Available at
www.ElsevierComputerScience.com
POWERED BY SCIENCE DIRECT®

INTERNATIONAL JOURNAL OF APPROXIMATE REASONING

ELSEVIER International Journal of Approximate Reasoning 35 (2004) 307–337
www.elsevier.com/locate/ijar

Integration of intelligent systems in development of smart adaptive systems

Esko K. Juuso

Control Engineering Laboratory, Department of Process and Environmental Engineering, University of Oulu, P.O. Box 4300, FIN-90014 Oulu, Finland

Received 1 February 2003; accepted 1 August 2003

Abstract

Different combinations of fuzzy logic and neural networks provide various ingredients for smart adaptive applications. Both expertise and data can be integrated in the development of intelligent systems. Evolutionary computation is also widely used in tuning of these systems. For small, specialised systems there is a large number of feasible solutions, but developing truly adaptive, and still understandable, systems for highly complex systems require more compact approaches in the basic level. Linguistic equation (LE) approach originating from fuzzy logic is an efficient technique for these problems. Insight to the process operation is maintained since all the modules can be assessed by expert knowledge and membership definitions relate measurements to appropriate operating areas. The LE approach increases the performance by combining various specialised models in a case-based approach: models can be generated automatically from data. The LE approach is also successfully extended to dynamic simulation and used in intelligent controller design. The integration of intelligent systems is based on understanding the different tasks of smart adaptive systems: modelling, intelligent analysers, detection of operating conditions, control and intelligent actuators. The system integration leads to a hybrid system: fuzzy set systems move gradually to higher levels, neural networks and evolutionary computing are used for tuning, and the whole system reinforced with efficient statistical analysis, signal processing and mechanistic modelling and simulation.

E-mail address: esko.juuso@oulu.fi (E.K. Juuso).
URL: http://www.ntsat.oulu.fi.

0888-613X/$ - see front matter © 2003 Elsevier Inc. All rights reserved.
doi:10.1016/j.ijar.2003.08.008

Keywords: Linguistic equations; Fuzzy set systems; Neural networks; Dynamic simulation; Intelligent analysers; Intelligent control

1. Introduction

Development of smart adaptive systems for non-linear, complex, multivariable and highly interactive industrial processes is a challenging task. Usually, the important quality variables can be estimated only from other measured variables. Physical limitations of actuators must be taken into account. Significant interactions between process variables cause interactions between the controllers. Various time delays depend strongly on operating conditions and can dramatically limit the performance and even destabilise the closed-loop system. For overall production processes, control systems take care of several subprocesses. There are many and long time-varying delays, process feedbacks at several levels, closed control loops, factors that exist and cannot be measured and interactions between physical and chemical factors. Uncertainty is an unavoidable part of the process control in real-world applications since there always are some unknown factors affecting to the process conditions. Successful applications require integration of data-based methods and expertise, especially if fast reactions to changing operating conditions are needed.

Smart adaptive systems are based on intelligent methods, i.e. individual subsystems are intelligent systems. The smartness of the overall systems depends on integration of these intelligent systems. A smart system needs a decision making unit: in Fig. 1 this is the control block. Putting everything in this block would result too complicated systems. A better alternative is to make a generic and configurable control block whose inputs are (calculated) variables related system properties which really should be controlled. This can be done with software sensors type intelligent analysers. This part is especially important in connection to continuous on-line analysers. Intelligent analysers may also include trend analysers. If operating conditions are changing, intelligent analysers are also needed for detection of the operating conditions. Measurement should be handled with digital signal processing. Also operation of actuators can be improved with intelligent modelling. Dynamic modelling and simulation is needed for comparing alternatives in controller design.

On-line analysers facilitate new measurements which also need new calculation modules to be used in everyday process control. Intelligent analysers provide solutions for this, but part of the work could be done also in digital signal processing. Possibilities of using intelligent techniques are improving also in actuators. Moving close to the process brings new challenges for the implementation of the intelligent systems.

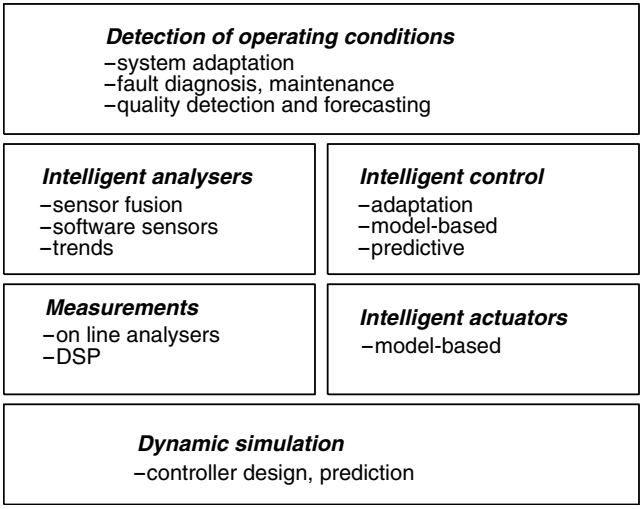

Fig. 1. Subsystems of a general smart adaptive system.

In this paper different intelligent methods are evaluated on basis of their possible contributions to the smart adaptive systems. Linguistic equation approach originating from fuzzy set systems is introduced as a integrating methodology in developing smart adaptive systems for complex applications.

2. Intelligent methods

Computational intelligence or soft computing understood as a mixture of fuzzy set systems, neural networks and genetic algorithms is an extension rather than a competitor for artificial intelligence (AI). Fuzzy logic emerged from approximate reasoning, and the connection of fuzzy rule-based systems and expert systems is clear, e.g. vocabulary of AI is kept in fuzzy logic [4]. Possibilities of the symbolic AI methodology are extended with numerical modelling, which is shared by fuzzy set systems and neural networks with an assistance of evolutionary computation, e.g. genetic algorithms. Numerical methods and symbolic approaches are both needed in computing with words.

Conjoint numeric, symbolic and neural computing will be very important in applications for complex systems. Symbolic computing has the key role in connecting applications and making them easier to access for non-specialist users. Rule-based expert systems used in AI are suited to codifying meta-knowledge of a system but rule-based systems are not easy to extend to complex systems. For applications, mapping quality to quantity can be done by fuzzy set theory. Behavioural models can also be based on neural networks

if the system is small enough. Genetic algorithms and related evolutionary strategies have recently used in many narrow applications.

2.1. Expert systems

Rule-based programming is commonly used in the development of expert systems but this paradigm leads to serious maintenance and testing problems in practical applications where rule-based systems become really massive. Therefore, linking the rule-based systems to more efficient modelling methods is essential for operability of the practical systems. Using declarative *Prolog* language (or extensions of it) on relation level can reduce amount of rules to some extent [12]. However, these simulations are still too slow for complex dynamical systems, and therefore, knowledge-based reasoning is embedded to fuzzy logic or linguistic equation applications.

In the modelling framework, rules are generated only if they are required by the programming system used in final implementation. Large rulebases can packed into a very compact matrix equations, e.g. in an diagnostical application small 7×11 matrix replaces 36,000 fuzzy rules. Methodologies used in development of expert systems are still useful in extracting expert knowledge from domain experts. Uncertainty and complexity can be handled with other methodologies. In our development and tuning system, conventional expert system approach is a part of the *FuzzEqu* Toolbox [19].

2.2. Fuzzy set systems

Fuzzy set systems provide extensions of the expert system techniques to uncertain and vague systems [40]. The traditions of physical modelling on the basis of understanding the system behaviour are continued with fuzzy rules and membership functions, which can represent gradually changing non-linear mappings together with abrupt changes. Various approaches using either expertise or data are used in constructing these mappings, but the increasing complexity of the application require more and more combined approaches classified to the upper right corner of Fig. 2. Heuristic knowledge and know-how can be introduced to the fuzzy set systems by a trial and error based approach. Data-based approaches rely usually on automatic generation of rules from predefined simple sets of membership functions. Both knowledge and data need to be used together in developing practical applications. Linguistic fuzzy models [2] are mostly used in the knowledge-based approach, and Takagi–Sugeno (TS) type fuzzy models [37] and fuzzy relational models [34] for data-driven methods.

