Higher Order Refinement Heuristics for Rule Validation

Hans - Werner Kelbassa

Email: kelbassa@uni-paderborn.de

Abstract

For rule validation there are no second and higher order refinement heuristics yet. This paper presents an example for second order refinement heuristics and introduces generic higher order refinement heuristics. It is proposed that rule refinement should be performed case-based, i.e. the whole spectrum from first order to higher order refinement heuristics can be processed instead of first order ones only. This approach extends the classical SEEK2 framework. Moreover, the generalization - specialization dichotomy is extended by defining context refinement as the third refinement class. It is shown that the system SEEK2 registers context refinement problems inadequate and that first order heuristics are suboptimal.

Introduction

The operating system OS/360 project revealed that large software systems could not be revised in an iterative manner; two successive versions of the OS/360 system in the *best case* have had *constant* error ratios (Bonsiepen and Coy 1990). This experience had demonstrated that there is a need for validation and verification systems. Critical scientists announced a maintenance crisis in the area of expert systems.

In the next section the technical term validation is defined and the term refinement is explained. Then rule refinement heuristics will be considered. Doing this the technical term *context refinement* is introduced, and the classical SEEK2 framework (Ginsberg 1986) is extended. Moreover, the SEEK2 performance will be interpreted. Also the functionality of a validation system with an *acquisition interface* will be described. Then the problem of *second order refinement* will be discussed using a simple example. The nature of second order refinement heuristics for rule validation is described. In order to generalize the result of the discussion concerning the refinement heuristics the *generic forms* of the first, second, and higher order refinement heuristics will be presented. Finally, it is shown that the SEEK2 system is invoking inadequate validation measures if context refinement problems appear. Because of this need for coping with context refinements, a new validation measure for

three refinement classes is defined and applied to a higher order refinement example.

Validation and Refinement

Supported by Gonzalez and Barr (2000) the following definition of rule validation is introduced, which is compatible with the TEIRESIAS rule tracing approach:

Rule validation is the process of ensuring that the reasoning path (rule trace) and the output of the rule-based system is equivalent to those of human experts when given the same inputs.

This definition emphasizes that the crucial tasks of the validation process are the comparison of the reasoning paths obtained from the rule-based system with those from the domain expert and, moreover, the identification and refinement of rules, which are 'responsible' for the misinterpretation or misdiagnosis of at least one case.

The technical term *rule refinement* distinguishes the initial rule-base construction phase from the revision phase to be performed later on, after the expert system has reached a considerable competence, but is not always able to find the valid experts reasoning paths. Failures recognized during the refinement phase should not lead to the removal of faulty rules but to their correction. The central idea of refinement systems is to find a *minimal* revision so that the falsified cases get valid reasoning paths without any side effects.

Validation System

Rule revision operations are interdependent, i.e. there are 'side effects' which cannot be anticipated. The pioneer system for rule refinement was SEEK/SEEK2, which gathered statistical knowledge regarding all rules of the rule base (in order to propose special refinements to the user). This meta knowledge was processed by heuristics which enabled SEEK to determine whether a rule should be generalized or specialized (Politakis 1985). A generalization causes a refined rule to fire more often. One possibility to generalize a faulty rule is to weaken the premises by substituting a present conjunction ($\wedge$) through a disjunction ($\vee$). A rule specialization means a refinement that makes it harder for the rule to be fired. One possibility to specialize a rule is to

insert at least one additional condition into the if-part of this rule, if for the present case this conjunction is accepted by the expert.

However, the problem with SEEK/SEEK2 is that the refinement dichotomy – generalization and specialization – is not complete. There is a third refinement class. Consider the simple rule $R1 := (A \wedge B) \rightarrow I$. If this rule is revised by inserting a negation, so, for example, it is possible to get the rule $R1^* = (A \wedge \neg B) \rightarrow I$. This refinement is no generalization and also no specialization. What will happen is that the refined rule will fire in another context. This rule $R1^*$ will fire not more and not less often than $R1$ did. Therefore it is to be ascertained that there is a third refinement class to be called 'context refinement' (contextualization).

