Some Revisions in VCIRS and Cases Reconstructing Perspectives

Irfan Subakti
Department of Informatics, Faculty of Information Technology
Institute Technology of Sepuluh Nopember Surabaya (ITS)
Kampus ITS, Keputih, Sukolilo, Surabaya, Indonesia
Email: yifana@gmail.com

Abstract

Since Variable-Centered Intelligent Rule System (VCIRS) was proposed, there're some researches dealing with. However, it's necessary to drop a line concerning about some revisions that it had to make, for the user's convenience. It's about the option during knowledge building.

The tree transformation process from an existing tree of a domain into VCIRS knowledge base (KB) is exciting to realize, in the perspective of cases reconstructing. There're two kinds of facts could be observed from. The first is from the user's perspective when she performs KB transforming from such tree into VCIRS KB through generating cases. The latter is about the system (i.e., VCIRS) perspective, whereas it's more concern into the structure of KB has to construct within VCIRS.

The conclusion obtained from this stage is it should be a prudential concern dealing with cases reconstruction from an existing tree of a domain into VCIRS KB, since it will change the implementation of both perspective either from the user or the system. By understanding this matter, it will make the transformation process from a tree of a domain into VCIRS KB easier to make.

Keywords: Variable-Centered Intelligent Rule Systems, cases reconstruction

1. Introduction

VCIRS has been implementing in some areas recently, since it's proposed at 2005 [Sub05]. It's involving fuzzy, genetic algorithm, game, etc. The stage we deal now is applying and proofing VCIRS into real world applications.

During implementing such areas there's some problem in interpreting VCIRS into coding, from [Sub05]. There's an implicit matter and its explanation to be conduct prominently. That's why it needs to revise some topics in VCIRS, even though there are not changes in substance within.

In other hand, with regarding to making VCIRS down to the earth there's something about processing between real world matter and VCIRS, i.e., cases reconstructing from real world tree/diagram of a domain. The first step to convey the idea from VCIRS in any domain is to imitate the tree diagram (i.e., thus it's KB) into VCIRS KB. Some problems come up from kind is it's very difficult to generate/reconstruct cases which are built the tree diagram. It also has a relation with the inference process, since a kind of tree/diagram influent the mechanism during cases reconstructing.

2. Variable-Centered Intelligent Rule System [Sub05]

VCIRS hybridizes techniques from Rule-based Systems and Ripple Down Rules, as depicted at fig. 1. VCIRS has a structure to organize the rule base so that easy knowledge building, powerful knowledge inferencing, and evolutional improvement of the system can be obtained at the same time.

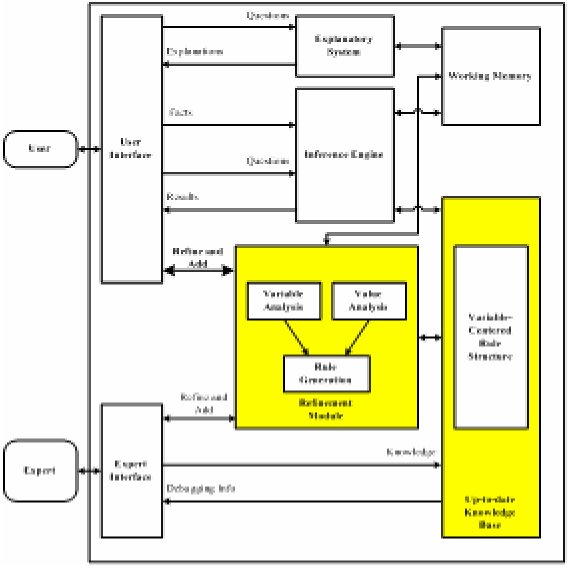

Figure 1. VCIRS architecture

A Variable-Centered Rule Structure is used to represent knowledge base and supports the Refinement Module to maintain an up-to-date knowledge base. It also records cases and their occurrence. The fundamental element of Variable-Centered Rule Structure is a variable, posted by the user. VCIRS maintain carefully the variable about its values, structure and occurrence. A sequence of variables constitutes a node, while a sequence of nodes composes a rule. Thus, the Variable-Centered Rule Structure contains a rule structure and a node structure centered on variables.

