Man Machine Cooperation for Learning a Complex Control Task with FLIP++

Markus Bonner Wolfgang Slany*

Christian Doppler Laboratory for Expert Systems
E184/2, TU Wien, A-1040 Vienna, Austria, Europe
mailto:{bonner|wsi}@dbai.tuwien.ac.at
http://www.dbai.tuwien.ac.at/staff/slany.html#sflip

April 18, 1996

Abstract

The control of complex nonlinear processes is hard to do for human experts. In our case it is not possible to find an efficient solution for automatic control of the process. Therefore we develop a system that should use on the one hand all available expert knowledge and on the other hand learn the remaining relation within the task with a neural net. Both types of knowledge should then be combined to achieve better results by bringing together experts and AI methods.

Keywords

Fuzzy control; technical decision support; optimization; man machine cooperation; neuro fuzzy learning; historic data; steel-making application

1 Introduction

The control of a complex chemical task within the steel-making plant is difficult. It is hard for human experts to always take the optimal control decisions. Also, most of the time these experts are not available for merely controlling the task. Simple mathematical models are not satisfying because of the strongly nonlinear task and because many important influences on the process cannot be exactly measured. Often they can just be estimated from previous processes. Therefore we need a system that holds the expert knowledge and controls the process. This step would already improve the over-all efficiency of the control by having the expert knowledge available the whole time. We want to improve this knowledge by also including information that can be learned from previous decisions.

We do not want to just learn what the experts already know, but gain new knowledge and a better understanding of the process from the information we

*Please send all correspondence regarding this paper to Wolfgang Slany.

have. We do not work independently from the experts and try to imitate their behaviour to replace them, but want to give them a tool, a partner, which allows them to learn and get a better understanding of their work. They should bring their improved knowledge into the system and make the system again a little bit more efficient and reliable. There always should be a bidirectional learning process. Each part should see the other part as a kind of teacher. By helping the teacher to become more competent one has the chance to learn more from an improved teacher. So we want to build up a neuro-fuzzy system which can be used in such a man machine cooperation. Here we still need the human computer expert as connection between the task expert and computer system, at least in some steps to include the expert knowledge in the control system. We want to demonstrate that a connection of intelligent control methods and human experts brings better solutions than either human or machine can do alone.

2 The model

The system we need to build up to control such a process requires to include the expert knowledge, to learn from existing data material, and then to control the process.

In this project we will contribute the expert knowledge in the single steps of building and training the system. We try to directly use the experts as advisors when we define the structure of the final control system. A part of their knowledge will be included in each learning step. Some explicit relations that can be easier implemented in some preparation steps will be put in a data input interface to obtain values that can be directly used for fuzzy control.

In two learning steps we will use SNNS, the Stuttgarter Neural Net Simulator, to build neural nets for training tasks. The knowledge we gain from the SNNS will be translated into a fuzzy rule-base. This happens in an interface to FLIP++, a fuzzy control library.

FLIP++ was developed by us to evaluate constraints within a scheduling project for a steel plant, the *FLIP++ library [2, 3] for real-world decision making. This allows optimizing under vague constraints of different importance using uncertain data, where compromises between antagonistic criteria can be modelled. Typical application areas include scheduling, design, configuration, planning, and classification. FLIP++ (first introduced in [1]) is just one part, the basis part of this project, and can be independently used for other fuzzy inference problems. For this C++ library we developed a graphical user interface, namely the InterFLIP++ library. This library will also be used for the new project in an adapted version.

Additionally we plan to use the Gnu-plot library to graphically display the functionality of the system to have a better handle on the control system. We can use it also to display the monitoring of the process to make it easier and faster for the operator to get information about the process itself.

3 Historic Data

Not all informations we can get from the process are trainable. Some input variables are just indices or identifications for some items. We have no direct information about these items besides their references. We just know that these hidden influences are not constant and change over time. There is no information a neural net or any other learning utility can use for training a system and derive rules from.

We have continuous and sudden changes over time of the different influence represented by the IDs. From the continuous changes we know the functions determined by statistical methods. With these linear functions we can make an estimation of the actual value of the concerned variable. In some cases we can assume sudden changes and set the variable to an a-priori known default value. If this is not possible, the operator has to tell the system the occurrence of such an unpredictable change and trigger a setting of this variable to a previously determined value. These estimations can then be included into controlling the whole task together with the measurable variables.