The SEEK2-like first order heuristics to be considered here are based on relative frequencies of recognized inference failures which are acquired from the domain expert by the validation interface. These relative frequencies enable the validation module to compute generalization and specialization measures for every rule of the given rule base in order to derive the *best* refinement operation to be proposed to the domain expert. The validation interface is described in (Kelbassa 1990). The idea pursued here is to acquire as much as possible *validation expertise* from the domain expert during the evaluation of the expert systems reasoning path, and to look for excellent validation measures which characterize the need for a special refinement with high certainty. The refinements proposed by the validation system need not be accepted by the domain expert; he can reject (empirical evaluation). The expert is still the final judge. The validation interface enables the combination of the conceptual ideas of TEIRESIAS and SEEK2. The central idea of TEIRESIAS was single case analysis by rule tracing (Davis 1979). The central idea of SEEK2 is multiple case analysis and automatic refinement. So the domain expert should be able to enter recognized reasoning failures or missing and wrong rule outputs by a validation interface.

The quality of the validity measures determines the efficiency of the refinement system. The ideal is that there is an exact measure for each possible refinement operation, but this ideal has not been reached yet. So the performance of SEEK2 seems to be on low level. The developer of SEEK2 reported that they carried out 199 refinement experiments but only 10 of these were accepted. So the acceptance was about 5% only (Ginsberg et al. 1988). This result reveals that there is a need for better validation measures and more powerful refinement heuristics.

Second and Higher Order Refinement

The current approach is to work with first order rule refinement heuristics if there are multiple refinement candidate rules for one case in which the reasoning path of the rule-based system differs from the domain experts one. So, for example, if the validation expert has found out that three rules were faulty in a special case, SEEK2 is processing three first order refinement heuristics, one for every refinement candidate rule. The order of the refinement heuristic is determined by the number of rules which are refined by this heuristic (Ginsberg 1986).

Here it is proposed to work with one refinement heuristic for every case. The drawback of first order heuristics is that their *local view* becomes inadequate, if there are multiple refinement problems for one or more cases. If there is one refinement candidate rule only which is 'responsible' for the wrong reasoning path (rule trace) of a special case then the SEEK2 first order approach is sufficient. But there is a need for *higher order* refinement heuristics if there are more than two refinement candidate rules for the evaluated case. Therefore, there should be a development of revision heuristics of second and higher order based on the available validation knowledge (meta knowledge).

Second Order Refinement Example

For the explanation of second order refinement heuristics an example is discussed. Consider the following rules contained in a rule base RB:

$RB := \{..., R_{18}, ..., R_{42}, ..., R_{46}, ..., R_{54}, ...\}$
$R_{18} := (A \wedge B) \rightarrow Hypothesis1$
$R_{42} := (C \vee D) \rightarrow Hypothesis3$
$R_{46} := (Hypothesis1 \wedge \neg Hypothesis3) \rightarrow I_8$
$R_{54} := (Hypothesis3 \wedge E) \rightarrow I_2$
User input for case $c_1 : D_1 = \{B, C, D, E\}$
Valid system output for case $c_1 : I_8$

In this example the forward chaining system is to be considered as interpretation system, which can derive the interpretations I_2 and I_8, if the output rules R_{54} and R_{46} are fired. Assume that the trace validation by the domain expert has shown that the required rule R_{18} did not fire and that the fired rule R_{42} was a wrong element in the reasoning path for case c_1. As a consequence the output rule R_{46} did not fire and case c_1 was falsified. In order to refine the reasoning path for case c_1 let the output rule R_{46} be a rule which should be fired for case c_1 *after* the rule R_{18} was generalized and *after* the rule R_{42} was specialized. If the input rule R_{18} is generalized by changing the premises, then at least one new reasoning path is added, i.e. to R_{46}. If the rule R_{42} is specialized by changing the premises, then at least one previous reasoning path is deleted, i.e. to R_{54}.

