C51

THE IMPLEMENTATION OF AN EXPERT SYSTEM FOR PROCESS IDENTIFICATION *

Marc HAEST, Georges BASTIN, Michel GEVERS and Vincent WERTZ
Laboratoire d'Automatique, de Dynamique et d'Analyse des Systèmes
Université Catholique de Louvain
Place du Levant 3
B-1348 Louvain-La-Neuve, Belgium

Abstract - The purpose of the paper is to illustrate on a simple example how a branch-and-bound procedure allowed us to reduce significantly the number of model structures investigated by ESPION, our expert system for process identification.

1. INTRODUCTION

System identification has never been an easy task. In addition to the data processing tools, the estimation methods and the validation criteria (see for example Ljung 1987, or Söderström and Stoica 1989), human experts use a lot of apparently empirical rules acquired through their already long experience in the area. Moreover, during an identification exercise, one has typically to choose among several alternatives at different decision levels, always revisiting earlier choices in the hope that an acceptable solution will be found somewhere, but keeping old results in mind in case something even worse is encountered. So, the situation typically belongs to the ones expert systems claim they are made for. They are intended to assist or replace man in fields where an insufficiently structured knowledge is admitted for to constitute an accurate, definite and unambiguous working methodology.

Therefore, at Louvain-La-Neuve, we started to implement an expert system in which a set of rules is used to mimic the behaviour of human experts during an identification exercise. Provided with a data set, ESPION organizes an "intelligent" search through the set of multiple input - single output structures in ARARX form, and ends up with a best model according to a quality index that incorporates a number of validation criteria. Simulations have shown that our set of rules already allowed us to identify a reasonably good model on most simulated and industrial data sets at the price of moderate computing efforts depending on the class of models to which the search is restricted (Haest *et al.* 1988, 1990a).

However, the development of our expert system also has a scientific objective since the question that is actually put is whether the heuristics that different practitioners have developed to organize their own search rules through the model set is based on experience and personal preferences, or whether a rationale and perhaps, some theoretical tools can be developed to organize this search in a more efficient way.

Indeed, when producing programs that do "intelligent" things in the same way fallible people do, we are in danger of producing nothing more than something that fails when human experts do. In other words, if we want to make the computer do the task even more efficiently than before, we should best understand the structure of the problem we want to solve. This knowledge, in turn, allows experts to expect to become more competent in the future ... provided it can be acquired. So, starting from this challenge, we investigated which contribution advanced search techniques from *artificial intelligence* and *integer programming* could provide us when faced with the problem at hand (Haest *et al.* 1990b).

Those peregrinations led us to propose a new strategy based on the so-called branch-and-bound procedure. Roughly speaking, the branch-and-bound procedure is a heuristic search, with an estimate of the remaining distance to a goal state, combined with the application of any particular domain specific technique that allows to prune the search tree judiciously.

*The results presented in this paper have been obtained within the framework of the Belgium Program on Concerted Research Actions and on Interuniversity Attraction Poles initiated by the Belgium State, Prime Minister's Office, Science Policy Programming. The scientific responsibility rests with its authors.

In this paper, after having briefly recalled how the expert system behaves in section 2, we will show that branch-and-bound can help to solve the identification problem and how its principles have been used to modify our original set of rules. However, we will have to say a few words about search in Section 3, before discussing our procedure in Section 4. Finally, its typical performances will be illustrated on a simple example and compared to the performances of the original set of rules in Section 5. Some concluding remarks are made in Section 6.

2. ESPION

The overall behaviour of ESPION can be roughly summarized by the following three-step loop (Haest *et al.* 1988, 1990a):

1. parsimonious walk through the model set,
2. selection of the best models according to a quality index,
3. if no satisfactory solution has been found, sampling period modification and repetition from step 1 until a "good enough" model is obtained.

