Practical Application of Fuzzy Logic: Experiences and Current State in Software Engineering

Norbert Sram*
* Óbuda University, PhD student, Budapest
*norbert.schramm@gmail.com, sramm.norbert@phd.uni-obuda.hu

Abstract — **Fuzzy logic has become an important tool for number of different applications ranging from the control of engineering system to artificial intelligence. Practical applications of fuzzy logic pose a unique set of problems. The design of systems, which apply fuzzy logic to make use of human knowledge and experience, is a daunting task without facing engineering problems of real world systems. In this paper the author describes some of the major issues faced in the process of designing, developing and applying fuzzy logic in real-world applications. The author presents a software solution which is designed to bypass these issues and improve the process of working with fuzzy logic by applying modern software engineering principles.**

I. INTRODUCTION

It has been proven that fuzzy logic [1, 2, 3, 4, 5] is a viable alternative to other algorithms in many fields. Yet the adoption of fuzzy logic in most cases is not as widespread as it could be. The reason is that the choice of using fuzzy logic to solve a problem in a specific domain has its own challenges. The first step is to build a fuzzy model from the expert knowledge available for the problem domain. This phase includes the design of the fuzzy solution. After the design and prototyping phase, the solution needs to be integrated into the production environment, the system which provides the fuzzy logic based algorithm with inputs and applies the output of the fuzzy logic solution. The integration phase can involve a series of problems, mostly depending on the target system. It is rare that the prototyping and the design of the fuzzy logic based algorithm can be done directly ported to the final system. Even from the high level view of the steps involved in the development of the fuzzy logic solution, it can be seen that involves steps which slow down the development and add a significant learning curve to the process. This is a factor which hinders the adoption of fuzzy solution in different fields.

II. ISSUES

In this section the author describes some of the issues faced in fuzzy logic based projects [6]. A general overview of the issues with the use of fuzzy logic:

1) Risk

There is a risk involved when designing a project and deciding to use fuzzy logic. This includes the previously mentioned phase of designing of the fuzzy logic solution for a specific algorithm. Modeling a domain specific problem in pseudo code is easier then achieving the same in a fuzzy logic description. It is highly unlikely that the domain expert has knowledge is/or an expert in fuzzy logic as well.

2) Existing tools not designed for production usage

Most tools available for developing fuzzy logic solutions have decent support for designing the algorithm. Some even provide great support for debugging and fine tuning those. The problem is that the development environment is rarely the same as the production environment, where the fuzzy logic solution will be applied. Also, deployment of the solution is not taken into consideration.

3) Learning curve and additional complexity

All professionals working on the project need to be aware of the fact that they are using fuzzy logic algorithms to solve a problem. This issues is present because the lack of appropriate tools and solutions. Other alternatives only require the professionals to learn a simple set of rules (for example an expert system) or a descriptive format which modifies the behavior of the algorithm. Fuzzy logic could be used the same way with appropriate support.

4) Customization and extension issues

Most tools available for fuzzy logic development involve a set of available fuzzy logic functions (operators, defuzzification, etc.) but they do not provide a straightforward way to add additional functions. In some cases there is no possibility to add custom functions to the solution. This can be a serious issue when a used function is not supported by the fuzzy logic implementation. This is a realistic case in most production cases.

5) Issues of modern software

In modern hardware and software environments the problem of concurrency and scaling became an important topic. The ability to trace the execution of an application is mandatory. Debugging step by step is no longer possible or helpful in every application domain. In clustered, multi-threaded and multiple concurrent user applications the classical debugger is not a viable solution.