The validation result for case c_1 leads to the following second order rule refinement heuristic RH2:

If rule R_{18} is generalized,
and
if rule R_{42} is specialized,
then the falsified case c_1 gets the valid reasoning path.

A case is falsified if the reasoning path of the rule-based system differs from the reasoning path of the domain expert. Using this heuristic RH2 the rule R_{46} will correctly fire more often if R_{18} was generalized and R_{42} was specialized. Whether these revision conditions are true, depends on the content of the given validation knowledge for *all cases*, which enables the calculation of measures regarding the different conditions in the if-part of the rule. This means that the available actual *validation expertise* is processed by first order heuristics, which compute for every rule of the rule

base the generalization and specialization measure. For the efficiency and utility of refinement heuristics it is important to find out to which extent it is possible to perform calculations regarding the rule conditions, which are based on the available validation expertise.

For example it may become true that the second (specialization) condition of rule RH2 is not satisfied. In this situation it is possible to think about the alternative second order heuristic RH2/2 which must be referring to at least one another case c_2:

If rule R_{18} is generalized,
and
if rule R_{42} is generalized,
then the falsified case c_2 gets the valid reasoning path.

The conditions of these two second order refinement heuristics are directly based on the case validation results from the domain expert. But whether these refinement heuristics will fire is depending on the computed *first* order measures concerning all processed cases, which do not directly take into account that there is sometimes more than one refinement candidate rule for at least one falsified case.

Let it be assumed that these first order measures evaluated the rule R_{18} and R_{42} to be *generalized*. Therefore the heuristic RH2 will not work, but the heuristic RH2/2 will work. One way to obtain a generalization for rule R_{42} is to extend the *disjunctive* if-part of R_{42} by at least one additional condition. If the domain expert accepts to add the condition B to the if-part of R_{42}, then we get the refinement $R_{42}^* := (B \lor C \lor D) \rightarrow Hypothesis3$. This rule can fire under the same circumstances as the previous rule R_{42}, and in addition if condition B is true, i.e. this generalized rule can fire more often than R_{42} did. As the rule R_{18} is a generalization candidate, too, it is possible to refine R_{18} in a similar manner; but also alternative refinement operations are available. One possibility to generalize rule R_{18} is to substitute the present conjunction by a disjunction: $R_{18}^* := (A \lor B) \rightarrow Hypothesis1$. These revisions will improve the validity of the reasoning paths for the outputs I_2 and I_8.

Higher Order Refinement Heuristics

The higher order refinement problem has been described by Politakis in the following way: "The problem is that a single rule change is often not enough to correct the case. It is not uncommon to find multiple problems with the rules because more than one conclusion was reached with confidence exceeding that for the expert's conclusion." [1]

The idea of higher order rule revision is focussing on multiple rule refinements for at least one case.

The above discussion claimed that rule refinement is to be performed in a SEEK2-like manner, i.e. either as generalization or as specialization (Ginsberg et al. 1988). In (Kelbassa 1990) a *third generic refinement class* was introduced: context refinement. In the following let GC mean

---

[1] P. G. Politakis 1985, p. 56

Generalization Candidate, let SC mean Specialization Candidate, and let CC mean Context Candidate. Moreover, let $x_1, ..., x_n$ ($n \in \mathbb{N}$) mean the indices of the refinement candidate rules. Further, let $c_1, ..., c_l$ ($l \geq 1, l \in \mathbb{N}$) be the falsified case(s) to be revised. Then the generic refinement heuristics have the following form.