In the knowledge-based approach, understanding of the rules is important, and the number of rules is tried to keep in minimum. The membership functions are also based on domain expertise. Non-linear behaviour of the system

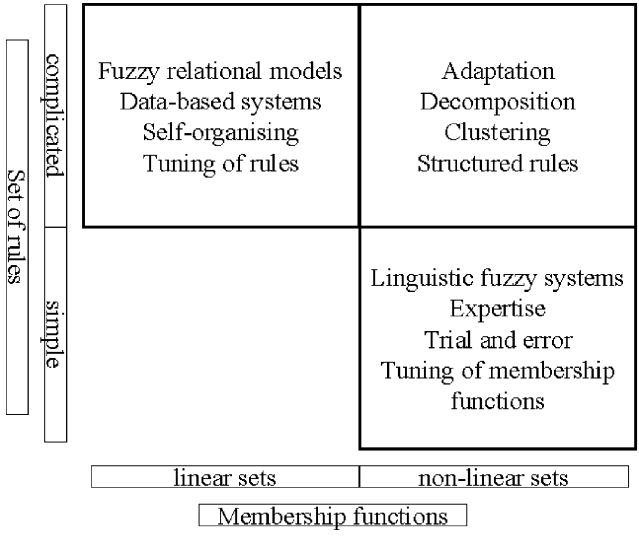

Fig. 2. Classification of fuzzy set systems.

should be included as much as possible in the distribution of the membership functions. The actual functions can be constructed from linear or non-linear parts. The knowledge-based approach is based on non-linear sets of membership functions and a fairly simple set of rules (Fig. 2). The operation of the system can be improved by tuning of the membership functions. This approach is well suited to human input. First systems operate fairly well but improving the system is time consuming as numerous parameters affect to the resulting system. Genetic algorithms suit well for tuning the gradual rule-based systems to approximate non-linear mappings between input and output signals, e.g. a strict fuzzy partition and triangular membership functions provide a basis for an efficient coding [32].

Constructing fuzzy models from data with the best possible accuracy alleviates the knowledge acquisition problem. Table-lookup scheme [38,39] is an example of algorithm for generating fuzzy rules. Fuzzy rule generation can also take into account contradictory data [28,29]. The data-driven approach leads to a black-box models if the interpretation of results is not addressed sufficiently. As understanding of the system is very important benefit, approaches relying on linear sets of symmetrical membership functions and complicated rule sets are not solutions for a smart adaptive system. Self-organising fuzzy approach is a good technique if the key issue is to find the parameter values for a new operating point, e.g. good results have been obtained with self-organising linguistic fuzzy controllers in medical applications.

The time needed for reliable on-line adaptation can be a problem in process control, e.g. varying operating conditions may destroy the performance completely. If the adaptation is needed only in the beginning of the operation, tuning time is not a problem. Black-box approaches result rather complicated rule bases, as the sets of membership functions are usually very simple (Fig. 2). A set of evenly spaced symmetrical membership functions is a common starting point. However, a complicated rule base means high overfitting risk since the model surface can adopt also noise or minor disturbances. Adaptation of these systems requires always a new development phase, which becomes difficult if operating conditions change continuously. Continuously changing system does not support accumulation of domain expertise.

Fuzzy models can be considered as a class of local modelling approaches, which attempts to solve a complex modelling problem by decomposing into number of simpler subproblems [1,2]. Understanding can be maintained in these specialised subsystems as both steady state and dynamic non-linear systems can be approximated in a fairly compact way. However, a huge number of parameters need to be tuned and updated when the process conditions change since the number of subsystems is very high. Changing operating conditions can be handled with a multimodel approach, and different clustering method can be used for finding suitable areas for modelling. Grid partitioning, which divides each linguistic variable into at least two linguistic terms, is not satisfactory for large-scale multivariable systems as too many modelling areas result overfitting.

Takagi–Sugeno (TS) type fuzzy models [37] are widely used for this purpose since the neuro-fuzzy ANFIS method [8] provides an efficient tuning method for these models. The smoothing problem around submodel borders can be handled with special techniques, e.g. smoothing maximum [2], or by making the area overlap very strong. Fitting results are very good with strongly overlapping models, but the process insight is lost as the individual models do not have any meaning. The ANFIS tuning increases overlap of the clusters and destroys meanings of the individual linear models, e.g. the role of some submodels may transform into a part of a smoothing algorithm. Sharper borders require non-linear consequent models, i.e. smoothing should be a part of each individual local model.

The classification depicted in Fig. 2 is mainly based on the ideas of numerical computation with fuzzy rules and membership functions. Fuzzy logic provides efficient, understandable tools for interpolation with a set of gradual rules, but fuzzy logic is primarily meant for multiple-valued logic and approximate reasoning, and approximate reasoning is a subtopic of artificial intelligence. These issues are important in higher levels of smart adaptive systems (Fig. 1), e.g. detecting operating conditions and fault diagnosis will be discussed in the application section.

The most important feature of fuzzy systems is that they provide clear explanation of the operation. However, data-based techniques are needed to make the tuning phase practical. How to reach both goals? An understandable set of structured rules and non-linear membership functions are essential for practical fuzzy systems. Selecting a suitable number of modelling areas is the key issue.

2.3. Artificial neural networks

Artificial neural networks (ANN) have been used as behavioural input–output models, which are difficult to connect to other models of the system [14]. Possibility to a more or less automatic modelling has increased popularity of neural networks, and really a huge amount of software tools is available for neural computing.

The most popular neural network architecture is the multilayer perceptron (MLP) with very close connection to the backpropagation learning [35]. Various optimisation methods, e.g. conjugate gradient, quasi-Newton and Levenberg–Marquardt, have been used in these networks to speedup learning. Since overfitting is the main problem, the generalisation is tried to improve for example with ensemble learning with early stop [30] and regularisation techniques [5]. Selecting the optimisation method is an important issue as very fast tuning may even prevent the validation stop. Precision is fairly good for small well-defined systems but problems arise in complex systems. Actually only small fragments of the overall system can be modelled at a time. Since fully representative data is very hard to get, ANNs can be recommended primarily to system development. A feasible approach is to generate a fuzzy set system from several specialised neural models. The ANFIS method based on the backpropagation method is an example of this generic methodology.

In addition to the widely popular feed forward networks, various useful methodologies, e.g. self-organising maps and radial basis networks, are used in applications. Self-organising maps (SOM) [26] can be used for finding operating conditions or simply for clustering. Radial basis networks [3] provide an interesting alternative as they can be used both as a clustering tool and a modelling tool. There is also a clear link to fuzzy set systems, i.e. both SOM and radial basis neurons can be represented as fuzzy rules. Radial basis models fill well the whole operating area, and fairly good results can be obtained if an appropriate number of neurons are used. Models-based normalised data use symmetric activation functions and a linear layer combining the neurons. For SOM-based models, pruning the rules is necessary since there are usually very many neurons close to the average operating conditions. On the other hand, additional rules are needed for boundary areas.

Connecting ANNs to other modelling techniques is vitally important as far as complex systems are concerned. A solution might be a neuro-fuzzy

approach, which makes the model more understandable. Neural computing provides a suitable identification method for working point adaptation if the generalisation aspect is taken into account.

2.4. Genetic algorithms

Genetic algorithms (GA) were long time considered to belong to general systems theoretians interested about extremely complex optimization studies [36]. GAs can be considered as an experimenting tool which produces a satisfactory solution which is not necessarily optimal. GAs are useful when this can be accepted, e.g. very large search space with noisy data and uncertain models. The algorithm is a reasonable way for processing population of alternatives. The crucial thing is to use appropriate fitness functions, i.e. they should be application dependent. Massive parallel processing is required in real applications.

