Knauf, Rainer; Gonzalez, Avelino J.; Jantke, Klaus P. :

Towards Validation of Case-Based Systems

Zuerst erschienen in:
Proceedings of the twelfth international Forida AI Research Society conference : Orlando, Florida, 3 - 5 May 1999 (FLAIRS 1999) / ed. by Amruth N. Kumar ... - Menlo Park, Calif. : AAAI Press, 1999, S. 516-520
Referenz-Link AAAI:
http://www.aaai.org/Library/Conferences/FLAIRS/FLAIRS-1999/Abstracts/flairs99-092.html

Towards Validation of Case-Based Systems*

Rainer Knauf
Technical University of Ilmenau
Faculty of Computer Science and Automation
PO Box 10 05 65, 98684 Ilmenau, Germany
Rainer.Knauf@TheoInf.TU-Ilmenau.de

Avelino J. Gonzalez
Dept. of Electrical and Computer Engineering
University of Central Florida
Orlando, FL 32816-2450. USA
ajg@ece.engr.ucf.edu

Klaus P. Jantke
Meme Media Laboratory
Hokkaido University
Kita-13, Nishi-8, Kita-ku, Saporro 060. Japan
jantke@meme.hokudai.ac.jp

Abstract

The present paper is an attempt to apply the authors' experiences of their work in validation of rule based systems (cf. (Knauf, Abel, Jantke and Gonzalez 1998). (Knauf, Jantke, Abel and Philippow 1997), (Abel, Knauf, and Gonzalez 1996), (Abel and Gonzalez 1997b), (Herrmann, Jantke, and Knauf 1997), and (Jantke, Knauf and Abel 1997),e.g.) to case based systems.

The objective of this work is both to come up with a framework for validation of case based systems and to answer the question how the knowledge representation and the problem solving paradigm influences the validation technology of an AI system.

It turns out, that the general steps of test case validation (test case generation, test case experimentation, evaluation, validity assessment, and system refinement), which are performed in cycles, can and should be the same for both paradigms, but the realization of the particular steps seems to be different.

Most of the steps can be performed with less expenditure and less human support for case based systems than for rule based systems. Generally, the more a knowledge representation level is explicit and abstract, the more the validation of the system seems to be complicated.

Introduction

There is no doubt about the necessity of an integrated approach towards validation of complex systems. In (Wise and Wise 1993) e.g. the authors clearly point out, that the lack of verification and validation approaches becomes sooner or later the limiting factor in the application of complex systems.

In the topical literature there are several concepts of verification and validation. Here, we follow (Boehm 1984) and (O'Keefe and O'Leary 1993) and distinguish *verification* and *validation* by the two circumscriptions of *building the system right* and *building the right system*, respectively. This perspective is illustrated in some of the authors' earlyer papers ((Abel and Gonzalez 1997b), (Jantke, Knauf and Abel 1997), (Knauf, Jantke, Abel and Philippow 1997)) and is accepted by many other authors.

For AI systems, particularly, validation is more important than for other systems which can often be evaluated just by verification.

There are a few general approaches for validation of AI systems. The authors' favorite approach is outlined in (Jantke, Knauf and Abel 1997). It's based on the fundamentals described in (Knauf, Jantke, Abel and Philippow 1997), e.g. and it consists of 5 steps, namely:

1. Test case generation
   Generate and optimize a set of test input combinations (test data) that will simulate the inputs to be seen by the system in actual operation. We refer to the pairs (*test data, expected output*) as test cases.
2. Test case experimentation
   In earlyer approaches of the authors this step consists of exercising the resulting set of test cases (from step 1 above) by the intelligent system as well as by the one or more validating experts in order to obtain and document the responses to each test case by the various sources. Here, there is no need to involve experts directly.
3. Evaluation
   This step interprets the results of the experimentation step and determines errors attributed to the system and reports it in an informal way.
4. Validity assessment
   This step analyzes the results reported above and reaches conclusions about the validity of the system.
5. System refinement
   In order to improve the final system, this step provides guidance on how to correct the errors detected in the system as a result of the previous 4 steps. This, hopefully, leads to an improved system.