The case presented by the user goes to working memory during knowledge building, then save permanently into Variable-Centered Rule Structure while the system records the rule information and calculates the occurrence of each case. Then, the rule information recorded is used by Variable Analysis to get the important degree. In other hand the occurrence of each case is used by Value Analysis to get the usage degree. The usage degree will help the user to be a guideline during knowledge building and inferencing for deciding which variable she has to visit first. Along with the important degree, the usage degree will support Rule Generation for producing the new rule/node.

3. Revisions

For the clearance of the knowledge acquisition (KA) process from the cases in the real world into KB in VCIRS, we need to pond the way the user choose the condition from the cases. It's depicted at fig. 2. Actually this mechanism is inspired by RDR, only rather lifting cornerstone case (CC) [Comp93] the rules in KB itself still elaborate here.

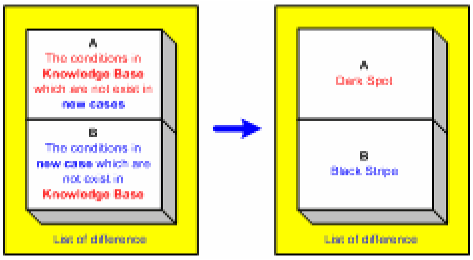

Figure 2. List of difference

By using this list of difference (LD), it's convenience for the user during KA process. The user only needs to choose the options easily and rapidly, and more important is it presents the clearance base on the perspective of user.

Further, it also needs to revise the options that the user easily and rapidly chooses. In [Sub05] during knowledge building algorithm, there're only 3 options the user can be chosen. After implementing through the research, according to Variable-Centered Rule Structure within VCIRS and cases presented in the real world, here the revisions are:

0. The user simply accepts the existing rules, no need to concern about the new cases presented. It means it's only the inferencing process.
1. The new rule being produced is come from the negation of section A at LD.
2. The new rule being produced is come from section B at LD.
3. The new rule being produced is the accumulation from option 1 and option 2 above.
4. The new rule is being produced is come from the new case presented without looking at LD. The place of this rule is the same level with the old case being checked. It means this new rule becomes the youngest sister within the old rules being checked or in other word this new rule will has the same parent with the old rule. However, just because its order is the latest one created, then this new rule becomes the youngest children from the parent of the old rules being checked.
5. The new rule being produced, all the things come from the new case presented without looking at LD, and the place of this kind is directly under the imaginer root within VCIRS KB. In other hand, thus the rule lays at the top level of KB.

For convenience, in the following there's example to make the 5 options above clear.

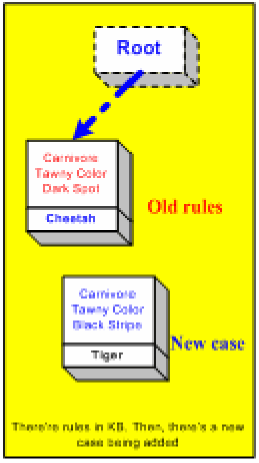

Figure 3. A new case being added

A new case being added into the KB in the VCIRS Variable-Centered Rule Structure as depicted at fig. 3.

Option 0 will be clear as depicted at fig. 4 in the following. Why the initial number is 0 (zero), is just because it's emphasizing the fact that actually there's nothing change in VCIRS KB.

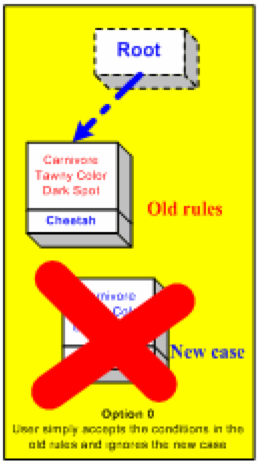
Root
Carnivore
Tawny Color
Dark Spot
Cheetah
Old rules
New case
Option 0
User simply accepts the conditions in the old rules and ignores the new case

Figure 4. Option 0

For the explanation of option 1, option 2, option 3, option 4 and option 5; it will be depicted at fig. 5-fig. 9 below.