Genetic algorithms can assist other methods of computational intelligence by optimising structures. For expert systems, optimisation of the rule base is a possible application. However, genetic algorithms are suitable for very complex systems which are on the other hand fairly problematic for expert system. Network structures of neural networks could be another application area, but the execution will be always very slow since both GAs and ANNs require a lot of computation. Development of linguistic equation systems from data is also a possible application area of genetic algorithms [15]. In the development environment, GAs are used in tuning of membership definitions.

3. Modelling with linguistic equations

Linguistic equation (LE) approach originates from fuzzy set systems: rule sets are replaced with equations, and effects of membership functions are handled with scaling. For non-linear models, the scaling technique must be non-linear as the model equations are linear. The scaling functions are called membership definitions as they have a close connection to the membership functions used in fuzzy set systems.

3.1. Membership definitions

Membership definitions provide non-linear mappings from the operation area of the (sub)system, defined with feasible ranges, to the linguistic values represented as a real-valued interval [−2, 2]. Membership functions can be generated from membership definitions.

3.1.1. Feasible ranges

Feasible range is defined as a membership function: the main area of operation is called the core area, and the whole variable range is the support area. For applications, a trapezoidal function providing linear transitions between 0 and 1 is sufficient (Fig. 3). The corner parameters can be defined on the basis of expert knowledge or extracted from data. The slope can be different on upper and lower part depending on the linearity or non-linearity of the system.

The support area is defined by the minimum and maximum values of the variable, i.e. in the support area

$$\min(x_j) \leqslant x_j \leqslant \max(x_j) \tag{1}$$

for each variable j, $j = 1, \ldots, m$.

The centre value, c_j, divides the support area into two parts, and the core area is defined by the centre values of the lower and the upper part, $(c_l)_j$ and $(c_h)_j$, correspondingly. This means that in the core area of variable j

$$(c_l)_j \leqslant x_j \leqslant (c_h)_j \tag{2}$$

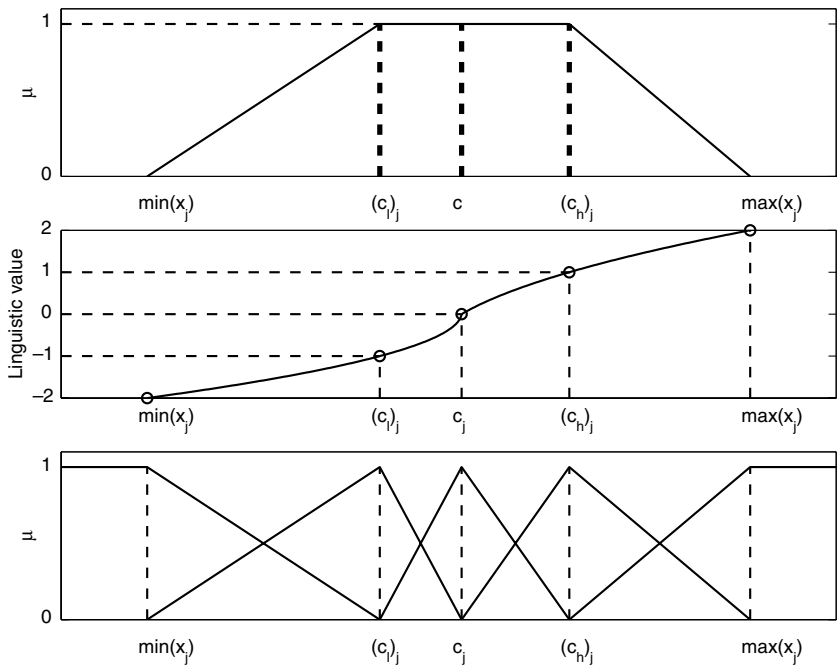

Fig. 3. Feasible range, membership definitions and membership functions.

and

$$\min(x_j) \leqslant (c_l)_j \leqslant (c_h)_j \leqslant \max(x_j). \tag{3}$$

The centre values can be defined as means or medians of the corresponding data sets. For feedback controllers, the centre values of error, change of error and change of control are all zeros. Defuzzification of the feasible range should result a value close to the centre point. If the feasible range is not highly asymmetrical, the mean or median is close to the defuzzification result of the trapezoidal feasible range.

The data-driven approach can meet two main problems. The data points do not always cover the whole area of operation, e.g. only the close neighbourhood of normal operation point may be covered, or we would like to extend the model of upper part later to the lower part. In these cases, expert knowledge is used in extending the feasible range.

On the other hand, process data contains often outliers, which must be removed before generating the feasible area, because the procedure described above is very sensitive to them. In steady-state modelling, the outlier points can be removed, but in dynamic modelling these points are needed for the analysis of interactions.

Feasible ranges should be consistent with membership definitions, and therefore they are defined together. Also the feasible ranges based on expert knowledge are verified and modified in the same way as the membership definitions. Alternatively, feasible ranges are generated from membership functions when existing rule-based fuzzy systems are converted to linguistic equations. Trapezoidal membership functions suit very well for diagnostic applications: warnings and alarms can be generated directly from the degrees of membership of the highest and lowest membership function.

3.1.2. Membership definitions

A membership definition is a (non-linear) mapping of variable values inside its range to a certain linguistic range, usually [−2, 2]. It more or less describes the distribution of variable values over its range. Membership definitions are by a function

$$x_j = f(X_j), \quad \forall \max(x_j) \leqslant x_j \leqslant \max(x_j), \quad X_j \in [-2, 2], \tag{4}$$

where x_j is the value of variable j and X_j is the corresponding linguistic value.

In present systems, the membership definitions consist of two second-order polynomials: one for negative values of $X \in [-2, 0)$ and one for positive values of $X \in [0, 2]$. So

$$x_j = f_j^-(X_j), \quad X_j \in [-2, 0) \tag{5}$$

and

$$x_j = f_j^+(X_j), \quad X_j \in [0, 2]. \tag{6}$$

Two facts must be pointed out concerning $f_j^-(X_j)$ and $f_j^+(X_j)$. They should be monotonous, increasing functions in order to result in realisable systems, i.e.

$$f_j^-((X_j)_1) > f_j^-((X_j)_2)\,\forall\,(X_j)_1 > (X_j)_2 \tag{7}$$

in the interval [−2, 0), and

$$f_j^+((X_j)_1) > f_j^+((X_j)_2)\,\forall\,(X_j)_1 > (X_j)_2 \tag{8}$$

in the interval (0, 2].

In order to keep the functions monotonous and increasing, corner points must satisfy Eq. (3). They are used in a continuous form in the linguistic equation systems. The lower part function is defined by values corresponding linguistic levels 2, −1 and 0, and the upper part function by values corresponding linguistic levels 0, 1 and 2. The upper and lower parts should overlap at the linguistic value 0.

Feasible ranges are the starting points in the construction of membership definitions. The corner parameters correspond to linguistic values −2, −1, 1, and 2. If feasible ranges are extracted from expert knowledge, the centre point is defined by defuzzifying with the centre of gravity method and it is used as the value corresponding to the linguistic value 0. For strongly asymmetrical feasible ranges, the centre value is defined on the basis of expert knowledge since the centre of gravity may be outside the core.

For automatic generation of the membership definitions, the means and medians of the data are used directly as centre values. These values will always satisfy Eq. (3).

