processing, intrusion detection, predictive analytics, robotics, computer vision, and natural language processing.

Contents

History

The first reasoning systems were theorem provers, systems that represent axioms and statements in First Order Logic and then use rules of logic such as modus ponens to infer new statements. Another early type of reasoning system were general problem solvers. These were systems such as the General Problem Solver designed by Newell and Simon. General problem solvers attempted to provide a generic planning engine that could represent and solve structured problems. They worked by decomposing problems into smaller more manageable sub-problems, solving each sub-problem and assembling the partial answers into one final answer. Another example general problem solver was the SOAR family of systems.

In practice these theorem provers and general problem solvers were seldom useful for practical applications and required specialized users with knowledge of logic to utilize. The first practical application of automated reasoning were expert systems. Expert systems focused on much more well defined domains than general problem solving such as medical diagnosis or analyzing faults in an aircraft. Expert systems also focused on more limited implementations of logic. Rather than attempting to implement the full range of logical expressions they typically focused on modus-ponens implemented via IF-THEN rules. Focusing on a specific domain and allowing only a restricted subset of logic improved the performance of such systems so that they were practical for use in the real world and not merely as research demonstrations as most previous automated reasoning systems had been. The engine used for automated reasoning in expert systems were typically called inference engines. Those used for more general logical inferencing are typically called theorem provers.[2]

With the rise in popularity of expert systems many new types of automated reasoning were applied to diverse problems in government and industry. Some such as case-based reasoning were off shoots of expert systems research. Others such as constraint satisfaction algorithms were also influenced by fields such as decision technology and linear programming. Also, a completely different approach, one not based on symbolic reasoning but on a connectionist model has also been extremely productive. This latter type of automated reasoning is especially well suited to pattern matching and signal detection types of problems such as text searching and face matching.

Use of logic

The term reasoning system can be used to apply to just about any kind of sophisticated decision system as illustrated by the specific areas described below. However, the most common use of the term reasoning system implies the computer representation of logic. Various implementations demonstrate significant variation in terms of systems of logic and formality. Most reasoning systems implement variations of propositional and symbolic (predicate) logic. These variations may be mathematically precise representations of formal logic systems (e.g., FOL), or extended and hybrid versions of those systems (e.g., Courteous logic[3]). Reasoning systems may explicitly implement additional logic types (e.g., modal, deontic, temporal logics). However, many reasoning systems implement imprecise and semi-formal approximations to recognised logic systems. These systems typically support a variety of procedural and semi-declarative techniques in order to model different reasoning strategies. They emphasise pragmatism over formality and may depend on custom extensions and attachments in order to solve real-world problems.

Many reasoning systems employ deductive reasoning to draw inferences from available knowledge. These inference engines support forward reasoning or backward reasoning to infer conclusions via modus ponens. The recursive reasoning methods they employ are termed ‘forward chaining’ and ‘backward chaining’, respectively. Although reasoning systems widely support deductive inference, some systems employ abductive, inductive, defeasible and other types of reasoning. Heuristics may also be employed to determine acceptable solutions to intractable problems.

Reasoning systems may employ the closed world assumption (CWA) or open world assumption (OWA). The OWA is often associated with ontological knowledge representation and the Semantic Web. Different systems exhibit a variety of approaches to negation. As well as logical or bitwise complement, systems may support existential forms of strong and weak negation including negation-as-failure and ‘inflationary’ negation (negation of non-ground atoms). Different reasoning systems may support monotonic or non-monotonic reasoning, stratification and other logical techniques.

Reasoning under uncertainty

Many reasoning systems provide capabilities for reasoning under uncertainty. This is important when building situated reasoning agents which must deal with uncertain representations of the world. There are several common approaches to handling uncertainty. These include the use of certainty factors, probabilistic methods such as Bayesian inference or Dempster–Shafer theory, multi-valued (‘fuzzy’) logic and various connectionist approaches.[4]

Types of reasoning system

This section provides a non-exhaustive and informal categorisation of common types of reasoning system. These categories are not absolute. They overlap to a significant degree and share a number of techniques, methods and algorithms.

Constraint solvers

Constraint solvers solve constraint satisfaction problems (CSPs). They support constraint programming. A constraint is a condition which must be met by any valid solution to a problem. Constraints are defined declaratively and applied to variables within given domains. Constraint solvers use search, backtracking and constraint propagation techniques to find solutions and determine optimal solutions. They may employ forms of linear and nonlinear programming. They are often used to perform optimization within highly combinatorial problem spaces. For example, they may be used to calculate optimal scheduling, design efficient integrated circuits or maximise productivity in a manufacturing process.[5]

Theorem provers

Theorem provers use automated reasoning techniques to determine proofs of mathematical theorems. They may also be used to verify existing proofs. In addition to academic use, typical applications of theorem provers include verification of the correctness of integrated circuits, software programs, engineering designs, etc.

