KNOWLEDGE BASE VERIFICATION: ISSUES AND APPROACHES
<Abstract>

Du Zhang

Department of Computer Science
California State University
Sacramento, CA 95819-6021
zhangd@csus.edu

Knowledge base (KB) is an important component of knowledge-based systems (KBS). Due to the fact that a KB is often built in an incremental and piecemeal fashion, potential errors may be inadvertently brought into it. KB verification refers to the process of ascertaining the correctness of a KB, which is an integral part of the effort to guarantee the overall correctness of a KBS. KB verification presents a problem of multiple dimensions. Though many verification tools and techniques have been developed or proposed, no coordinated effort has been made in establishing a standard for evaluation and comparison of those tools. It is our hope that the issues raised below and the summary of the existing approaches may serve as a steppingstone toward the convergence of a unified thoery on KB verification.

ISSUES

Knowledge Representation Dependency. The issue regards whether a verifier is designed to deal with KB of a particular knowledge representation formalism.

Domain Dependency. This issue has to do with whether metaknowledge about the problem domain is utilized in a verifier. Accordingly, there are domain dependent and domain independent KB verifications.

Verification Criteria. This issue relates to how to define various types of anomalies in a KB.

Flat-Model vs. Hierarchical-Model of KB. A KB may be viewed either as a flat structure or as a hierarchical structure, thus resulting in different verification strategies.

Monotonicity of KB. Whether a verifier is capable of handling nonmonotonic KB is another dimension of the KB verification.

Certainty Factors and Temporal Operators. The issue here is whether a verifier can detect anomalies in a KB containing certainty factors and/or temporal operators.

Detection-only vs. Detection-and-Correction. While most verifiers are capable of detecting errors, some are capable of detecting and correcting errors.

Exhaustive vs. Heuristic Checking. The use of some heuristic search techniques in the verification process can greatly reduce the computational effort.

Static vs. Dynamic Checking. Depending on whether a functional inference engine is needed during verification, there are static and dynamic checking.

Participation of Domain Expert in Detection. Some verifier requires the participation of domain expert in order to detect anomalies in a KB.

Mode of Operation. Whether a KB verifier can be invoked during each development phase, or only at the end of the development cycle, whether the knowledge engineer, or the domain expert, or the independent third party, or any combination of them will be in charge of the verification process, presents yet another issue that needs to be determined when designing the verifier.

Performance Measures. There is need to establish some performance measures for comparison of different verifiers.

APPROACHES

There have been different approaches proposed or developed for KB verification. The list of approaches below is by no means a complete one. An important point is that depending on how a KB is modeled, there are many different ways of verifying the desired properties the KB possesses.

Decision Table Approach. Decision tables provide a means of organizing and documenting a set of rules in a manner that allows easy inspection and analysis. In a decision table, conditions and actions of the rules are arranged such that the testing of a set of rules for conditions of ambiguity, redundancy and completeness can be easily facilitated.

Machine Learning Approach. The essential idea of machine learning based approach is to generate examples from the given KB by using some learning strategies and confirm the examples to verify its correctness.

Logical Approach. This approach is based on conducting some logical operations either directly on a KB or on an equivalent set of logic formulas of the KB to derive verification results.

Petri Net Approach. This approach consists primarily of two steps: translate a KB into a Petri net model, and analyze the Petri net model for properties that reflect the anomalies in the KB.

KB-Reduction Approach Under this approach, a KBS is viewed as an implicit partial function that takes all possible input sets as its domain and all possible sets of conclusions as its range. Each conclusion c is labeled by a minimal disjunctive normal form expression that represents all the possible minimal input sets which cause the KB to assert c. All the potential inconsistencies and redundancies in a KB can be detected by examining the intermediate and final results produced during the process of constructing the function.

Metaknowledge Approach. This approach is usually adopted in domain dependent verification. It advocates the use of metaknowledge, the knowledge about domain knowledge compiled in a KB, in verifying the correctness of a KB.

Graph Approach. The essential idea behind this approach is to treat the rule base as graph generators and to analyze the graphs produced by the generating functions for certain criteria that pinpoint anomalies in a KB.

Relational Approach. There are attributes used in rules of a KB. Each attribute may draw values from a domain. The union of domains of all the attributes can be considered as the attribute space. Since conditions in a rule may be expressed in terms of attribute-value pairs, an individual rule can be construed as a function whose domain is the attribute space and whose range is a subset of the attribute space. Once the concepts of attribute space and rule functions are in place, different relations between rule functions may be defined that reflect potential errors in a KB. Verifying the KB amounts to detecting those relations.

Refinement Approach KB refinement, which is an important aspect of knowledge acquisition and is characterized by the addition, deletion and alteration of rule-components in an existing KB, has been used to improve the robustness of KB. KB refinement and verification, to a great extent, share a common goal of converting an initial KB into a high performance KB. This approach is based on the following three steps: (1) selecting a database of cases with known conclusions, i.e., each case contains not only a record of the case observations but also a record of the domain expert's conclusion for the case; (2) conducting empirical analysis of rule behavior by gathering certain statistics concerning rule behavior with respect to the database of cases; (3) generating suggestions for rule refinements by the application of refinement heuristics that relate the statistical behavior and structural properties of rules to appropriate classes of rule refinements. In essence, the approach identifies potential errors in a KB through the case database, statistical concepts and heuristics.

Syntactic Inspection Approach. Syntactic inspection approach relies on analyzing the syntactic properties of a KB to detect potential errors. It therefore is dependent on the knowledge representation in nature. Syntax for a KB may include: (1) conventions on how rules and facts in a KB are formed, grouped and prioritized (rules can also be classified according to their status, such as active, inactive, triggered or fired); (2) attribute sets as well as their domains; (3) ranges of certainty factors to be used in rules as well as regulations on how certainty factors are propagated during inference; (4) declarations of some KB components such as final hypotheses, askable attributes or sets of illegal values for attributes. Once the syntactic conventions are in place, potential errors in a KB can be defined as cases exhibiting certain specific syntactic characteristics. Verification is therefore geared toward the detection of those cases.

Incoherence Detection Approach. The notion of KB coherence was proposed by Ayel and Laurent. For facts in a KB, there is usually a set of properties such as coherence constraints, arity of attributes, contradictory values and so forth. To indicate that all facts satisfy the properties, a predicate FC is introduced. A set of facts is coherent if and only if FC is true on it. Similarly, there is also a set of properties associated with rules of a KB. Rule properties include, among other things, internal coherence of a rule, redundancy/conflict/circular rules. Given another predicate RC, a set of rules is coherent if and only if RC is true on it. A KB is statically coherent if sets of rules and facts in the KB are coherent. A KB is dynamically coherent if there exists the coherence of all sets of facts that are deducible by using facts and rules in the KB. Verifying a KB is to detect incoherence in it and the process is treated as a heuristic search problem rather than a logical one, therefore circumventing the combinatorial explosion during verification.

Traditional Software Engineering Approach. There has been a whole host of verification and validation techniques in traditional software engineering arena. A recent trend is to adopt or modify the existing software engineering techniques for KB verification. Some recent attempts include the use of checklist approach, assertional approach and the object-oriented process specification approach.