Now we have defined five points

$$(\min(x_j), -2), ((c_l)_j, -1), (c_j, 0), ((c_h)_j, 1), (\max(x_j), 2).$$

Next, the coefficients of the second-order polynomials,

$$\begin{aligned} f_j^- &= a_j^- X_j^2 + b_j^- X_j + c_j, \quad X_j \in [-2, 0), \\ f_j^+ &= a_j^+ X_j^2 + b_j^+ X_j + c_j, \quad X_j \in [0, 2] \end{aligned} \tag{9}$$

are solved fulfilling these points. The center point, c_j, is already defined. Four linear equations are needed for solving the other coefficients:

$$\begin{aligned} 4a_j^- - 2b_j^- + c_j &= \min(x_j), \\ a_j^- - b_j^- + c_j &= (c_l)_j, \\ a_j^+ + b_j^+ + c_j &= (c_h)_j, \\ 4a_j^+ + 2b_j^+ + c_j &= \max(x_j). \end{aligned} \tag{10}$$

The same linear equations are used in tuning of the membership definitions. Fig. 4 shows examples of membership definitions.

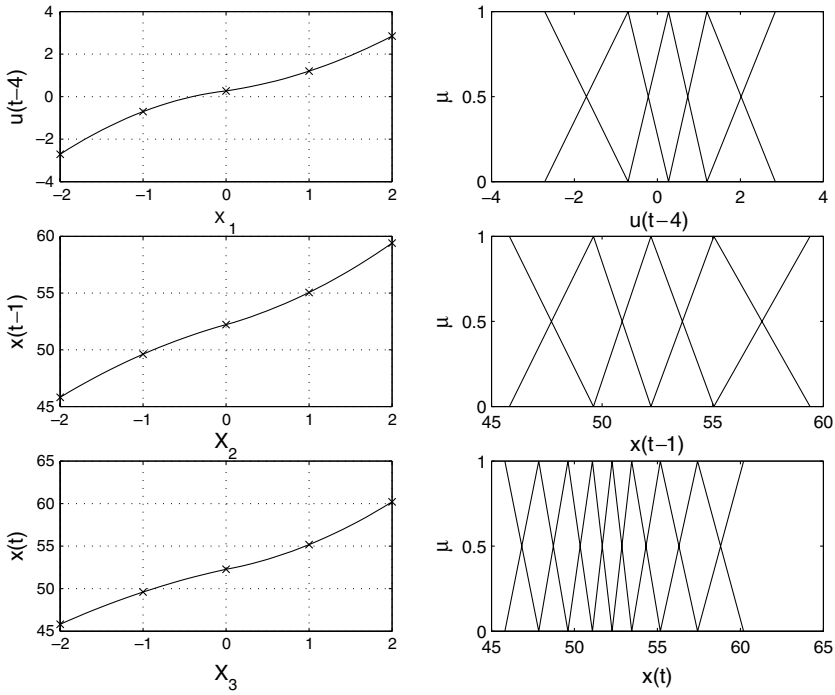

Fig. 4. Membership definitions and triangular membership functions.

The derivatives of functions f_j^- and f_j^+ should always be positive. Since Eq. (3) is not sufficient to guarantee that, the resulting derivatives are corrected to positive area by changing the coefficients of the polynomials. This correction will slightly expand the feasible range since the centre value is not changed.

In the LE models, the non-linear scaling is performed twice: first scaling from real values to the interval [−2, 2] before applying linguistic equations, and then scaling from the interval [−2, 2] to real values after applying linguistic equations.

In the case of polynomial membership definitions, the linguistic level of the input variable j is calculated according to equation

$$X_j = \begin{cases} 2 & \text{with } x_j \geqslant \max(x_j), \\ \dfrac{-b_j^+ + \sqrt{b_j^{+2} - 4a_j^+(c_j - x_j)}}{2a_j^+} & \text{with } c_j \leqslant x_j \leqslant \max(x_j), \\ \dfrac{-b_j^- + \sqrt{b_j^{-2} - 4a_j^-(c_j - x_j)}}{2a_j^-} & \text{with } \min(x_j) \leqslant x_j \leqslant c_j, \\ -2 & \text{with } x_j \leqslant \min(x_j), \end{cases} \tag{11}$$

where a_j^-, b_j^-, a_j^+ and b_j^+ are coefficients of the polynomials (Eq. (9)), c_j is real value corresponding to the linguistic value 0 and x_j is the real value. $\min(x_j)$ and $\max(x_j)$ are minimum and maximum values of the real data corresponding to the linguistic values −2 and 2.

After the linguistic level of the model output, X_{output}, is calculated with linguistic equation model, it is converted to real value of the output, x_{output}, using the following equation:

$$x_{\text{output}} = \begin{cases} a^-_{\text{output}} X^2_{\text{output}} + b^-_{\text{output}} X_{\text{output}} + c_{\text{output}} & \text{with } X_{\text{output}} < 0, \\ a^+_{\text{output}} X^2_{\text{output}} + b^+_{\text{output}} X_{\text{output}} + c_{\text{output}} & \text{with } X_{\text{output}} \geqslant 0, \end{cases} \tag{12}$$

where a^-_{output}, b^-_{output}, a^+_{output} and b^+_{output} are coefficients of the polynomials (Eq. (9)), and c_{output} is the real value corresponding to the linguistic value 0.

3.1.3. Membership functions

Membership functions can be generated from membership definitions on a chosen partition. A strict partition

$$\forall x_j \in U_j, \quad \sum_{i=1}^{n} \mu_{F_j^i}(x_j) = 1, \tag{13}$$

where $\mu_{F_j^i}$ are membership functions of fuzzy sets F_j^i, suits well for automatic generation. Fig. 3 shows the default locations. The overlap between adjacent linguistic terms expresses the progressive variation from one term to the other. The number of membership functions can be different for each variable, e.g. variables $u(t-4)$ and $x(t-1)$ have five and variable $x(t)$ nine membership functions in Fig. 4. Triangular functions are used for modelling and control, and trapezoidal functions for fault diagnosis.

3.2. Interactions

The basic element of a linguistic equation (LE) model is a compact equation

$$\sum_{j=1}^{m} A_{ij} X_j + B_i = 0, \tag{14}$$

where X_j is a linguistic level for the variable j, $j = 1, \ldots, m$. In the original system [18], the linguistic values very low, low, normal, high, and very high were replaced by integer numbers −2, −1, 0, 1 and 2. The direction of the interaction is represented by interaction coefficients A_{ij}. The bias term B_i was introduced for fault diagnosis systems. A LE model with several equations is represented as a matrix equation

$$AX + B = 0, \tag{15}$$

where the interaction matrix A contains all coefficients A_{ij} and the bias vector B all bias terms B_i.

In small systems, the directions are usually quite clear: only the absolute values of the coefficients need to be defined. For more complex systems, a set of alternative equations is developed first, and the final set of equations is selected on the basis of error measures and process knowledge. Variable selection becomes very important especially when normal process data is used. All possible combinations were taken into account and the strongest interactions are selected. Each interaction will include at least one new variable to the model.

For large systems, the number of possible variable combinations becomes very large, e.g. the case models of the web break indicator system includes 24 variables, which means that there are 2024 alternative three variable interactions. Most of these alternatives are useless, and therefore, several methodologies are used for selecting alternative groups for processing. Variable selection is started with manual methods, and the final variable selection is based on generated alternative interactions assessed with domain expertise. Some variable combinations should be avoided, and this is done by defining non-groups. Real-valued linguistic equations provide a basis for sophisticated non-linear systems. The real-valued equation systems have been used because of easier tuning.

3.3. Development and tuning

Linguistic equation models consist of two parts: interactions are handled with linear equations, and non-linearities are taken into account by membership definitions. Modelling with linguistic equations has following stages (Fig. 5):

- Membership definitions are generated by using preprocessed data, means or medians, and second order polynomials.