978-1-4577-0045-3/11/$26.00 ©2011 IEEE

III. Fuzzy Logic and Software Engineering

In the phase of designing a fuzzy logic solution the designer is not concerned about the integration environment. The main focus is on achieving the best possible results in a test or simulation environment. The main problem with this approach is that it introduces additional risk factors if the development environment and/or platform used for creating the fuzzy solution differs from the target/integration platform. Problems can be related to unsupported operators, differences in the arithmetic model of the two platforms and so on. All these issues are risk factors in the realization of a project. These issues can be resolved by using the same environment for development and production. In this context an environment covers the fuzzy logic tools (operators, membership functions, defuzzifiaction methods, etc.) and the software realization of the fuzzy logic tools, which include arithmetic operation and other factors which contribute to the correctness of the application. The unified environment - usable for development and production as well - is a reasonable solution which builds on basic principle used in software development, abstraction. It would eliminate the risk of migration from development environment to production. Also, all used fuzzy operations would be supported without additional effort. Correctness can be further insured with the usage of automated testing.

A proper abstraction for a fuzzy logic solution would be to use it as a domain specific language for the algorithm. One of the major advantages of fuzzy logic is that it is definitions resemble human reasoning. A domain specific language [7] can be built together with the domain expert to minimize the learning curve and reduce complexity involved in the solution. In this approach membership functions are used as a layer of abstraction for the algorithm. For this approach to work a proper support is needed for the fuzzy logic implementation. By proper support the author means an interface, which makes it possible to declaratively create a domain specific language, which can be used by the domain experts. It is important to abstract away the concept of fuzzy logic to minimize the initial learning curve and to increase productivity. From software engineering perspective this is one of the easier ways to adopt and use fuzzy logic, where only the designer of the DSL (fuzzy logic based approach) needs to have knowledge of fuzzy logic.

The execution of a fuzzy logic solution in production is usually a black box. The surrounding software solution provides the inputs and receives the output after the processing. The issue with this is that there is no way of accessing partial results. If a result of the fuzzy logic algorithm is incorrect or differs from the development environment results, there is no simple way to pinpoint the source of the error. There is no way to trace the steps the fuzzy logic solution does to get to the final result. The issue might be reproducible with the appropriate steps, but there is no guarantee and it still requires an effort. This is a mandatory requirement in modern software environments which serve numerous clients all around the clock.

IV. Fuzzy Logic Solution

The mentioned issues in the previous sections are not an issue in many use cases, but still these are important problems which need to be taken into consideration. Fuzzy logic solutions have the possibility to efficiently solve the problem of concurrent programming in modern software environments. By definition a fuzzy logic decision always produces the same output for the same inputs. This means that it is referentially transparent. This property is true for all of the steps taken to reach the final result. This means that with a proper implementation for fuzzy logic tools the decision steps can be executed concurrently to take advantage of modern hardware. The designed fuzzy logic based DSL can be described as a declarative approach to domain logic development which takes advantage of modern hardware environments. This is a significant advantage. There is no need to design the algorithm with parallel [8, 9, 10] execution in mind. The DSL takes care of it where the dependencies make it possible.

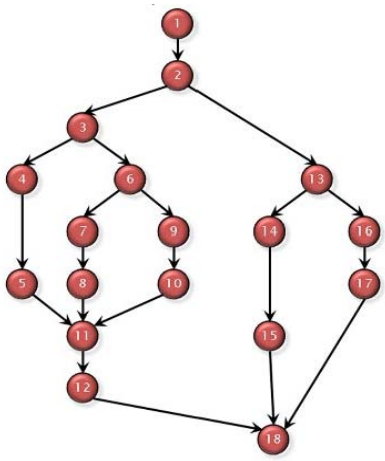

Figure 1. Execution graph

Based on the dependencies, which include partial inputs, partial results and so on it possible to build an execution graph [11]. The execution graph is the workflow for the fuzzy logic decision implementation. Figure 1. shows how an execution graph can look like. In case of a fuzzy solution we know ahead of time some initial dependencies. For example all fuzzy rules can be executed concurrently and their results collected. Fuzzy operators and aggregations can be concurrently processed with data parallel algorithms [12, 13].

The fuzzy logic solution proposed by the author is referentially transparent and based on persistant data structures [14]. The advantages are that no race conditions can occur, no need for synchronization between execution threads, also all calculations are done in a different context, meaning every partial and final result is available.

V. Requirements Based on Experience

In this section the author highlights the main requirements for a modern fuzzy solution based on his experience:

1) Compatibility

Import and export from/to other formats. This reduces the effort required for migration process. The supported formats might contain restrictions but provide an interface between different solutions.