Once the first step has been completed, the expert system is left with a set of presumably good models. To select the best one(s), a *quality index* is attached to them. It is incremented by one during the second step each time the model satisfies one criterion from a list of specified *validation tools*. The models are then sorted according to their quality indices. In what concerns the sampling period validation, a number of "rules of thumb" are tested during the three steps. If one or more of these rules are found to be violated, depending on the current stage of the expertise, the sampling period is increased and the data set filtered appropriately. Steps 1 and 2 are then repeated on the new data set until satisfactory results are obtained.

However, in this paper, we shall be mainly concerned with the first step, whose aim is obviously to avoid estimating all the models in the model set as would be done in an exhaustive search. Since a model structure is specified by a set of integer numbers, say q, the first step can be thought of as a walk through the points of the space $\mathbf{Z}_+^q$ until a q-tuple is obtained for which the corresponding model is considered "good enough". We will not go into the details about this early implementation of the first step. Let us only say that, in its original version, the travel path of our expert system through the space $\mathbf{Z}_+^q$ was driven by a set of rules that attempted to mimic the way in which human experts proceed when they try to identify models from data. The principle of this walk was to start with a model of small dimension, the delays of which had been roughly estimated through crosscorrelation analyses. Then, the model dimension was progressively increased, while minimizing the number of structures investigated at each model dimension, until an *elbow* was detected in the graph representing the evolution of the best prediction error variance obtained at each model dimension with respect to the model dimension. What should be kept in mind here, is that the search was conducted from the lower model dimensions to the higher ones until the prediction error variance ceased to decrease significantly.

3. BRANCH-AND-BOUND

So, during the first step, the task of the expert system is to find a good model in the maze of all candidate structures. It turns out that search is precisely one of the major problem solving paradigms used in artificial intelligence and other areas of computer science such as integer programming and *game theory*. Every search procedure can be described as the crossing of a *graph* in which each *node* represents one particular *state* of the problem being solved and each *arc* represents available *operators* that can be used when attempting to progress from the current state of the solution process closer to a node that represents the completely solved problem. All the possible states of the problem form what is called a *state space*.

However, before solving our problem, we must decide what the states are and define the operators. We have already spoken about the identification problem as if it were a way of travelling through the points of the space $\mathbf{Z}_+^q$ in which the states merely represent model structures. In what concerns the operators, the simplest way one can think of a movement from one model structure to another is simply by adding or deleting one parameter at a time. With these two elementary operators, we are able to go everywhere in the state space no matter where the starting model structure is in that space. Of course, we should not allow our search procedure to cycle in it. With this restriction, we thus have to explore a *tree* and it is common to talk about search trees as if they were family-trees. Branches connect parents with children, the node at the top of the tree, the one from which the exploration is started, is called the *root* node, and so on.

Now, several procedures have been developed to solve search problems, one of which, called *branch-and-bound*, allows considerable flexibility when design-

ing specific algorithms for a wide variety of problems that can be encountered in practice. Let us assume that an objective function has to be minimized and that an upper bound $f_u(x)$ on its best value has already been obtained. The principle of branch-and-bound then goes as follows (see for example Hillier and Lieberman 1989). The set of all admissible solutions is first divided into several subsets. Then, a lower bound $f_l(x)$ is obtained for the value of the objective function within each of these subsets. Those subsets whose lower bound exceeds the current upper bound are excluded from further consideration. A subset that is excluded from the inquiry is said to be pruned. After the appropriate subsets have been pruned, one of the remaining subsets is chosen to be divided further into several subsets. Their lower bounds are obtained in turn and used as before to prune some of them. Another subset is selected from the remaining ones to be split again, and so on: this process is repeated until a solution is found, the objective function of which is no greater than all the lower bounds of the remaining subsets. We are sure that this solution is optimal since none of the remaining subsets can contain a better solution.