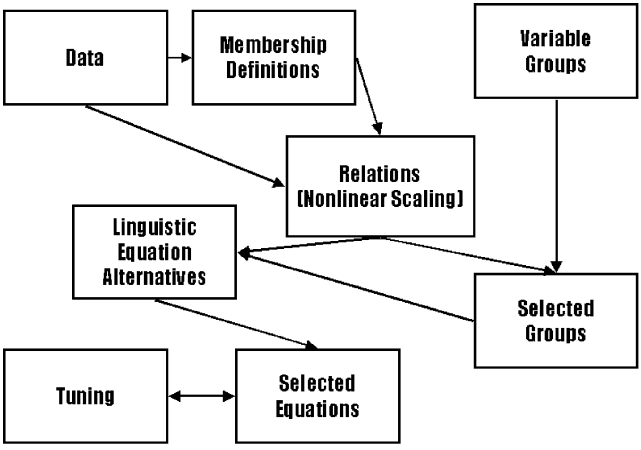

Fig. 5. Modelling with linguistic equations.

- Linguistic relations are obtained by non-linear scaling with the membership definitions.
- Linguistic equations are generated from the scaled data (linguistic relations) with linear regression.
- Selecting equations from alternatives is based either on the overall fit or on the prediction performance.
- Tuning modifies membership definitions, linguistic equations or both to improve fitting to the training data.

The tuning algorithm reduces the error between model and training data [18]. Membership definitions are tuned for one variable of the equation at a time with a linear neural network using the set of equation defined by (10). For the matrix equation (15), only one variable from each equation can be selected for tuning, i.e. membership definitions for other variables are either fixed or tuned with other equations. The recently developed genetic tuning method can handle several variables at a time by variating parameters of membership definitions.

Linguistic equations provide a very compact implementation method for systems originating from different sources. Structural restrictions are beneficial in tuning and adaptation, and the limitation can be coped with fuzzy extensions if necessary. Since only five parameters are needed for each variable, the LE systems can be adapted to various operating conditions. Recursive implementation makes also on-line adaptation possible.

An important benefit of the linear approach is that the models can be inverted, technically to any direction. Locations are selected for all input variables X_j, and the output X_{output} calculated from equation (14). In fault diagnosis, decisions are based on the fitting result as there are no output variables. Clustering is used for case-based modelling. Also variable groups are needed for these multivariable cases. In principle the equations could also be non-linear, but only linear equations have been needed in the applications so far.

The FuzzEqu Toolbox provides tools for experimenting with different methods and windows [19]. Various statistical and clustering methods are used interactively in the preprocessing stage. Membership definitions and feasible ranges can be analysed from data or defined manually. After the matrix calculations, the outputs are mapped from linguistic level to the real scale. Linguistic equation models are built interactively (Fig. 5).

3.4. Dynamic modelling and simulation

Dynamic fuzzy models can be constructed on the basis of state–space models, input–output models or semi-mechanistic models [1]. In the state–space models, fuzzy antecedent propositions are combined with a deterministic mathematical presentation of the consequent. The most common structure for

the input–output models is the NARX/Nonlinear AutoRegressive with eXogenous input model

$$y(k+1) = F(y(k), \ldots, y(k-n+1), u(k), \ldots, u(k-m+1)), \tag{16}$$

where k denotes discrete time samples, n and m are integers related to the systems' order. This structure is directly used for multiple input, single output (MISO) systems. Multiple input, multiple output (MIMO) systems can be built as a set of coupled MISO models.

Delays can be taken into account by moving the values of input variables correspondingly. The basic form of the linguistic equation (LE) model is a static mapping in the same way as fuzzy set systems and neural networks, and therefore dynamic models will include several inputs and outputs originating from a single variable. External dynamic models provide the dynamic behaviour:

- Rather simple input–output LE models, where the old value of the simulated variable and the current value of the control variable as inputs and the new value of the simulated variable as an output, can be used since non-linearities are taken into account by membership definitions.
- Linear state–space models of different operating conditions are transformed into a linguistic equation model. Since the LE model can handle non-linearities, at least some of the rules can be combined.
- In semi-mechanistic models, approximation rules are combined into linguistic equations. The approximation rules can be based on qualitative knowledge, and the mechanistic models take care of the basic level dynamic simulation.

Need for higher order models can be tested by applying classical identification to the data after scaling with membership definitions. As the parametric models, e.g. autoregressive moving average (ARMAX), autoregressive with exogeneous inputs (ARX), Box–Jenkins and Output-Error (OE), are based on linear techniques, non-linear scaling reduces the number of input and output signals needed for modelling. Normally, this analysis confirms the applicability of the above mentioned simple input–output LE model.

3.5. Case-based modelling

An effective approach to the modelling of complex non-linear systems is to partition the system into subsets and approximate each subset by a special linguistic equation model. Case-based modelling requires hybrid techniques since various clustering techniques, e.g. fuzzy-c means (FCM), subtractive, SOM, are needed for finding suitable modelling areas.

Feasible ranges define areas for submodels in the linguistic equation approach. Overlapping does not need to be based on the strict partition (Eq.

(13)). Multimodels can also be based on an extension of the Takagi–Sugeno (TS) type models.

In small dynamic models, a single equation includes all the interactions, i.e. also variables affecting to the working point of the model are included to the model. For larger models, the equation system is a set of equations where each equation describes an interaction between two to four variables. The development work starts with an automatic generation of membership definitions, which are then used in generation of interaction alternatives. Initial estimates of the delays are developed by correlation analysis. The delays should be assessed against process knowledge, especially if normal on-line process data is used [17]. The delays can also be tuned with linguistic equation models. An appropriate handling of delays will extend the operating area of the model considerably.

4. Linguistic equations: a tool for developing hybrid systems

Integration of various intelligent techniques is needed for development and tuning of smart adaptive systems for practical applications.

4.1. Hybrid systems

Various methodologies of computational intelligence have their strong areas but also weaknesses (Fig. 6). Expert systems and fuzzy set systems are knowledge oriented, and neural networks data oriented. Genetic algorithms are suitable for optimisation of heterogeneous systems. Small fuzzy systems are easy to build but their tuning is difficult in complicated systems. Fuzzy clustering and rule generation methods extend fuzzy methods towards data-based techniques. Domain expertise is used in selecting the structures for neural networks. In neuro-fuzzy methods, fuzzy systems are developed with neural networks.

Linguistic equations provide a technique for combining expertise and data to make the overall production control easier [18]. The technique has been developed in process control and modelling, where it extends the possibilities of fuzzy set systems. Linguistic equation approach combines various intelligent modelling techniques on a unified framework: a close connection to fuzzy set systems was important already in the early applications; data-driven modelling properties have brought the approach close ANN techniques. Fuzzy modelling and control was earlier the main application area. Neural networks and genetic algorithms are integrated to the tuning algorithms.

Data-driven fuzzy modelling are based on various methodologies, e.g. fuzzy clustering, self-organizing maps, neurofuzzy methods and linguistic equations. Different approaches are combined in the tuning phase: the linguistic equation

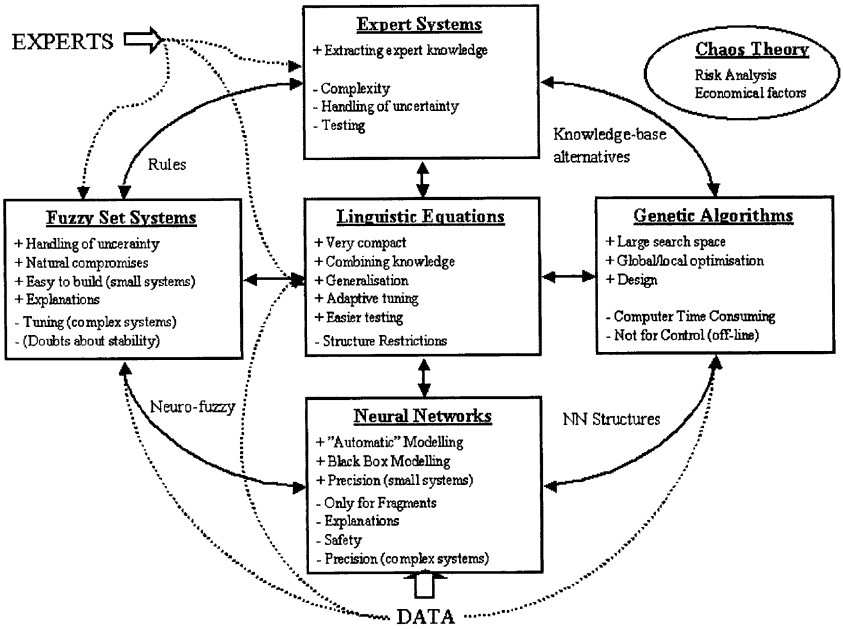

Fig. 6. Methodologies of computational intelligence [14].

approach is designed for combining different sources of information. Intelligent modelling from data and expertise can be done with FuzzEqu Toolbox created in Matlab environment [19].