These steps are iterative in nature, where the process can be conducted again after the improvements have been made. Figure 1 illustrates the steps outlined here.

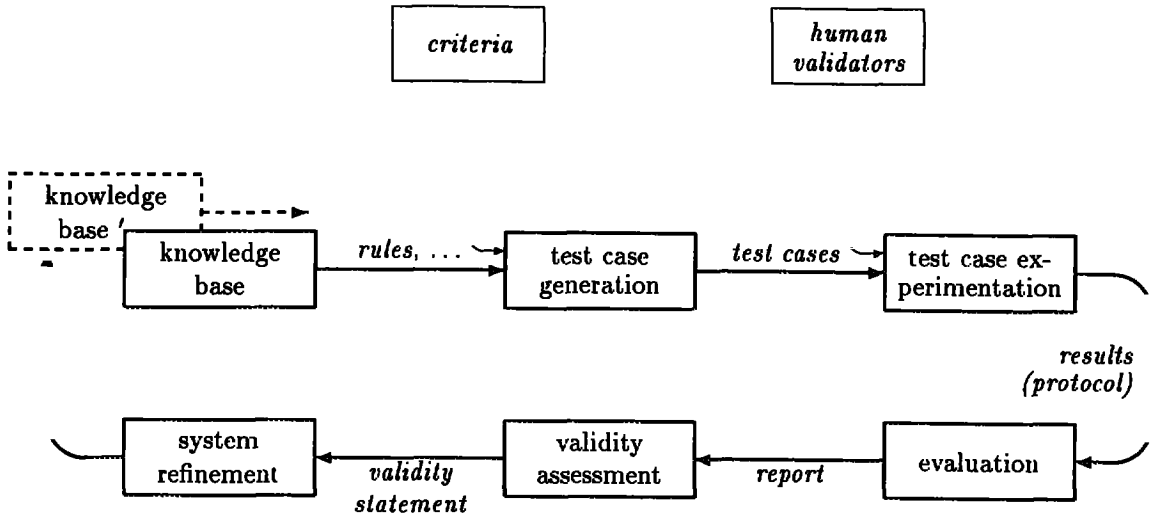

Figure 1: Steps in the Proposed Validation Process

The authors feel that particular paradigms of knowledge representation and problem solving require particular approaches to validation. Generally, this framework seems to be useful for all kinds of paradigms. However, it has to be refined for case based systems. Thus, the objective of the present paper is to introduce some first ideas of a technique for case based systems and to piont out the common issues and differences.

Characteristics of Case Based Systems

The knowledge representation of case-based reasoning systems (CBR systems) consists of just a library with cases (respectively, problems to be solved) with known solutions: symptoms with associated diagnoses, required process states with control instructions to reach it, system requirements with system compositions to meet it, e.g.

The problem solving paradigm of CBR systems consits of

1. the retrieval of a case out of the library, which is most similar to the given one,
2. the adaption of the solution from the case library towards a solution of a currently presented case, and
3. the update of the case library, i.e. the technique of adding and/or removing cases to/from the case library.

The core of the latter function is a learning problem: to add new cases usually means to learn some new knowledge. A very special class of learning problems, which is case-based learning of a formal language, is already pretty well researched, and there are even tools to validate the case-based learning principle (cf. (Beick and Jantke 1998)).

Applying the General Framework to Case Based Systems

Test Case Generation

CBR systems, by their nature, have a built-in set of test cases in their case library. These cases can and should be used (under consideration of the techinques of test case experimentation outlined below) as test cases as well:

1. The main reason to use the cases of the library as a basis for test cases is, that these cases are "real life cases", i.e. they come from practice or, at least they are deemed to be correct. Thus, the probability to have really representative cases is very high.
2. Furthermore, these cases have a known solution, i.e. there seems to be no need to involve experts in the evaluation proccdure.