4. APPLICATION

We now show how branch-and-bound can be used during an identification exercise. The basic assumption made here is that the "good enough" structures we are looking for actually exist somewhere in our working space. On the other hand, the F-test (Ljung 1987, Haest *et al.* 1988, Söderström and Stoica 1989, Haest *et al.* 1990a, 1990b) allows us to decide whether or not a given structure is significantly better than another. The problem is then to extract an optimal solution from the set of "good enough" structures. In the following, a structure will be considered as optimal if all its underparametrizations yield a significantly worst prediction error variance, while none of its overparametrizations yield a significantly better variance. A model M_2 is called an underparametrization of a model M_1 if it is obtained from M_1 by setting some of its parameters to zero. M_1 is an overparametrization of M_2 if M_2 is an underparametrization of M_1.

With this definition, one way to implement branch-and-bound is to start in the state space with a structure of high dimension. Doing this, we are almost sure to start with an overparametrization of the structure we are looking for. Conversely, we can generate all admissible structures in the state space by deleting parameters from this initial structure. Then, the state space can be partitioned simply by generating the children of the root structure. This is done by removing parameters from the parent structure. Clearly, those children are of lower dimension than their parent structure. Therefore, their prediction error variances are all greater than the one associated to their parent structure. Moreover, those prediction error variances certainly constitute lower bounds on the best value of the objective function that can be achieved in their respective subsets.

Children that are significantly worse than their parents can be definitely excluded from further investigation together with all their underparametrizations. The other children, those that are not significantly worse than their parents, are kept in the arena. At the next step, the unpartitioned structure with the best prediction error variance is selected to be partitioned further and the whole process continues.

An overestimation of the objective function is obtained the first time a dead-end is encountered along one path of the tree. This occurs when a structure is found all the children of which are either significantly worse or already excluded from the set of admissible structures. Such a structure represents a candidate for the optimal solution. In the sequel, each time a better solution is encountered, it replaces the current solution and the upper bound on the objective function is modified accordingly. The procedure can be stopped as soon as at least one solution has been found and all the remaining unpartitioned structures have been pruned or have greater prediction error variances than the one associated with the current solution. Since we are running the risk of estimating many overparametrizations of the optimal structure before it can be discovered if only one parameter is removed at a time from parent structures, the "landscape" is first mapped at "equidistant" structures by deleting many more than one parameter at a time. As will be seen in the next Section with an example, a quick search in such a "reduced" space together with the use of the F-test give valuable information on subsets in which the "best" model should be searched for.

5. AN EXAMPLE

Branch-and-bound was run on 1000 sampled data obtained from the following seven parameter, two input - one output ARX system

$$(1 + z^{-1} + 0.5z^{-2})\, y(t) = z^{-2}(1 - z^{-1} + 0.5z^{-2})\, u_1(t) + z^{-7}(1 + 0.2z^{-1})\, u_2(t) + e(t)$$

where pseudo random binary signals with amplitude 1 (variance 1) were used for u_1 and u_2 and a Gaussian

white noise with zero mean and variance 0.25 was taken for e, leading to an output signal variance of around 14. In what follows, the notation $n_a(\tau_1\text{-}n_{b1})(\tau_2\text{-}n_{b2})$ has been used to characterize two input - one output structures where n_a is the number of parameters in the autoregressive polynomial and τ_i and n_{bi} are the positions of the first and last nonzero parameters of the i-th exogeneous polynomial, respectively.

The starting structure was chosen to be 10 (1-10)(1-10) and its prediction error variance was computed to be $\sigma^2 = 0.2494$. The structures obtained during the first step by stripping the starting one of five parameters at a time are shown in Table 1 with their prediction error variances. Since three of these underparametrizations do not differ significantly from the starting one, it is believed that the latter is an overparametrization of the structure we are looking for. The structures that differ significantly from the starting one, 10 (6-10)(1-10) and 10 (1-10)(1-5), are pruned forever (†) and the procedure is restarted from 10 (1-10)(6-10), the child with the best prediction error variance obtained so far. The four structures that should be partitioned during the second step are listed in Table 2. In fact, only two of them need to be estimated since the others have already been excluded from further consideration. For example, it makes no sense to compute the prediction error variance associated with 10 (6-10)(6-10) since this structure is an underparametrization of 10 (6-10)(1-10), which has been pruned after the first step.