4.2. Linguistic equations and fuzzy set systems

Fuzzy set systems can be changed into linguistic equation models by replacing linguistic labels with real numbers, denoted as membership locations. Usually equidistant locations are used as a starting point, e.g. negative large, negative small, zero, positive small, and positive large with numbers −2, −1, 0, 1 and 2 (Fig. 3). The resulting relations are used in the same way as scaled data in the data-driven approach shown in Fig. 5. Slight adjusting of the membership locations is usually needed, and membership definitions are generated from adjusted locations and the center points of the corresponding membership functions (Fig. 7). The model surface is directly used for very small fuzzy set systems as there are too few relations for the parameter-based approach.

Fuzzy set systems can be generated from linguistic equation systems if a sufficient number of variables are known or variated by selecting membership locations (Fig. 7). Some restrictions for the variable selection are defined by the set of equations, i.e. only one variable per equation can be solved. Fuzzy rules

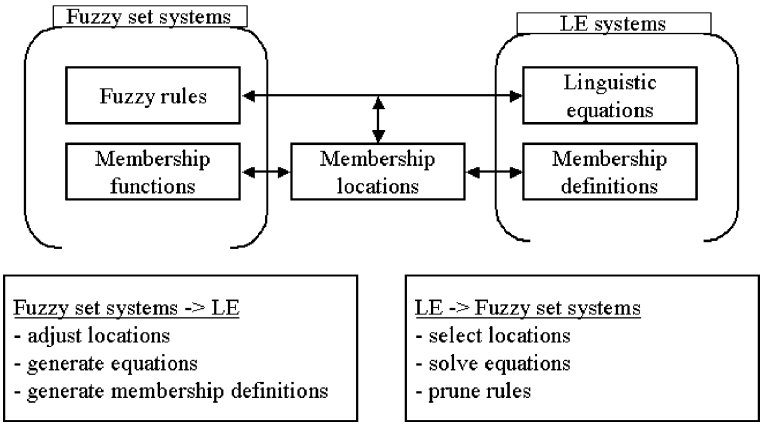

Fig. 7. Fuzzy set systems and linguistic equations.

can be generated either simultaneously or sequentially: the simultaneous approach results a compact rule set, and the sequential approach a set which is easier to interpret and prune.

Every equation can define the location of each membership function for one selected variable. Singleton models represent the LE model quite accurately if the locations of the membership functions are based on the shapes of the membership definitions in such a way that the linear surfaces are on appropriate areas. Takagi–Sugeno (TS) fuzzy models have the same requirement, but the locations of the membership functions are different. Linguistic fuzzy models are developed from singleton models. Fuzzy relational models are useful for development of fuzzy LE models. There will be fairly few non-zero elements since non-linearities are included to the membership functions.

Other fuzzy modelling approaches can be used as channels for combining different sources of information. The FuzzEqu Toolbox [19] includes routines for building a single LE system from large fuzzy systems including various ruleblocks implemented in FuzzyCon or Matlab FuzzyLogic Toolbox.

4.3. Fuzzy linguistic equations

Fuzziness is taken into account by membership definitions—linguistic equations approach does not necessarily need any uncertainty or fuzziness. However, also the linguistic equations can be used in fuzzy form, e.g.

$$\sum_{j=1}^{m} A_{ij}X_j + B_i = \begin{cases} -1 & \text{with } \mu = \mu_{-1}, \\ 0 & \text{with } \mu = \mu_0, \\ 1 & \text{with } \mu = \mu_1, \end{cases} \tag{17}$$

i.e. all the variables are not included to the model. The fact that experts do not always agree with interactions can also be taken into account by using several interaction matrices with different coefficient values. On the other hand, the directions of interactions can depend on the working area in non-linear systems. In these cases, different interaction matrices have different degrees of membership. Different combined effects can be taken into account as well [18].

Fuzzy linguistic equations are essential in present applications of linguistic equations in fault diagnosis and dynamic modelling. Fuzzy methods take care of the smooth transitions between the cases, and the degree of membership evaluated from the fuzziness of the equations,

$$S_j = \sum_{j=1}^{m} A_{ij} X_j + B_i \tag{18}$$

is the basis of the calculation of quality and risk variables. All the variables, both input and output, are included. The interaction coefficients and bias terms are normalised for the comparison of equations. The fuzziness S_j can be used for clustering the data on the basis of the interaction directions. For larger models, the equation system is a set of equations where each equation describes an interaction between two to four variables. A multimodel approach based on fuzzy LE models has been developed for combining specialised submodels [17]. The approach is aimed for systems that cannot be sufficiently described with a single set of membership definitions because of very strong non-linearities. Additional properties can be achieved since also equations and delays can be different in different submodels. In the multimodel approach, the working area defined by a separate working point model. The submodels are developed by the case-based modelling approach.

4.4. Multimodel approach

Compact LE models provide a good basis for multimodel systems. For large data sets, the material can be compressed or decomposed by clustering. In the FuzzEqu Toolbox, fuzzy clustering (FCM) and self-organising map (SOM) are the main methods: fuzzy c-means clustering is done with the functions of the Fuzzy Logic Toolbox and clustering with SOM is based on functions available in the Neural Network Toolbox. FCM is usually used when the number of clusters is known, but a suitable number of clusters can be estimated by comparing partition coefficients and entropies. Subtractive clustering is normally used when the number of clusters is unknown, but number of clusters can be iterated by changing clustering parameters.

In the compressing case, the cluster centers or locations will replace the corresponding columns in the data set, and the compressed data is used for finding working point models. In the clustering case, the modelling is done

separately for each cluster, and the data set is replaced by the data associated to the individual cluster.

Shape of clusters is important in multimodel approach. All the above described methods lead to more or less symmetric clusters for normalised or unnormalised data. Linguistic fuzzy clustering or linguistic neural networks based on non-linear scaling detect clusters of different geometrical shapes and thus reduce considerably the number of submodels needed. This also reduces the overfitting risk.

The LE provides a compact modeling of more or less smooth input–output dependencies. The Fuzzy-ROSA method (FRM) serves for a data-based rule generation to model a given input–output dependency and is efficient for modeling complicated local non-linear structures. These properties are combined in a hybrid data-based modeling concept which is applied to dynamic simulation of a solar power plant (Fig. 8). The performance of the simulator is considerably enhanced with this concept, and the hybrid simulator can be used in control design. The hybrid approach was tested in data-based modeling of dynamic behavior of a solar plant [22].

Temporal reasoning on process history is based on shapes analysed from the data. In the LE-based trend analysis, features of dynamic behaviour are handled by scaling with membership definitions; fuzzy reasoning methods are transformed into linguistic equations. This approach is integrated to the new LECont control concept implemented in G2.

4.5. Linguistic neural networks

Non-linear multivariable models can be developed by using SOM as a clustering method, and generating LE models from these neurons [23]. The resulting LE model, which can be used to any direction, represents the SOM network very accurately. Most cases in the time series are also well represented.