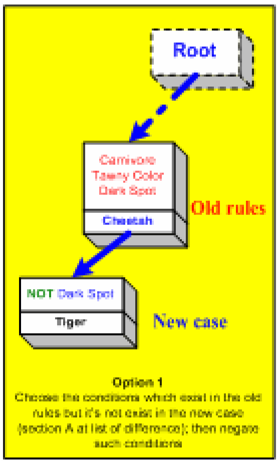
Root
Carnivore
Tawny Color
Dark Spot
Cheetah
Old rules
NOT Dark Spot
Tiger
New case
Option 1
Choose the conditions which exist in the old rules but it's not exist in the new case (section A at list of difference); then negate such conditions

Figure 5. Option 1

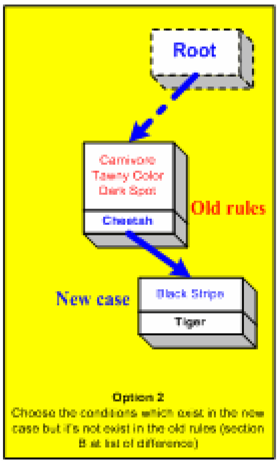
Root
Carnivore
Tawny Color
Dark Spot
Cheetah
Old rules
New case
Black Stripe
Tiger
Option 2
Choose the conditions which exist in the new case but it's not exist in the old rules (section B at list of difference)

Figure 6. Option 2

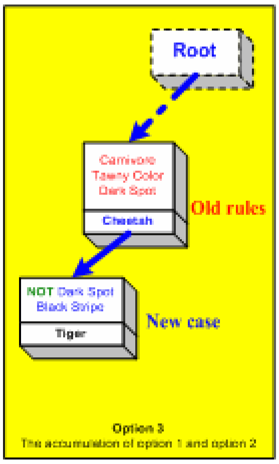

Figure 7. Option 3

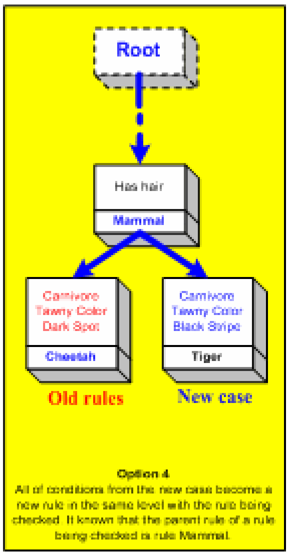
Root
Has hair
Mammal
Carnivore
Tawny Color
Dark Spot
Cheetah
Carnivore
Tawny Color
Black Stripe
Tiger
Old rules
New case
Option 4
All of conditions from the new case become a new rule in the same level with the rule being checked. It known that the parent rule of a rule being checked is rule Mammal.

Figure 8. Option 4

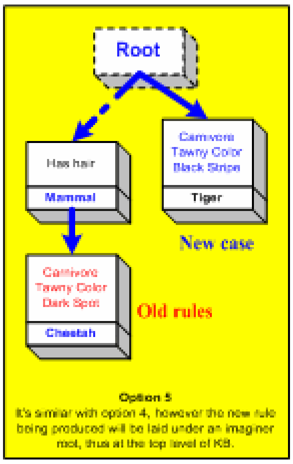
Root
Has hair
Mammal
Carnivore
Tawny Color
Black Stripe
Tiger
New case
Carnivore
Tawny Color
Dark Spot
Cheetah
Old rules
Option 5
It's similar with option 4, however the new rule being produced will be laid under an imaginer root, thus at the top level of KB.

Figure 9. Option 5

4. Case Reconstructing Perspectives

In the real world, there're many systems already which drawn in a tree/diagram. The problem came up, in the term of KA's perspective, between the real world's tree/diagram and KB within VCIRS.

For example, there's a tree/diagram of animal matter, i.e., Zookeeper rule base, as presented in [Win92] as depicted at fig. 10 below.

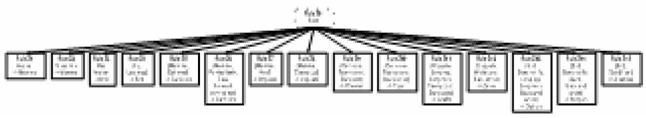

Figure 10. Zookeeper rule base

From fig. 10, when it transforms into VCIRS, there's no problem with the KA process as it easily will be formed into VCIRS KB as depicted at fig. 11 in the following for instance. Of course the result will be varied depends on the case presented and confirmed by the user. Fig. 11 is only the example, there're many forms indeed can be produced.

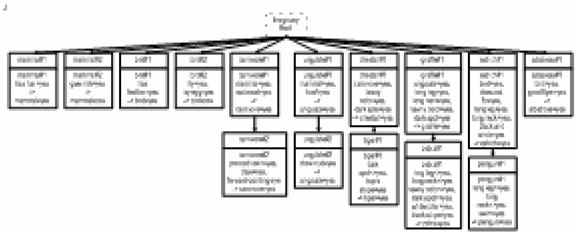

Figure 11. Zookeeper rule base in VCIRS

After all, there's no problem during KA for the user, both in RBS [Baur90] and RDR [Kang95]. It because there're 2 parts apparently, i.e., clause and conclusion, so that there's no confuse to allocate which one the case has to enter into. User easily reconstruct the cases have to present from a rule base into KB within VCIRS.