On the other hand, there might be some disadvantages of using them as test cases: Of course, a CBR system should deliver the solution of the case in the case base, if we use exactly this case as a test case. Thus, we can't evaluate whether the system delivers good solutions for other cases and, which is the real core of this question, whether the system has

1. a qualified similarity concept (a qualified *case retrieval function*),
2. a qualified concept of adapting a case solution from the case library towards a solution of a currently presented (test) case, i.e. a qualified *case solution adaptation function*, and
3. a qualified concept of putting up new cases and removing historical cases (a qualified *case base updating function*).

But these are disadvantages only at first glance. The first two of the questions above are answered by the methodology proposed in (Gonzalez, Xu and Gupta 1997). However, the last one does not.

In the following section we adopt the ideas of (Gonzalez, Xu and Gupta 1997) and suggest some refinements as well as an idea to validate the case base updating function.

Test Case Experimentation

In (Gonzalez, Xu and Gupta 1997), e.g. the authors introduce a so called *Case Library Subset Test* (CLST) technique. It is devided into the steps

1. *Retrieval Test*, which is intended to evaluate the similarity concept and
2. *Adaptation Test*, which is intended to evaluate the concept of adapting a solution of the case library towards a solution of the presented case.

For both of these tests the cases in the case library serve as test cases as well. Here, we introduce these tests, suggest some refinements and introduce a third test, which is the *Updating Test*.

Differently from the general way (sketched in the introduction section of this paper), we do not need the support of human experts here because we have test cases with known solutions. Further refinements of this technique may introduce some doubt in the correctness of this solutions and include the experts again.

The Retrieval Test In (Gonzalez, Xu and Gupta 1997) the Retrieval Test has been successfully passed, if the CBR system indicates the historical case in the case library as the most similar one to the presented test case, which has been cloned from it. Theoretically, this is pretty natural.

The fact that the authors suggest this kind of test indicates, that this may not always happen in practice. Systems which don't find an identical case as the most similar one, have a poor similarity concept, indeed.

The authors feel, that this test can't really test the entire similarity concept; it just indicates, that this concept works fine in case of identity, which is just a special case of similarity.

Here, we suggest a refined retrieval test function on a very general level. Of course, it has to be refined depending on the similarity concept. Usually, similarities are

- either just flat scalar values, which indicate somehow a "distance" between two cases in a "case space"
- or structured terms, which indicate the "most special template", which fits on two (or more) cases.

A very practical example for the latter concept can be found in (Jantke and Arnold 1997). For both concepts there is (at least a partial) binary ordering relation $\sqsubseteq$ between (at least) two cases of the case base CB.

Thus, the suggested Retrieval Test runs as follows:

1. Remove a case out of the case base and use it as a test case.
2. Present this case to the CBR system and ask it for the most similar case in the (remaining) case base.
3. If the selected case of the case base is not the most similar one, the system failed the Retrieval Test for this test case.

   Otherwise do the following:

   (a) Temporarily add a more similar case to the case base and ask again for the most similar case.

   (b) If the system found this added case now as the most similar one, the system passed the Retrieval Test for this test case. Otherwise it failed.

All these steps have to be carried out for each of the library cases, of course.

The Adaptation Test The Adaptation Test described in (Gonzalez, Xu and Gupta 1997) runs as follows: The test case set is the same as in the Retrieval Test. However, before presenting a test case to the system, its historical (and identical) case is removed from the case library.

The output of this test contains retrieved cases and their solutions. The built-in adaptation function forms a final solution for the presented test case, which can be compared with the (known) correct solution of this case. If the formed solution is the same as the one of the removed case, the Adaptation Test has been passed for this test case, otherwise it has been failed.

Of course, further refinements are imagible here as well, but the authors feel, that this strongly depends on the nature of the solutions and cannot be done on a general level.

For example, the term "the same solution as the one of the removed case" can be "softened" by introducing a similarity concept for the solutions as well.

The Updating Test This test is another refinement of the technique proposed in (Gonzalez, Xu and Gupta 1997), but it is not really different from the tests above. If

1. the Retrieval Test came up with the "most similar case", which is "very unsimilar" (the "distance" in the "case space" is larger than a pre-defined minimal similarity or the structured term describing the similarity is just a variable, e.g.) or
2. the Adaptation Test was not able to form the correct solution,

then there is is a strong indication to add the presented case to the case library.