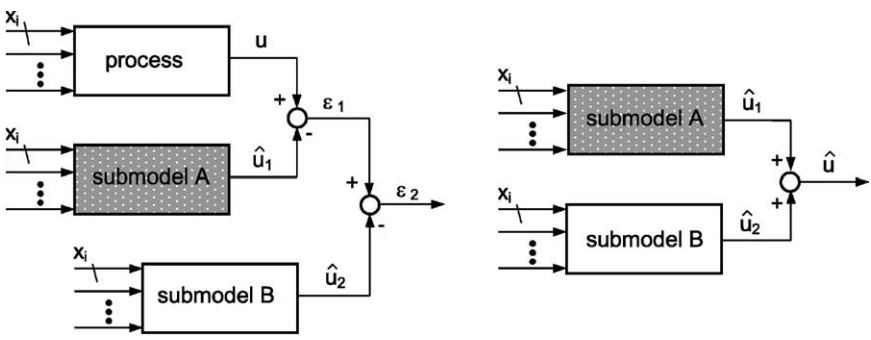

Fig. 8. Cascaded modelling (left) and resulting model (right) [22].

The differences in the interactions, which can be seen as an increased fuzziness of the model, are related to changes in operation conditions, which should be handled separately. The LE model can be considered as a new type of neural network, linguistic SOM network, where each neuron weight has also a linguistic meaning. This network can be adapted to changing conditions, e.g. different regions and industry, by adjusting membership definitions. This approach provides tools for domain experts to adapt Kohonen networks to different operating conditions without reconstructing the network.

4.6. Integration

Several sets of linguistic equations can be combined by the matrix presentation, where each row is generated separately. Some of the equations may be generated automatically from data, some from expertise presented by fuzzy set systems, and some are defined directly from prior knowledge. Each linguistic equation represents a multivariable interaction: the directions and strengths of interactions are defined by coefficients of the interaction matrix. Only the variables with non-zero coefficient belong to the interaction.

Fuzzy models on any fuzzy partition can be generated from LE models: rules or relations are developed either sequentially or simultaneously [18], and membership functions are generated from the membership definitions on any location. For integrating different methodologies of fuzzy set system development, the FuzzEqu Toolbox has links with FuzzyCon, Matlab Fuzzy Logic Toolbox, DataEngine, WinROSA and Dora for Windows.

Combining neural computing and non-linear scaling provides a wide novel set of models: linguistic neural networks. This extension results new application areas for old networks, e.g. linguistic perceptron networks operate also in non-linear applications.

5. Tasks and applications

Subsystems shown in Fig. 1 require different modelling methods. Steady-state modelling can be used for generating systems for diagnosis, software sensors and trend analysis (Fig. 9). Dynamic modelling is needed for prediction of trends and detecting of fluctuations. Also control design needs dynamic models.

5.1. Dynamic simulation

According to the test results at the Plataforma Solar de Almeria, the dynamic simulator of the solar collector field represents very accurately the field operation. In steady weather conditions, the present simulator operates within 2 K. Oscillatory conditions are also handled correctly. The simulator is based

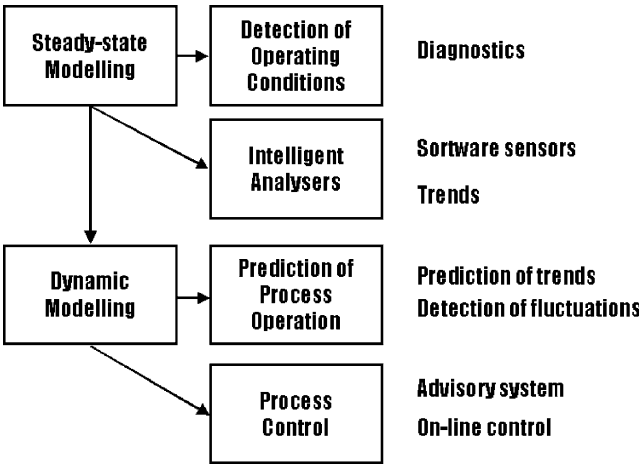

Fig. 9. Development of intelligent applications.

on the multimodel LE approach with four specialised LE models developed for different operating conditions. The LE model provides a good overall behaviour in different operating conditions (Fig. 10). The model of the solar power plant has been improved by introducing additional fuzzy models for special situations [22]. These models were developed with Fuzzy-ROSA method [10,28]. Dynamic LE models have been used in tuning of LE controllers for a solar power plant [22] and for a lime kiln [17].

5.2. Intelligent analysers

Most intelligent analysers are software sensors which combine several on-line measurements to predict output, detect input changes or trends. A fuzzy software sensor for a fermentation process is described in [24]. Trend analysis was applied to the same process in [25]. A linguistic equation based intelligent analysers for continuous cooking and a comparison of different fuzzy methods and linguistic equations is described in [31]. The model predicts the quality of the pulp four hours ahead (Fig. 11). LE-based analysers have been developed for fuel quality in a lime kiln [17]. Another area of applications is in detection of operating conditions, e.g. web break sensitivity indicators [11] belong to this area. In solar applications, the operating conditions are defined with irradiation and temperature difference between inlet and outlet [20,21].

5.3. Intelligent control

Fuzzy control applications can be converted to linguistic equation form by replacing linguistic levels for error, error derivative and change of control by

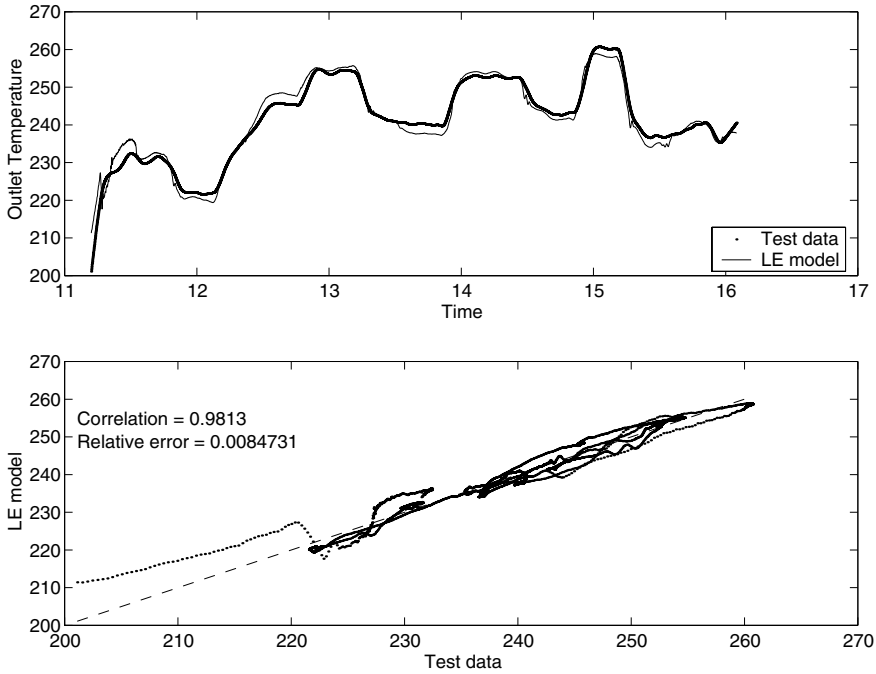

Fig. 10. Simulation results of the solar collector field [22].

real values. The first direct LE controller was implemented in 1996 for a solar power plant [20,21], and later the multilevel LE controller was installed for an industrial lime kiln [9].