However, the very different circumstance comes up when the tree/diagram has a form like fig. 12.

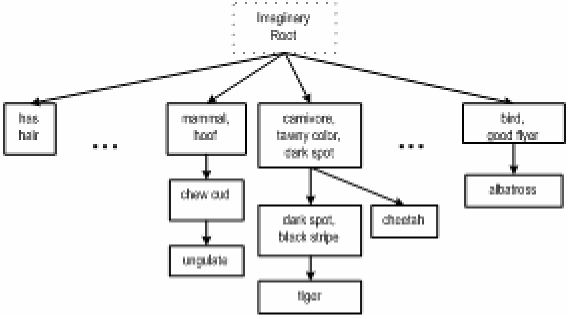

Figure 12. The problem caused by a kind of tree

For a particular path, one by one the above can be solved for cases reconstruction. However, first of all it should be noted that for an existing node which has children (i.e., more than 2 child nodes) it will trait as a node being reconstructed with a clause as the conditions state, added by a marked conclusion that will be inherited to its children, as depicted at fig. 13 in the following.

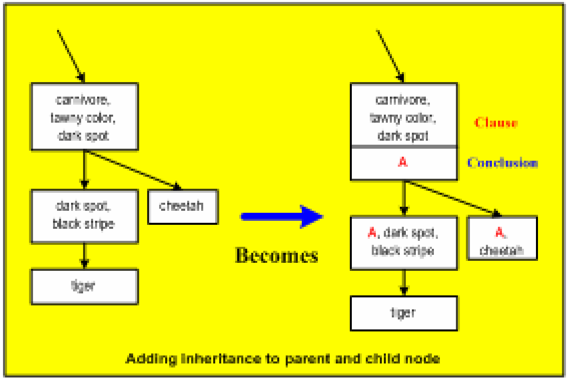
carnivore, tawny color, dark spot
dark spot, black stripe
cheetah
tiger
Becomes
carnivore, tawny color, dark spot
Clause
A
Conclusion
A, dark spot, black stripe
A, cheetah
tiger
Adding inheritance to parent and child node

Figure 13. Inheritance matter

The options for the cases obtained for cases reconstructing are perform by picking each conditions which has not a clause and a conclusion part, and trait it as either clause or conclusion or both. There're so many alternatives can be presented to be 6 options, as presented below:

1. All nodes trait as a clause, except the last one to be a conclusion.
2. All nodes trait as a clause and also trying to get a conclusion by conclusion generating.
3. All nodes trait as a conclusion and also trying to get a clause by clause generating.
4. All nodes trait as a conclusion, except the first one to be a clause.
5. A combination of existing nodes, whether it will be trait as a clause or a conclusion. A marked clause should be added for following new node.
6. All features in option 5 added by clause and conclusion generation.

The explanation of each option will be described in the following.

Option 1

The case will be obtained is only one node containing with clauses (i.e., from all nodes, except the last one, see fig. 12) and having a conclusion (i.e., the last node at fig. 12). Either in the RBS KB or VCIRS KB, it will create a flat structure. Here, it can be seen the interesting perspective from the system, whereas VCIRS KB doesn't have any advantages comparing with RBS KB, in the term of KB structure, KB building and inferencing, even KB refining as presented in [Sub05].

From the perspective of the user, this option is the simplest one. User doesn't need to generate cases (i.e., whether manually or by helping the system's generating process). She, then, can simply just pick up all the presented nodes in the existing tree and enter it into KB of desired system. This option is depicted at fig. 14.

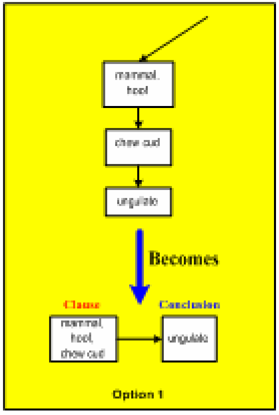

Figure 14. An example of option 1

Option 2

The case will be obtained is a node containing with clauses (i.e., from all nodes, see fig. 12) and trying to have a conclusion afterwards. The conclusion generating is a hard part for user perspective, since she forced to find another fact that doesn't exist in the existing tree. This finding conclusion can be helped by the system (e.g., conclusion generating from existing KBs, internet), even though it seems still count on the user itself by either manual entering or final judging. Further, either in the RBS KB or VCIRS KB, it will create a flat structure.