If the built-in updating function followed this indication, the Updating Test has been passed for the considered test case, otherwise it has been failed.

Evaluation and Validity Assessment

There are several ways to estimate the CBR system's validity, which differ in their "structureness" of the validity statement. The most informative way is reporting

1. the cases which failed the Retrieval Test,
2. the cases which failed the Adaptation Test,
3. the cases which failed the Updating Test together with the cases, which the Updating Test indicated an update for, and
4. the entire set of test cases.

Another way is to report an statistical interpretation of the protocol, i.e.

1. a *Retrieval Ability*, which is the number of cases which failed the Retrieval Test divided by the total number of test cases,
2. an *Adaptation Ability*, which is the number of cases which failed the Adaptation Test divided by the total number of test cases, and
3. an *Updating Ability*, which is the number of cases which failed the Updating Test divided by the number of test cases, for which the Updating Test indicated an update.

System Refinement

For CBR systems that support the updating function, we cannot suggest detailed refinement activities. The only thing which can be done with the results of the tests above is to let the systems' disigners know, where are the weaknesses of the system, i.e. which of the three functions didn't work satisfactory and for which test cases this was the case.

For CBR systems, which do not support the updating function, refinement activities should consist in making the case base "more representative", i.e. in collecting new cases for the library and performing the test above again.

Summary and Conclusions

Case based reasoning is a pretty common method of problem solving. Humans usually solve a certain class of problems in a case based manner: architects look for similar designs and modify it to become a solution for the current problem, authors of webpages look for similar pages and modify it towards the required one, disign engineers often do their job in a case based manner, ...

However, the general steps of validation of case based reasoning systems should be the same as the steps described by the authors in (Knauf, Abel, Jantke and Gonzalez 1998) for the validation of rule based systems, but the realization of the particular steps is very different.

The main advantage of solving problems in a case based manner is that there is no need for an explicit knowledge representation. Thus, there is even no need for explicit knowlege, i.e. the knowledge has not to be converted to an abstract and general level.

Validation with test cases means to apply the knowledge in a knowledge base to concrete particular problems, i.e. cases. In case of rule based systems finding test cases means to translate the general-level knowledge expressed as rules down to concrete knowledge expressed as test cases and their expected solutions. This translation process needs a lot of expertise which leads to the "reasonable" set of test cases. In contrast, finding test cases for case based systems can be performed by simply using the cases of the case base.

The test case experimentation of case based systems has the advantage that the expected solutions, which are deemed to be correct, are available. This is not true for rule based systems. Here, we have to find out solutions, which are deemed to be correct, in a very complicated process of asking more or less competent human experts.

The comparation of the evaluation, validity assessment, and system refinement steps of both paradigms is still not very well considered by the authors. However, these steps seem to be less costly for case based systems than for rule based systems as well.

A summary of common and different issues is outlined in table 1.

However, the validation of case based systems is still not very well researched by the authors, and there are a lot of questions left, for example:

- How can we test the coverage of the "problem space" by the case library?
- How can the term "consistency" of a case library be defined?
- How can be tested, whether the updating function keeps a case base consistently?
- How can we handle the dynamic character of case based systems?

The latter question is very substantial. Case based systems typically change over time. This is intended, and it is a crucial feature of case based systems. Consequently, the behavior of these systems normally changes over time. This has at least two implications:

1. Without modifying the criteria, a case based system which is valid today might be invalid tomorrow and vice versa.
2. There might be a need to look at the validity problem differently over time.

All this, usually, does not apply to rule based systems.