- *First order rule refinement heuristic:*

    IF rule $R_{x_1} \in \{$ GC, SC, CC $\}$,
    THEN the falsified case(s) $c_1, ..., c_l (l \geq 1)$
    get(s) valid reasoning path(s)

- *Second order rule refinement heuristic:*

    IF rule $R_{x_1} \in \{$ GC, SC, CC $\}$,
    AND
    IF rule $R_{x_2} \in \{$ GC, SC, CC $\}$,
    THEN the falsified case(s) $c_1, ..., c_l (l \geq 1)$
    get(s) valid reasoning path(s)

- *Higher order rule refinement heuristic:*

    IF rule $R_{x_1} \in \{$ GC, SC, CC $\}$,
    AND
    IF rule $R_{x_2} \in \{$ GC, SC, CC $\}$,
    AND
    ....
    AND
    IF rule $R_{x_n} \in \{$ GC, SC, CC $\}$,
    THEN the falsified case(s) $c_1, ..., c_l (l \geq 1)$
    get(s) valid reasoning path(s).

In order to revise the reasoning paths it is often not sufficient to apply common used first order heuristics although there are *multiple* refinement problems revealed for one case (or class of cases). If revision in this situation is done with first order refinement heuristics, then there must be a special sequence strategy - instead of one higher order heuristic. The above described first, second, and higher order rule refinement heuristics enable a case-based rule revision view because it is possible to perform *every* case revision by one refinement heuristic only.

Refinement Problems

For the performance of rule refinement heuristics it is crucial to find well-defined measures for the different validation problems. If the rule $(A \land B) \rightarrow I$ is fired, and the validating expert rejects this system result, because the correct rule should be $(A \land \neg B) \rightarrow I$, then SEEK2 registers a need for specialization although the target refinement is context refinement. The statistical measures which SEEK2 invokes if the fired rule is *not* the right one, are the measures $SpecA(R_x)$ and $SpecB(R_x)$ with R_x meaning any rule of the given rule base ($R_x \in RB$). The validation measure $SpecA(R_x)$ is defined as follows:

$SpecA(R_x)$ is the number of cases in which
(a) this rule's conclusion should not have been reached but was, and
(b) if this rule had failed to fire, the correct conclusion would have been reached.

If there is more than one fired rule that concludes the incorrect result, then none of these rules has its SpecA measure incremented. *Instead of this SEEK2 has an additional concept to cover this situation called SpecB(R_x): each of these faulty rules get its SpecB measure incremented.*

However, the SpecA and the SpecB measures register context problems inadequate. A context refinement is *neither* a generalization *nor* a specialization. Unfortunately SEEK2 has no validation measure which is regarding context refinement (Ginsberg 1986). Therefore there is a need for a new measure. This becomes obvious if the $Gen(R_x)$ measure is examined now.

The above example with the SpecA measure interpreted an explicit inference failure. In addition, there are implicit inference failures, too. If the expert system did *not* fire the rule $(A \wedge B) \rightarrow I$ because the if-part of this rule was not satisfied, the target refinement can, for example, get the rule $(A \wedge \neg B) \rightarrow I$. This means that a context refinement is required in order to cope with this implicit inference failure. In this situation SEEK2 invokes the validation measure $Gen(R_x)$ for the missing rule trace element $R_x \in RB$.

The validation measure $Gen(R_x)$ is defined in the following way:
$Gen(R_x)$ is the number of cases in which
(a) this rule's conclusion should have been reached but was not,
(b) had this rule been satisfied, the right conclusion would have been reached, and
(c) of all the rules for which the preceeding clauses hold in the case, this one is the closest to being satisfied.

The SEEK2 system registers the implicit inference failure as generalization problem although there is a context refinement on target. So the $Gen(R_x)$ measure is inadequate, too.

A basic heuristic in SEEK2 is the generalization heuristic:

$$Gen(R_x) > Spec(R_x).$$

Here $Spec(R_x)$ means: $Spec(R_x) = SpecA(R_x) + SpecB(R_x)$. If the value of $Gen(R_x)$ is larger than the value of $Spec(R_x)$, then R_x becomes a generalization candidate. The idea of this heuristic is excellent, but this heuristic is unable to cope with context problems. The measures for generalization and specialization have not to cover context problems, i.e., the measure for context refinements must be separated from the registration of the *genuine* generalization and specialization problems. This is already possible during the evaluation session after the validation interface has acquired the target refinement. However, this topic is an interesting one for future rule validation research.