As in the option 1, from the system perspective the VCIRS KB doesn't have any advantages comparing with RBS KB, in the term of KB structure, KB building and inferencing, also in KB refining. This option is described at fig. 15.

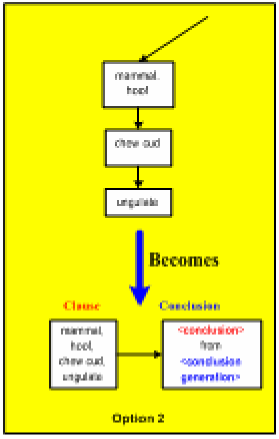

Figure 15. An example of option 2

Option 3

The case will be obtained is a node containing with conclusions (i.e., from all nodes, see fig. 12) and trying to have a clause, at least, beforehand. The clause generating is also a hard part for user perspective, since she forced to find another fact that doesn't exist in the existing tree. This finding clause also can be helped by the system by involving the user, just like option 2 does. Again, either in the RBS KB or VCIRS KB, it will create a flat structure.

Similar to option 1, from the system perspective the VCIRS KB doesn't have any advantages of KB structure, comparing with RBS KB about KB building, inferencing and refining. Option 3 will be depicted at fig. 16.

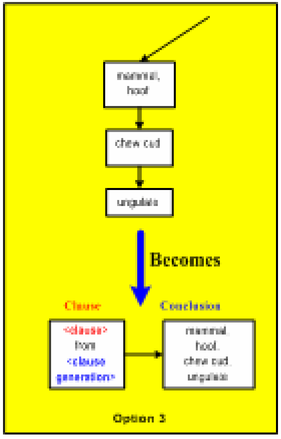

Figure 16. An example of option 3

Option 4

The case will be obtained is only one node containing with conclusions (i.e., from all nodes, except the first one, see fig. 12) and having a clause (i.e., the first node at fig. 12). Either in the RBS KB or VCIRS KB, it will create a flat structure. Again, it can be seen that from the system perspective there's not any advantages in VCIRS KB comparing with RBS KB, in the term of KB structure, building, inferencing and refining.

From the perspective of the user, this option is also the simplest one. User doesn't need to generate cases. All she needs is simply just pick up all the presented nodes in the existing tree and then proceed. This option will be described at fig. 17.

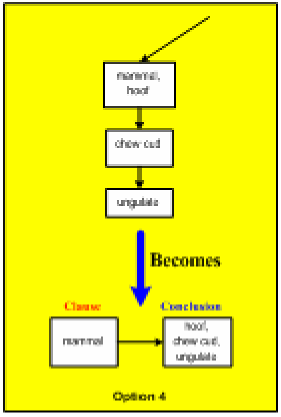

Figure 17. An example of option 4

Option 5

This is the second most difficult to handle. System needs to know when a node has to be either a clause or a conclusion. It should be careful to choose, because one picks an existing node as a conclusion (i.e., it created a node in cases reconstructing) then in the following path there will be another new node will be created too.

There will be impact in the system perspective, since VCIRS KB will get the advantage from it through the values obtained (i.e., variable and value analysis which are further influent KB structure, building, inferencing and refining) [Sub05].

The more serious circumstance will be occurred on the user perspective as consequence, if option 5 is chosen. It's very difficult for user to choose which one an existing node to be either a clause or a conclusion. The greater node user picks the wider coverage the KB will be, and vice versa. However, it also means the limited conditions for each node (i.e., clauses) will be impliedly. In other hand, the fewer clauses in a node make such node has more possibility to fire/execute.

There're 2 approaches for node separating to be either clause or conclusion. The simplest one is trying and error. The user can be trying to separate such nodes manually, and then let see what's happen. There's no rule about where's the cut line has to be laid. The complex one is trying each combination possibilities. Of course by trying and error approach, it will consume time before get the best result. So that it rather to employ optimization method such Genetic Algorithm (GA) to overcome such problem, than manual method by user trying. However, in this stage, some researches still in progress before prominent result can be obtained.