The relevant historic data about each variable represented by some identifications are stored in a matrix separately for each ID. This historic information is then used to predict some continuous or sudden change of the concerned variable and to give an estimation of its actual value. After each use of a certain identification, the according values in the matrix are updated by the newly received values from the control task. By reason of either known linear relations or unpredictable sudden changes over time, we will use simple functions and checks to include the a-priori known expert knowledge. This estimation of the values of input variables will be implemented in the data read-in interface.

This separation of not measurable time sensitive information form the other input data allows us to simplify the learning process by neglecting the time influence on the process. So we do not have to include lots of historic data in the training. The second positive effect is to achieve a clear structure in the whole system by always treating the informations with adequate methods. First we use algebraic and boolean methods to prepare the values we cannot measure and then we use fuzzy logic [4] to control the system through a rule-base trained by neural nets.

4 Learning

The main idea is not to learn with brute force from the existing data. This would not work because of the great number of influences and the number of old data we have to learn from. Also, we need a great number of samples to cover the complexity of the task. Even if we successfully try once to learn something from the existing data, the learned model would not function well for a long time. The reason for it is the permanent change in some influences through time. We do not have direct information about this time dependency. If we would try to include enough historic information in the learning process to cover these changes over time, the learning task would become too large and complicated to be practicably solvable. To find a solution, we structure the task and the learning process.

First we find out which influences are really important and which can be neglected. This is a very important step if we want an efficient and transparent system. If we find out the relevant input variables and their relations, we have the chance to build up a system that works more efficiently than the human experts do now with statistical methods.

Then we try to get the time dependent information out of the learning process by pretending to know these values while learning. During operation the change over time is calculated with numerical methods. The used models are built up by experts at the plant through watching the process and evaluating the results with statistical methods. After this preparation of the time relevant inputs, we can use them together with the time constant values for optimizing the process. We structured the learning process into six steps as follows:

4.1 First Step: Finding out the relevant Variables

This is a very important step in the learning process. We try to find out which variables are relevant for the result and which can be neglected with minimal effect on the result. Here we can reduce the problem by about 80% in this case. Like in the whole project, we try here to use as much expert knowledge as possible to shorten the learning process and always have some control about the direction in which the learning process is developing. At the same time the experts themselves can instantly learn from our results, even during the learning process.

We start with the influences the experts told us and checked the relations between them with neural nets created with SNNS, the Stuttgarter Neural Net Simulator. Thus we can find out the importances of single input variables.

4.2 Second Step: Building up a Hierarchy

In this step we try to build up a hierarchy with the relevant variables. This is done according to their influence on the process. The most important variables stand in a higher level of this hierarchy than less significant influences. Another rule for the structuring is the relation between the single influences. Related variables like chemical elements or weights are put together if they are of similar importance for the process. Thus is built up a structure with the aim to gain simpler rules with less variables. This should make the resulting rule-base easier to read and to understand for human experts.

As in the first step we try to use expert knowledge to build up a structure, that we cannot learn from the data. At the same time we give the experts a feedback of what we learned until now and, vice versa, we have a check by the experts. This step is not done automatically. We just follow the importances learned in the first step and some plausibility rules. If we would need this step more often, we could implement some tools that support the building up of this hierarchy according to the information from the variable evaluation. But it would be hard to automatically include the expert knowledge if we have no expert system that has this knowledge and supports our problem.

4.3 Third Step: Dividing Variables into Linguistic Terms

Dividing the single variables selected by the first step into a number of linguistic terms is the next step. This step is supported by the FLIP++ library by using predefined variable types and adjusting them to the actual needs using parameter sets. Here we try to use as few as necessary terms to keep the effort in the following steps relatively small and build up a system that does not include more complexity than the task itself. According to the data and the insights of the experts, the membership functions for the single linguistic terms are created. The main reason why we do not want to learn the membership function, at least for the time being, is the correspondence from the membership functions to the experts imaginations concerning the represented linguistic terms. The experts are the human beings, who then most likely read the resulting rule-base. The terms used in the rules should as closely as possible represent the human sight of these linguistic terms. This is necessary to understand the rules and gain new knowledge out of the trained neuro fuzzy system.