Let $Con(R_x)$ be the number of cases, in which context refinement is required, so that the falsified cases get valid reasoning paths. This validation measure must enable the validation module to decide which kind of refinement is to be performed for rule R_x. Therefore the target refinement class for any rule R_x must be selected on the basis of the maximum of the functions $Con(R_x)$, $Gen(R_x)$, and $Spec(R_x)$. *Let $Tar(R_x)$ be the refinement class which is to be determined as the target one for rule R_x. Then $Tar(R_x)$ is defined by the following expression:*

$$Tar(R_x) := max\{Con(R_x), Gen(R_x), Spec(R_x)\};$$
$$Con(R_x) \in \mathbb{N}, Gen(R_x) \in \mathbb{N}, Spec(R_x) \in \mathbb{N}.$$

This new measure will enable the validation module to process context refinements. [2]

Higher Order Refinement Example

Let it be assumed that the rules R_{46}, $R_{54} R_{64}$ are refinement candidates, and that the following refinement heuristics have been obtained from the domain experts rule trace evaluation (validation expertise); these ones are to be considered as elements of a higher order refinement example:

RH3 :=　IF　rule R_{46} is specialized (ϕ_S^2),
　　　　　and
　　　　　rule R_{54} is generalized (ϕ_G^1),
　　　　　and
　　　　　rule R_{64} is contextualized (ϕ_C^2),
　　　THEN **8** falsified cases get valid reasoning paths (rule traces)

RH1 :=　IF　rule R_{46} is generalized (ϕ_G^2),
　　　THEN **4** falsified cases get valid reasoning paths (rule traces)

RH1/2 :=　IF　rule R_{64} is contextualized (ϕ_C^1),
　　　THEN **2** falsified cases get valid reasoning paths (rule traces)

RH3/2 :=　IF　rule R_{46} is specialized (ϕ_S^1),
　　　　　and
　　　　　rule R_{54} is specialized (ϕ_S^2),
　　　　　and
　　　　　rule R_{64} is generalized (ϕ_G^3),
　　　THEN **9** falsified cases get valid reasoning paths (rule traces)

In these rules the symbols ϕ_S^1, ϕ_S^2, ϕ_G^1, ϕ_G^2, ϕ_G^3, ϕ_C^1, and ϕ_C^2, characterize *elementary* rule refinement operations. The index C means contextualization (ϕ_C), the index G means generalization (ϕ_G), and the index S means specialization (ϕ_S); the set of all rule refinements is $\Phi := \{\phi_C, \phi_G, \phi_S\}$.

First order gain calculation:
$Tar(R_{46}) := max\{Con(R_{46}), Gen(R_{46}), Spec(R_{46})\}$
$Tar(R_{46}) = max\{0, 4, 8 + 9\} = Spec(R_{46}) \equiv \phi_S(R_{46})$

$Tar(R_{54}) := max\{Con(R_{54}), Gen(R_{54}), Spec(R_{54})\}$
$Tar(R_{54}) = max\{0, 8, 9\} = Spec(R_{54}) \equiv \phi_S(R_{54})$

$Tar(R_{64}) := max\{Con(R_{64}), Gen(R_{64}), Spec(R_{64})\}$
$Tar(R_{64}) = max\{8 + 2, 9, 0\} = Con(R_{64}) \equiv \phi_C(R_{64})$

---

[2] The above heuristics do not determine by a significance measure *whether* a special rule should be refined (see P. G. Politakis 1985, p. 69), but *how* a rule should be corrected – so the rule refinement problem is more complicated.