References

Abel, Th.; Gonzalez, J.A. 1997. Influences of criteria on the validation of AI systems. In (Jantke and Grieser 1997), 43-48

Abel, Th.; Knauf, R.; Gonzalez, A. 1996. Generation of a minimal set of test cases that is functionally equivalent to an exhaustive set, for use in knowledge-based system validation. In (Stewman 1996), 280-284

Beick, H.-R.; Jantke, K.P. 1998. Tools for Validation of CBL Principles. In (Grieser, Beick and Jantke 1998), 53-71

Table 1: Comparation of Validation Expenses and Benefits

	Rule-based Systems	Case-based Systems
Test Case Generation		
test cases	have to be generated	are available in the case base
domain coverage of the test case set	depends on the knowledge base and can not be guaranteed	depends on the case base and can not be guaranteed
efficiency of the experimentation with test cases	strongly depends on the talents of humans to reduce the test case set	is not that important because the case base is definitely finite and there is no human work to manage the experimentation
validation criteria	have to be provided, ranked and rated by human experts	are not needed
Test Case Experimentation		
test case solutions	have to be provided by human experts	are available in the case base
experimentation session	needs humans and can be supported by machine	can be done automatically
human validators (experts)	have to be involved to provide and rate solutions	are not needed
Evaluation and Validity Assessment		
generation of a validity statement	can be done automatically	can be done automatically
System Refinement		
system therapy	has to be done by human experts	*(is still not well researched)*
Human Support of the Validation Management		
knowledge engineers	have to be involved for criteria identification and to manage their ranking and rating	do not have to be involved in the validation session

Boehm, B.W. 1984. Verifying and validating software requirements and design specifications. In IEEE Trans. Software, 1(1):75-88

Dankel, D.D. ed. 1997. Proc. of the 10th Florida AI Research Symposium (FLAIRS-97). Daytona Beach, FL: Florida AI Research Society

Gens, W. ed. 1997. Proc. of 42nd International Scientific Colloquium (IWK-97), Vol. 2. Ilmenau, Germany, Ilmenau University of Technology

Grieser, G.; Beick, H.-R.; Jantke, K.P. eds. 1998. Aspects of Intelligent Systems Validation. Sapporo, Japan: Meme Media Management, Report MEME-MMM-98-2

Gonzalez, A.; Xu, L.; Gupta, U. 1997. Validation techniques for case based reasoning systems. In (Gens 1997), 71-76

Herrmann, J.; Jantke, K.P.; Knauf, R. 1997. Using structural knowledge for system validation. In (Dankel 1997), 82-86

Jantke, K.P.; Arnold, O.: Logical Case Memory Systems: Foundations, Learning Issues and Applications. In (Gens 1997), 3-8

Jantke, K.P.; Grieser, G. eds. 1997. Proc. of the 5th Leipziger Informatik-Tage 1997 (LIT-97), Workshop Validation von Anwendungssoftware. Leipzig, Germany: Forschungsinstitut für InformationsTechnologien Leipzig e.V.

Jantke, K. P.; Knauf, R.; Abel, Th.: The Turing test approach to validaton. In (Terano 1997), 35-45

Knauf, R.; Abel, T.; Jantke, K.P.; Gonzalez, A.J. 1998. A Framework for Validation of Knowledge Based Systems. In (Grieser, Beick and Jantke 1998), 1-20

Knauf, R.; Jantke, K.P.; Abel, T.; Philippow, I. 1997. Fundamentals of a TURING test approach to validation of AI systems. In (Gens 1997), 59-64

O'Keefe, R.M.; O'Leary, D.E. 1993. Expert system verification and validation: A survey and tutorial. In: Artificial Intelligence Review, 7(1993):3-42

Stewman, J.H. ed. 1996. Proc. of the 9th Florida AI Research Symposium (FLAIRS-96). Key West, FL: Florida AI Research Society

Terano, T. ed. 1997. Proc. of the 15th International Joint Conference on Artificial Intelligence, IJCAI-97, Workshop W32, Validation, Verification & Refinement of AI Systems & Subsystems, 1997, Nagoya, Japan

Wise, J.A.; Wise, M.A. 1993. Basic considerations in verification and validation. In: Wise/Hopkin/Stager (eds.): Verification and Validation of Complex Systems: Human Factors Issues. Vol. 110 of NATO ASI Series F: Computer and System Sciences, Springer-Verlag, 1993, 87-95