and *distance-to-travel* to those present in the working memory, as pictorially illustrated in Fig. 6. Current agenda: empty. The inference process is complete. The results are equal to those obtained if the order of firing rules would be 1, 2, 3, and 4. It is easy to check, that any order of rule firing (for example 1, 3, 2, 4, 4, 3) will give identical results.

V. Conclusion

In this comment, I described a way the FuzzyCLIPS addresses the issue of allowing fuzzy evidence to be aggregated when fuzzy inferences are made about the same fuzzy variable by different rules when an expert system works in the fuzzy logic controller regime. It seems to me that the much simpler way to handle the abovementioned difficulty in the fashion described in the cited paper would be allowing the agenda to work as a queue (a FIFO data structure).

References

[1] FuzzyCLIPS Version 6.04a User's Guide, Nat. Res. Council, Ottawa, ON, Canada, 1998.

[2] Z. A. Sosnowski, "A fuzzy extension of CLIPS rule-based shell," in *Proc. Int. Fuzzy Engrg. Symp. Human Friendly Syst.*, vol. 1, Yokohama, Japan, Nov. 1991, pp. 503–512.

[3] ——, "Representation of fuzzy data in RETE network," in *Proc. 7th Int. Fuzzy Syst. Assoc. World Congress*, vol. II, Prague, 1997, pp. 504–508.

Authors' Reply

Guilherme N. DeSouza and Avinash C. Kak

***Index Terms*—Expert systems, fuzzy expert systems, fuzzy logic, fuzzy reasoning, fuzzyshell, Rete networks.**

We would like to thank Dr. Sosnowski for his interest in our work on FuzzyShell. While he has made some cogent observations about FuzzyShell, his main conclusions about our work are ill founded. We believe that his erroneous conclusions were caused by his misunderstanding the difference between *rule firing* and *evidence aggregation* in our work.

Referring to FuzzyShell, Dr. Sosnowski states:

> "*. . . preventing to trigger another rule before firing of all rules that contain the same linguistic variable in the consequent side would violate the one of the fundamental principle of expert system (. . .) namely rule priority.*"

Dr. Sosnowski is wrong. FuzzyShell does *not* demand that all the rules that address the same linguistic variable in their consequents be fired before any other rule with that linguistic variable in its antecedent is allowed to fire. The main paragraph in the left column on p. 578 of our paper makes clear the distinction between rule firing and evidence aggregation. While all the rules that address the same linguistic variable in their consequents are used for evidence aggregation, *only one of them is fired*. The choice of the rule that is fired is dictated completely by the conflict resolution strategy used.

Manuscript received August 22, 2000.

The authors are with the School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN 47906 USA.

Publisher Item Identifier S 1063-6706(00)10680-0.

The difference between evidence aggregation and rule firing goes to the very heart of FuzzyShell. We would like to reproduce here some of the ending sentences of the main paragraph in the left column of p. 578 of our paper.

> "*Therefore, as far as fuzzy inference is concerned, we may think of rule-1 and rule-2 as being bundled together. But note that bundling together does not imply that rule-2 should be fired immediately after rule-1. In general, the consequent side of a rule (such as rule-2) will include action elements that are nonfuzzy; the inference process with regard to these action elements must proceed in the traditional manner.*"

To further explain this distinction between rule firing and evidence aggregation as implemented in FuzzyShell, we will use the following example:

Rule 1 IF (X is A) THEN (Y is B) and (W is D)
Rule 2 IF (X is A′) THEN (Y is B′)
Rule 3 IF (Y is B) THEN (Z is C).

Assuming that the conflict resolution strategy is set to *depth* and assuming that the facts in the working memory match the antecedents of both Rules 1 and 2, FuzzyShell will initially put Rule 1 and Rule 2 into the agenda, with Rule 2 set first for firing on account of its recency. Since the consequents of both the rules affect the antecedent of Rule 3, the fuzzy evidence corresponding to **(Y is B)** of Rule 1 and **(Y is B′)** of Rule 2 is *aggregated* into one single linguistic term that will subsequently be used to enable Rule 3. *Note, however, that this does NOT imply that Rule 1 must be fired before Rule 3*. The evidence aggregation step has no bearing on the order in which the rules are fired. In this example, where we assumed the *depth* strategy, the rules will be fired in the order: Rule 2 → Rule 3 → Rule 1. That can be easily established by observing that for the *depth* conflict resolution strategy, the term **(W is D)** in the consequent of Rule 1 is not asserted in the working memory until after Rule 3 is fired, regardless of whether W is a fuzzy variable or a crisp one.

We will now address what Dr. Sosnowski refers to as a simpler solution for dealing with rule chaining in fuzzy expert systems. This solution relies on a refinement of fuzzy evidence through a mechanism that amounts to reassertion of facts in the working memory. What he refers to as "*refining the existing value of variables in the working memory according to global contributions*" is, we believe, inherently inefficient because it relies on a reassertion of facts in the working memory after the computation of their "global contribution." This strategy entails the following: 1) rules that were previously fired be placed back in the agenda after the system calculates the global contributions and 2) fuzzy variables be temporarily defuzzified, to be "reassessed" again at a later time and then defuzzified again. Placing the rules back in the agenda due to the reassertion of facts with global contribution decreases the performance of the system by unnecessarily firing the same rule repeatedly. Intermediate defuzzification and reassessment of the fuzzy variables make the system assert erroneous fuzzy facts even if it is only for a short period of time. This can be a serious problem especially for real-time applications.

As for the unnecessary firing of rules, as the example in Section IV of Dr. Sosnowski's comment shows, a rule that depends on two other rules (Rules 3 and 4 in that example) will be fired twice instead of just once in FuzzyShell. In general, if m rules depend on n other rules, $n*m$ rules will be fired in the proposed system, as opposed to $n+m$ rules in FuzzyShell. This difference in the number of rules fired becomes even larger if we consider longer chains of rules. For example, m rules that depend on n rules which also depend on k other rules will cause the system to fire $m*n*k$ times, while it would fire only $m+n+k$ times in FuzzyShell.

Finally, we would like to take issue with Dr. Sosnowski's observation that expert systems are based on a set of fundamental principles. Expert systems are *not* based on a set of *fundamental principles*. In contrast with other more formal branches of knowledge such as probability theory, various forms of logic (propositional logic, predicate calculus), etc., that are based on a set of axioms, the branch of knowledge that is called expert systems is based essentially on rules of thumb and common sense, especially when it comes to issues such as rule-firing order, representation of uncertainty, nonmonotonicity, etc. It is of course possible to design toy expert systems that could be called well-principled. Such systems could, for example, adhere strictly to the principles of Baysian belief networks, fuzzy logic, or nonmonotonic reasoning, etc. But all expert systems that have ever been devised for dealing with large and complex real-life scenarios depend heavily on heuristics. To even ensure that the underlying knowledge based is consistent is often not possible in practice.