4.4 Fourth Step: Creating all possible Rules

After we have build up a representation of all relevant variables in form of fuzzy variables with linguistic terms and corresponding membership functions, we create a rule-base using these variables. We create all possible rules that can be built with the variables we have. According to the hierarchy from step two, we build rules as combinations of all linguistic terms of all variables for each layer and branch of the hierarchy. The nodes in this hierarchy are intermediate variables designed according to the variables they are derived from. The hierarchy with its intermediate variables reduces the number of possible rules by putting intermediate results together. In the next level we have just one and not many variables to consider when building all possible rules.

4.5 Fifth Step: Classifying the Rules

From all created rules we have to select only those which have a representation in the existing data. Therefore we need to classify all these rules and then delete the rules with no strong representation in the data. For this classification we have an importance field in each rule which holds the information about the credibility of this single rule. Here we use a value ranging from 0 to 1. 0 means there is no reason to trust this rule and 1 means that this rule represents the relation of the included variables. Values in between indicate a partial trust in this rule.

Now we have to find out which rules of all possible rules with the relevant data are a representation of actual data material. For all rules, which are a representation of given data, the importance is set to the relation between given data and linguistic value of the conclusion part of this rule. This can be done by using the difference between the fuzzified resulting value and the membership value of the linguistic term of the rule. To improve the performance and not implement a new learning method, we will just develop an interface to SNNS. We use this interface to build up an adequate representation of our fuzzy rules as a neural net. We train this net with the collected information about the process. The trained net is then retranslated into fuzzy rules. This means that all rules receive an importance value representing the credibility of each single rule according to the data material.

4.6 Sixth Step: Selecting the more reliable Rules

We now have lots of rules where most importances still are set to some default value, which means that there have been no examples in the data saying anything about this rule. And there are rules with different importance values in the range of 0 to 1 which are learned by a neural net using the collected information from the process.

To build up a working fuzzy system we have to eliminate all irrelevant rules. We can do this by selecting only rules with an importance above a certain value.

If we take for example a value of 0.7 as threshold, only rules with a learned reliability of more than 70 percent are accepted and all others are neglected. The threshold determines the number of rules we then have in the rule-base. It should be selected in a way such that the number of rules is kept as small as absolutely necessary and still a more or less continuous and smooth functionality of the whole system is given. The interpolation of the examples in the data material should be soft, smooth, and defined in the whole work space.

5 Conclusion

We now have to prove that our idea works, that the combination of human experts and AI methods delivers better results and more efficient solutions for complex problems. After this control problem we can try this idea on other projects within the *FLIP++ library, for instance on management problems. One could be personal planing for which a company already showed interest.

To design and implement some kind of expert system which manages the man machine cooperation, the information flow from the expert to the machine can be a next step.

We hope to soon be able to report about the implementation and installation of this system and about the successful cooperation of artificial intelligence and human experts to improve the control of a complex process on the one hand and the understanding of it on the other hand.

References

[1] Markus Bonner, Stefan Mayer, Andreas Raggl, and Wolfgang Slany. FLIP++: A fuzzy logic inference processor library. In *Workshop Notes of the IJCAI'95 Workshop on Fuzzy Logic in Artificial Intelligence*, pages 83–92, Montreal, Canada, August 1995. URL: ftp://ftp.dbai.tuwien.ac.at/pub/papers/slany/ijcai95wsflai1.ps.gz.

[2] Wolfgang Slany. Fuzzy scheduling. CD-Technical Report 94/66, Christian Doppler Laboratory for Expert Systems, Technical University of Vienna, 1994. URL: ftp://ftp.dbai.tuwien.ac.at/pub/papers/slany/cd-tr9466.ps.gz.

[3] Wolfgang Slany. Scheduling as a fuzzy multiple criteria optimization problem. *Fuzzy Sets and Systems*, 78:197–222, March 1996. Issue 2. Special Issue on Fuzzy Multiple Criteria Decision Making; URL: ftp://ftp.dbai.tuwien.ac.at/pub/papers/slany/fss96.ps.gz.

[4] Hans-Jürgen Zimmermann. *Fuzzy Set Theory — and Its Applications.* Kluwer Academic Publishers, 2nd, revised edition, 1991.