Logic programs

Logic programs (LPs) are software programs written using programming languages whose primitives and expressions provide direct representations of constructs drawn from mathematical logic. An example of a general-purpose logic programming language is Prolog. LPs represent the direct application of logic programming to solve problems. Logic programming is characterised by highly declarative approaches based on formal logic, and has wide application across many disciplines.

Rule engines

Rule engines represent conditional logic as discrete rules. Rule sets can be managed and applied separately to other functionality. They have wide applicability across many domains. Many rule engines implement reasoning capabilities. A common approach is to implement production systems to support forward or backward chaining. Each rule (‘production’) binds a conjunction of predicate clauses to a list of executable actions. At run-time, the rule engine matches productions against facts and executes (‘fires’) the associated action list for each match. If those actions remove or modify any facts, or assert new facts, the engine immediately re-computes the set of matches. Rule engines are widely used to model and apply business rules, to control decision-making in automated processes and to enforce business and technical policies.

Deductive classifier

Deductive classifiers arose slightly later than rule-based systems and were a component of a new type of artificial intelligence knowledge representation tool known as frame languages. A frame language describes the problem domain as a set of classes, subclasses, and relations among the classes. It is similar to the object-oriented model. Unlike object-oriented models however, frame languages have a formal semantics based on first order logic. They utilize this semantics to provide input to the deductive classifier. The classifier in turn can analyze a given model (known as an ontology) and determine if the various relations described in the model are consistent. If the ontology is not consistent the classifier will highlight the declarations that are inconsistent. If the ontology is consistent the classifier can then do further reasoning and draw additional conclusions about the relations of the objects in the ontology. For example, it may determine that an object is actually a subclass or instance of additional classes as those described by the user. Classifier's are an important technology in analyzing the ontologies used to describe models in the Semantic web.[6] [7]

Machine learning systems

Machine learning systems evolve their behavior over time based on experience. This may involve reasoning over observed events or example data provided for training purposes. For example, machine learning systems may use inductive reasoning to generate hypotheses for observed facts. Learning systems search for generalised rules or functions that yield results in line with observations and then use these generalisations to control future behavior.

Case-based reasoning systems

Case-based reasoning (CBR) systems provide solutions to problems by analysing similarities to other problems for which known solutions already exist. They use analogical reasoning to infer solutions based on case histories. CBR systems are commonly used in customer/technical support and call centre scenarios and have applications in industrial manufacture, agriculture, medicine, law and many other areas.

Procedural reasoning systems

A procedural reasoning system (PRS) uses reasoning techniques to select plans from a procedural knowledge base. Each plan represents a course of action for achievement of a given goal. The PRS implements a belief-desire-intention model by reasoning over facts ('beliefs') to select appropriate plans ('intentions') for given goals ('desires'). Typical applications of PRS include management, monitoring and fault detection systems.

References

1. Wos, Larry; Owerbeek, Ross; Ewing, Lusk; Boyle, Jim (1984). Automated Reasoning: Introductions and Applications. Prentice Hall. p. 4. ISBN 0-13-054453-1.
2. Hayes-Roth, Frederick; Waterman, Donald; Lenat, Douglas (1983). Building Expert Systems. AddisonWesley. ISBN 0-201-10686-8.
3. Grosof, Benjamin N. (30 December 1997). "Courteous Logic Programs: Prioritized Conflict Handling For Rules" (Postscript). IBM Research Report. RC 20836 (92273).
4. Moses, Yoram; Vardi, Moshe Y; Fagin, Ronald; Halpern, Joseph Y (2003). Reasoning About

Knowledge. MIT Press. ISBN 978-0-262-56200-3.
5. Schalkoff, Robert (2011). Intelligent Systems: Principles, Paradigms and Pragmatics: Principles, Paradigms and Pragmatics. Jones & Bartlett Learning. ISBN 978-0-7637-8017-3.
6. MacGregor, Robert (June 1991). "Using a description classifier to enhance knowledge representation". IEEE Expert 6 (3): 41–46. doi:10.1109/64.87683. Retrieved 10 November 2013.
7. Berners-Lee, Tim; Hendler, James; Lassila, Ora (May 17, 2001). "The Semantic Web A new form of Web content that is meaningful to computers will unleash a revolution of new possibilities". Scientific American 284: 34–43. doi:10.1038/scientificamerican0501-34.

Retrieved from "https://en.wikipedia.org/w/index.php?title=Reasoning_system&oldid=702132673"

Categories: Deductive reasoning | Problem solving | Reasoning | Inductive reasoning | Cognitive architecture | Rule engines | Expert systems | Logic programming | Automated theorem proving | Constraint programming | Applied machine learning | Artificial intelligence

---