models	σ^2
5(1-10)(1-10)	.252
10(6-10)(1-10)	1.52†
10(1-5)(1-10)	.251
10(1-10)(6-10)	.250
10(1-10)(1-5)	1.21†

Table 1

models	σ^2
5(1-10)(6-10)	.253
10(6-10)(6-10)	†
10(1-5)(6-10)	.252
10(1-10)(0-0)	†

Table 2

The overall behaviour of the branch-and-bound search is summarized in Table 3 where only the structures the estimation of which has been actually needed are numbered (n). The number of parameters (s_f) that are removed at a time when attempting to progress from a particular structure is given for each step (s) together with the parent structure number (f) and all the explored children (models). Model dimensions are also given (d) in the last column. During the third step, the partition of the best remaining structure, 10(1-5)(1-10), needs the estimation of only one new model. Then, two other structures appear during steps 4 to 8. However, after completion of the eighth step, we are unable to go further by stripping five parameters at a time without significantly increasing the prediction error variance. At this scale, the best approximation of the structure we are looking for is 5 (1-5)(6-10). During step 9, we now try to strip four parameters at a time from this latter structure. Since all children have to be pruned, we can jump to step 10 immediately where it is tried to strip three parameters at a time.

s	f	s_f	n	models	σ^2	d
			1	10(1-10)(1-10)	0.2494	30
1	1	5	2	5(1-10)(1-10)	0.2520	25
			3	10(6-10)(1-10)	1.5166†	25
			4	10(1-5)(1-10)	0.2507	25
			5	10(1-10)(6-10)	0.2504	25
			6	10(1-10)(1-5)	1.2075†	25
2	5	5	7	5(1-10)(6-10)	0.2531	20
			8	10(1-5)(6-10)	0.2517	20
3	4	5	9	5(1-5)(1-10)	0.2531	20
4	8	5	10	5(1-5)(6-10)	0.2541	15
5	2	5	11	0(1-10)(1-10)	0.7290†	20
6	7	5				
7	9	5				
8	10	5				
9	10	4	12	1(1-5)(6-10)	1.1298†	11
			13	5(5-5)(6-10)	1.5264†	11
			14	5(1-1)(6-10)	1.5595†	11
			15	5(1-5)(10-10)	1.2228†	11
			16	5(1-5)(6-6)	1.2240†	11
10	10	3	17	2(1-5)(6-10)	0.2555	12
			18	5(4-5)(6-10)	1.5129†	12
			19	5(1-2)(6-10)	0.5107†	12
			20	5(1-5)(9-10)	1.2208†	12
			21	5(1-5)(6-7)	0.2638	12
11	17	3	22	2(1-5)(6-7)	0.2865†	9
12	21	3				
13	17	2	23	2(3-5)(6-10)	1.2715†	10
			24	2(1-3)(6-10)	0.3844†	10
			25	2(1-5)(8-10)	1.2277†	10
			26	2(1-5)(6-8)	0.2570	10
14	26	2				
15	26	1	27	2(2-5)(6-8)	0.2572	9
			28	2(1-4)(6-8)	0.2570	9
			29	2(1-5)(7-8)	0.2574	9
16	28	1	30	2(2-4)(6-8)	0.2572	8
			31	2(1-4)(7-8)	0.2575	8
17	27	1	32	2(2-5)(7-8)	0.2576	8
18	30	1	33	2(2-4)(7-8)	0.2577	7
19	29	1				
20	31	1				
21	32	1				
22	33	1				

Table 3

Two promising structures are encountered there: the others have to be pruned. Then, only one uninterest-

ing structure is estimated during steps 11 and 12. At this stage, we are unable to go further by stripping three parameters at a time. Both 2 (1-5)(6-10) and 5 (1-5)(6-7) have all their underparametrizations significantly worse, but the former is preferred because it has the least prediction error variance. So, the best approximation of the system becomes 2 (1-5)(6-10). In step 13, we now try to strip two parameters at a time. Starting from 2 (1-5)(6-10), one promising structure is encountered, but no new structure appears during step 14. So, we enter step 15 where it is tried to strip one parameter at a time, and so on.