Note that it should be added a marked clause if in a path there's a conclusion node being reconstructed and then followed by the other clause node being reconstructed as described at fig. 18.

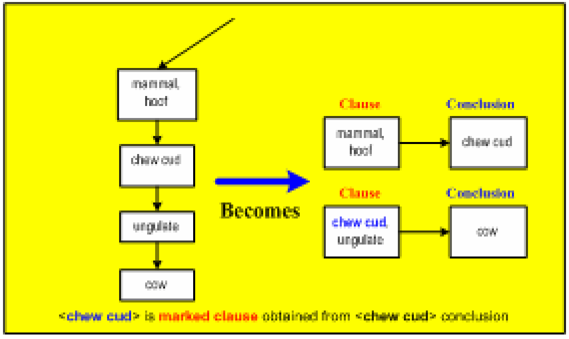

Figure 18. A marked clause for option 5

Option 6

Option 6 is the most difficult to cope. This option is option 5, described earlier added by clause and conclusion generation which are also used in option 2 and 3. An example is depicted at fig. 19 below.

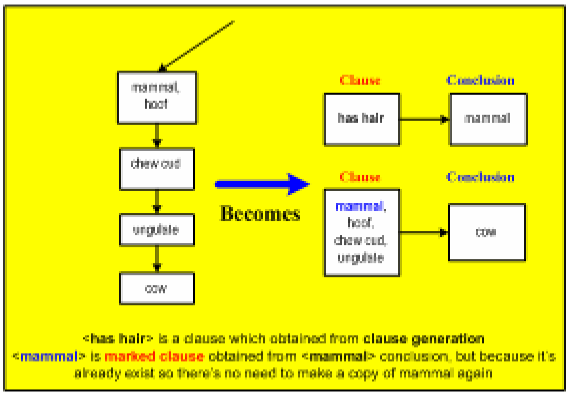
mammal, hoof
chew cud
ungulate
cow
Becomes
Clause
Conclusion
has hair
mammal
Clause
Conclusion
mammal, hoof, chew cud, ungulate
cow
<has hair> is a clause which obtained from clause generation
<mammal> is marked clause obtained from <mammal> conclusion, but because it's already exist so there's no need to make a copy of mammal again

Figure 19. An example of option 6

By understanding all the options above, it's easier to cope with any tree/diagram of a domain when it needs to transform it into VCIRS KB.

5. Conclusion

After VCIRS deployed, there're many things have to concern. Here, in this paper the things separate into 2 kinds. First is about the revisions VCIRS has to make. It's about creating the list of difference (LD) for clearance in the KA process, and the addition of options that the user can be chosen easily and rapidly during KB building. Providing LD for VCIRS will dramatically changes the convenience during KB building and an addition of options in KB building will make the structure of VCIRS KB can be realized.

The second is about cases reconstructing perspectives, i.e., from the system and from the user perspective. The input is a tree/diagram of a domain, the process is cases reconstructing and the result is a KB (i.e., either in RBS KB or VCIRS KB point of view). There're 6 options, an opportunity a case can be reconstructed, starting from the simplest one to the complex one. The choosing of these options will impact prominently both in the user and the system. The user, then, has to be concern about the approach she has to be chosen after all. By understanding this, it will make easier the transformation from a tree/diagram of domain into VCIRS KB.

6. References

[Baur90] Baur, G.R. and Pigford, D.V., *Expert Systems for Business: Concepts and Applications*, Boyd & Fraser Publishing Company, Boston-USA, 1990.

[Comp93] Compton, P., Kang, B., Preston, P. and Mulholland, M., "Knowledge acquisition without analysis," *Proc. of European Knowledge Acquisition Workshop*, Springer Verlag, 1993.

[Kang95] Kang, B., P. Compton and P. Preston, "Multiple classification ripple down rules: evaluation and possibilities," in *Proc. of the 9th AAAI-Sponsored Banff Knowledge Acquisition for Knowledge-Based Systems Workshop*, Banff, Canada, University of Calgary, 1995.

[Sub05] Irfan Subakti, "A variable-centered intelligent rule system," *Proc. of the 1st Annual International Conference: Information and Communication Technology Seminar 2005 (ICTS2005)*, Vol. 1, No. 1, August 2005, Institute Technology of Sepuluh Nopember (ITS), Surabaya, Indonesia, 2005, pp. 167-174.

[Win92] Winston, P.H., *Artificial Intelligence*, Addison-Wesley, Third Edition, 1992.