First order gain result: RH1/2 fires - **2 cases.**

Higher order gain calculation: RH3 or RH3/2?
RH3: RH3 can validate 8 cases + 2 cases with RH1/2
  RH3: RH3 + RH1/2 = **10 cases**
RH3/2: RH3/2 can validate **9 cases**

This example shows that first order heuristics lead to a suboptimal result. The first order gain calculation result is 2 cases only, because the only heuristic which can fire is RH1/2. Higher order gain calculation leads to better results; the minimum gain is 9 cases, the maximum gain is 10 cases. If the heuristic RH3 is executed, then RH1/2 can fire, too, and the higher order gain is 10 cases. If the heuristic RH3/2 is executed, then the higher order gain is 9 cases. Whether it is better to fire RH3 or RH3/2 depends on the nature of the refinements $\phi_C^1(R_{64})$ and $\phi_C^2(R_{64})$. If the refinements $\phi_C^1(R_{64})$ and $\phi_C^2(R_{64})$ are not *compatible* so that both can not be performed simultaneously, rather either refinement $\phi_C^1(R_{64})$ or refinement ϕ_C^2 only, then heuristic RH3/2 should fire, else RH3. This higher order rule refinement example shows that first order reasoning is not optimal and that higher order gain calculation can find the maximum.

Conclusion

For the time being there is no definition and testing of second and higher order rule refinement heuristics. So here the second order rule refinement heuristic has been discussed and formalized. Moreover, the generic form of higher order rule refinement heuristics has been defined.

The classical SEEK2 approach is based on the generalization – specialization dichotomy. It is shown, that SEEK2 is incomplete with regard to the context problems and that context refinements must be considered as the third refinement class. Furthermore, it is outlined that SEEK2 invokes inadequate validation measures if context problems appear, and that SEEK2 has no validation measure for coping with context refinements. This means that a new validation measure as defined above must be processed.

There is a need for higher order refinement heuristics if there are more than two refinement candidate rules for any falsified case, because the first order heuristics get inadequate in this situation. Therefore it has been proposed to work with one refinement heuristic for every falsified case, and to apply higher order gain calculation if multiple refinement problems appear. SEEK2 is looking for the correct input – output inference, but it is not looking for the right intermediate reasoning steps (Ginsberg 1986). Here the definition of the technical term validation at the beginning is including that the reasoning path (rule trace) of the expert system must be the same as the inference chain of the domain expert because the intermediate inferences are important, too. SEEK2 does not gather specialization information for intermediate inferences (Ginsberg et al. 1988). Therefore it has been suggested here to acquire this validation information directly from the domain expert by the validation interface.

References

Bonsiepen, L., Coy, W. 1990. Szenen einer Krise - Ist Knowledge Engineering eine Antwort auf die Dauerkrise des Software Engineering? *Kuenstliche Intelligenz 4 (2): 5 - 11*

Davis, R. 1979. Interactive Transfer of Expertise: Acquisition of New Inference Rules. *Artificial Intelligence 35 (2): 197 - 226*

Ginsberg, A. 1986. *Refinement of Expert System Knowledge Bases: A Metalinguistic Framework for Heuristic Analysis.* Technical Report, CBM-TR-147, Department of Computer Science, Rutgers University

Ginsberg, A., Weiss, S. M., Politakis, P. 1988. Automatic Knowledge Base Refinement for Classification Systems. *Artificial Intelligence 35 (2): 197 - 226*

Gonzalez, A. J., Barr, V. 2000. Validation and verification of intelligent systems – what are they and how are they different? *Journal of Experimental and Theoretical Artificial Intelligence 12: 407 - 420*

Kelbassa, H.-W. 1990. Fallbezogene Revision und Validierung von regelbasiertem Expertenwissen fuer die Altlastenbeurteilung. In W. Pillmann and A. Jaeschke (eds.): *Informatik fuer den Umweltschutz.* Berlin, Germany: Springer-Verlag. 276 - 285

Politakis, P. G. 1985. *Empirical Analysis for Expert Systems.* Boston, Mass.: Pitman Publishing Inc.