2) Completeness

Support for multiple inference schemes, operators, membership functions etc. Most fuzzy logic libraries lack the essential functions for in depth examples or research topics.

3) Correctness

Extensive test coverage to verify correctness of basic building blocks for fuzzy logic applications. If the essential building blocks are proved correct the risk of incorrect results from the fuzzy logic solution is reduced. Also, migration and integration checks are automatized.

4) Ease of use

The basic principle is that simple things should be simple, complex operations should be possible. Component based extensions through provided interfaces. The steps and components of fuzzy logic need to be reflected in the implementation as well.

5) Tools

Provide utilities for monitoring, tracing, improving fuzzy logic solutions.

6) Knowledge and documentation

The fuzzy solution should be usable for both research and development. Both fields have their own requirements and both fields complement each other. The fuzzy solution should take advantage of this fact.

VI. Conclusion

Fuzzy logic solutions need to adept to the requirements of modern software in order to remain a viable approach for the broad field of applications. Fuzzy logic is a complex topic which requires proper software based solution to achieve efficient usage in real life use cases. In some cases, hiding this complexity is the key to success. In other cases by embracing its possibilities and advantages modern software engineering problems can be conquered elegantly and efficiently.

References

[1] G Bellmann, R.E., Zadeh., L.A., *Local and fuzzy logic,* Modern Uses of Multiple-Valued Logic, edited by Dunn, J.M., Epstein, G., Reidel Publ., Dordrecht,1977, The Netherlands, pp. 103-165.

[2] Dubois, D. Prade, H., *What are fuzzy rules and how to use them*, Fuzzy Sets and Systems 84, 1996, pp. 169-185.

[3] Dubois, D. Prade, H., *Fundamentals of fuzzy sets,* The handbook of fuzzy sets series, Kluwer Academic Publishers, Boston, 1999.

[4] Fullér, R., *Fuzzy Reasoning and Fuzzy Optimization*, Turku Centre for Computer science, TUCS General Publication, N0 9, September 1998., ISBN 952-12-0283-1.

[5] Marta Takacs, Approximate Reasoning in Fuzzy Systems Based on Pseudo analysis and Uninorm Residuum, Acta Polytechnica Hungarica, Volume 1, Issue Number 2, 2004. , pp.49-62.

[6] Norbert Sram, Marta Takacs, Multilevel Risk Management Based Fuzzy Model for the Minnesota Code, SISY, 2011, pp. 471 – 476.

[7] Felienne Hermans , Martin Pinzger , Arie Deursen, Domain-Specific Languages in Practice: A User Study on the Success Factors, Proceedings of the 12th International Conference on Model Driven Engineering Languages and Systems, October 04-09, 2009, Denver, CO.

[8] Tim Harris, Simon Marlow, Simon Peyton Jones, Maurice Herlihy, Composable memory transactions, Proceedings of the tenth ACM SIGPLAN symposium on Principles and practice of parallel programming, pp. 48–60, 2005.

[9] Tim Harris and Simon Peyton Jones, Transactional memory with data invariants, ACM SIGPLAN Workshop on Transactional Computing, 2006.

[10] Anthony Discolo, Tim Harris, Simon Marlow, Simon Peyton Jones, Satnam Singh, Lock -Free Data Structures using STMs in Haskell, Functional and Logic Programming,pp.65–80, 2006.

[11] Sram, N., A hybrid approach to decision making and problem analysis, RAAD 2010, pp. 431 – 433.

[12] Erik Wynters, Parallel processing on NVIDIA graphics processing units using CUDA, Journal of Computing Sciences in Colleges, v.26 n.3, p.58-66, January 2011

[13] Sadaf Qamar , Syed Hasan Adil, Comparative analysis of data mining techniques for financial data using parallel processing, Proceedings of the 6th International Conference on Frontiers of Information Technology, December 16-18, 2009, Abbottabad, Pakistan

[14] Chris Okasaki, Purely Functional Data Structures, 1998