After completion of the whole process, the only structure all the underparametrizations of which are significantly worse, while none of its overparametrizations is significantly better, is 2 (2-4)(7-8), the structure with which the data were generated. Note that only 33 structures have been estimated. However, a few more could have been investigated if another confidence level had been used for the F-test, but what really matters is that many more could have been generated if another strategy had been used to adapt the "stripping factor" s_f. On the other hand, since it first explores model structures in low dimensions before increasing the number of parameters, ESPION was unable to exploit the pruning principle in its earlier version. Run on the same data, it estimated 149 structures before discovering the one with which the data were generated. Note, however, that an exhaustive search over all ARX underparametrizations of the root structure would have required the estimation of 30,250 structures!

6. CONCLUSIONS

In this paper, we have shown how a branch-and-bound procedure can be used during an identification exercise. The major innovation introduced here is that a quick exploration of the model set can be made by mapping it at "equidistant" model structures by deleting many more than one parameter at a time. This method, combined with the use of statistical tests, gives valuable information on subsets in which the best structure should be searched for and allows to save considerable computer time. Once a smaller subset that most probably contains the "best" model has been identified, it can be mapped in turn at a lower scale and so on until an acceptable solution has been found.

Typical performances of this search technique have been illustrated on a simple example and compared to the performances of an earlier version of our expert system. The results are quite encouraging since, when limited to ARX structures, the procedure already allows us to significantly reduce the number of trial structures explored by ESPION. Now, the procedure can be extended easily to ARARX model structures since these structures accept ARX overparametrizations of minimal dimension: before estimating an ARARX structure one should always first estimate its ARX overparametrization of minimal dimension to see whether this latter structure should not be pruned according to other models that could have been already obtained.

However, this technique cannot be applied to structures with a moving average polynomial in their noise model such as ARMAX and Box-Jenkins structures. Even if this fact does not prevent us to use branch-and-bound theoretically in this context, one should keep in mind that those structures are far more expensive to estimate than ARX ones. It is also well known that overparametrizations can give rise to numerical problems. Good heuristics are being developed here to be used in combination with the branch-and-bound procedure to robustify it, direct the search more competently and decrease the risk of making useless time consuming movements. Some classical validation tools could be useful for this purpose.

Finally, we have seen that the number of structures explored during the search is highly dependent on the strategy used to adapt the stripping factor. Investigations are currently carried out in order to optimize the updating of this important design parameter.

REFERENCES

Haest, M., G. Bastin, M. Gevers and V. Wertz (1988). An Expert System for System Identification. *Proc. 1st IFAC Workshop on Artificial Intellignece in Real-time Control*, Swansea, United Kingdom, pp. 101-106.

Haest, M., G. Bastin, M. Gevers and V. Wertz (1990a). ESPION: an Expert System for System Identification. *Automatica's Special Issue on Identification and Parameter Estimation*, Vol. 26, No. 1, pp. 85-95.

Haest, M., G. Bastin, M. Gevers and V. Wertz (1990b). On the use of Search Methodologies in System Identification. Technical Report AP90.04. Department of Automatic Control, University of Louvain.

Hillier, F. S. and G. J. Lieberman (1989). *Introduction to Operations Research*. McGraw-Hill, New York.

Ljung, L. (1987). *System Identification. Theory for the user*. Prentice Hall, Englewood Cliffs, New Jersey.

Söderström, T. and P. Stoica (1989). *System Identification*. Prentice Hall, Englewood Cliffs, New Jersey.