A PI type fuzzy controller is represented in the following form

$$\Delta u = e + \Delta e, \tag{19}$$

which is a special case of the matrix equation $AX = 0$ with the interaction matrix $A = [\,1 \quad 1 \quad -1\,]$, and variables $X = [\,e \quad \Delta e \quad \Delta u\,]^T$. Each cell in Fig. 12 is a rule represented with integer numbers as the linguistic values *NB*, *NS*, *ZO*, *PS*, *PB* are replaced by integer numbers −2, −1, 0, 1 and 2. All these rules can be obtained from Eq. (12) if −2 and 2 are the minimum and maximum values respectively. Eq. (19) is also applicable on any fuzzy partition, and the procedure produces always a rule set which is complete, consistent, and continuous. If a non-complete set is satisfactory, a part of the rules can be deleted already before tuning. The best similarity with the LE controller is achieved if the positions of all the fuzzy sets are coordinated with the real valued linguistic equations, i.e. linguistic values are replaced by real numbers.

Similarly, *a fuzzy PD type controller* can be represented by a single linguistic equation

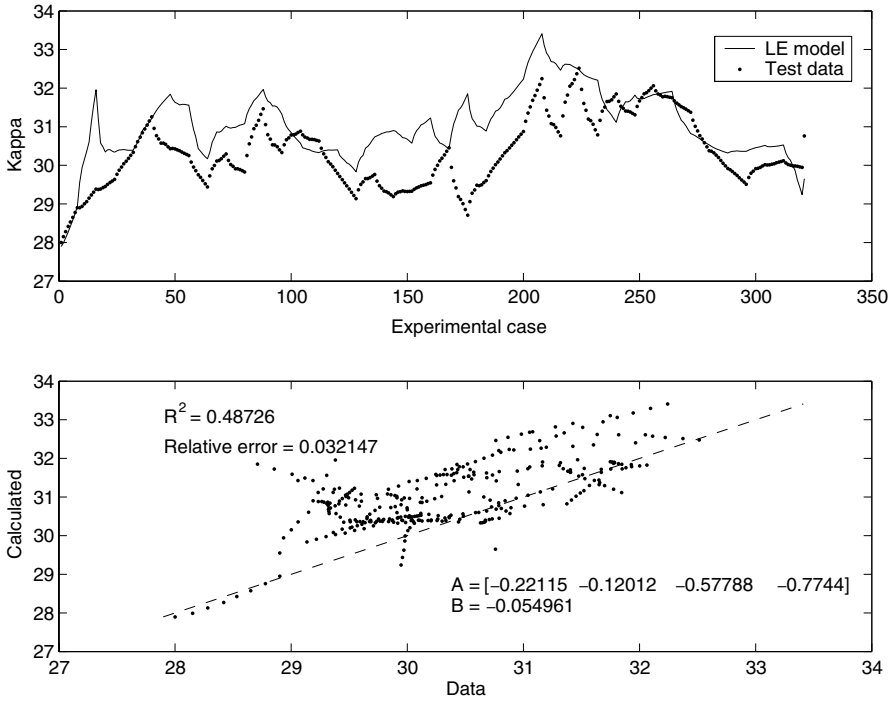

Fig. 11. Prediction results in continuous cooking.

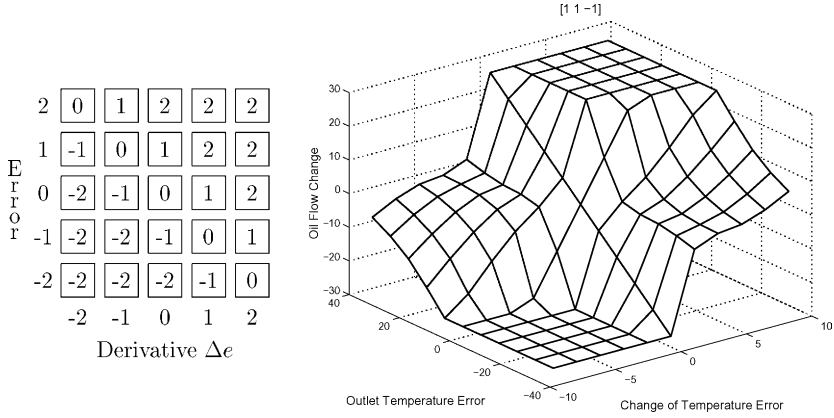

Fig. 12. The rule base of a fuzzy PI controller represented by integer number and the control surface of the corresponding LE controller.

$$u = e + \Delta e, \tag{20}$$

which is a special case of the matrix equation $AX = 0$ with the interaction matrix $A = [1 \quad 1 \quad -1]$, and variables $X = [e \quad \Delta e \quad u]^T$.

The operation of a multilevel LE controller corresponds to a three level cascade controller:

- Basic PI type LE controller handles the normal operation with symmetrical membership definitions;
- Operation condition controller changes the control surface of the basic LE controller by modifying membership definitions for the change in the control variable u;
- Predictive LE controller changes membership definition for the derivative of the error e. This level contains both the braking and asymmetrical action.

In the solar application, several additional features have been included to the controller to avoid too high temperatures:

- The inlet temperature changes considerably from the average level corresponding to case known to the controller;
- Temperature is rising so fast that the controller cannot handle efficiently the evolving situation;
- The temperature difference between the inlet and the outlet is too high compared to acceptable level corresponding to the recent average of the corrected irradiation.

Additional change of control is introduced if at least one of these cases is active. The main purpose for introducing these features is to avoid oscillatory conditions which are difficult to control. The first and second ones of these actions are predictive, and the third one is corrective. All these properties are implemented into a very compact control program. Modularity is beneficial for the tuning of the controller to various operating conditions, and most important is that the same controller can operate on the whole working area. The present control has also a cascade level for changing the temperature set point (Fig. 13).

The basic LE controller could be represented as a fuzzy controller, but the additional levels described above would make the fuzzy controller to complicated to implement. As the implementation is very compact, features of the LE controller can extended, e.g. feedforward and feedback controllers are used together in lime kiln control [9].

5.4. Fault diagnosis and performance monitoring

Model-based approach has been an essential part of the fault detection already a long time [6]. As the classical approach utilising analytical models has

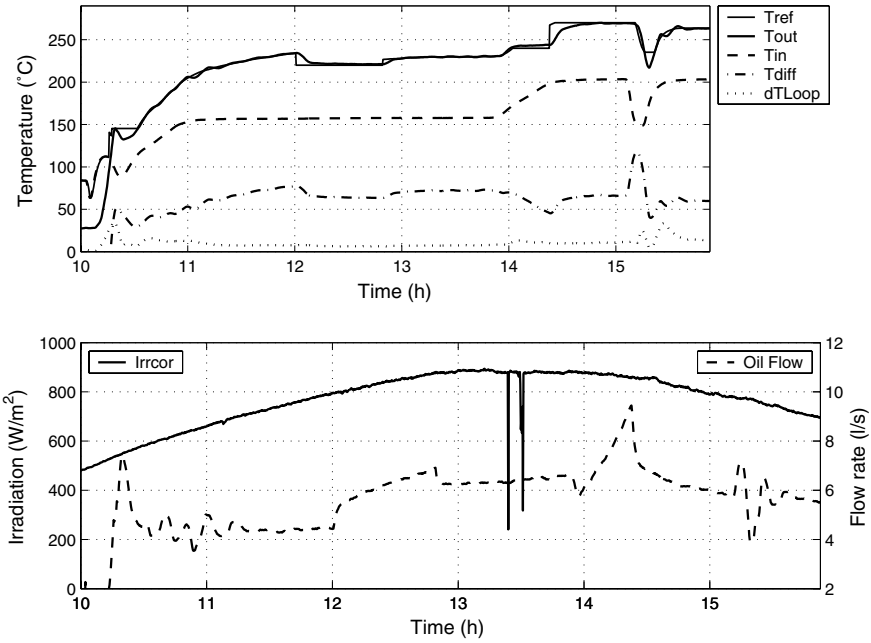

Fig. 13. Test results of the LE controller: temperatures, oil flow and irradiation.

difficulties in handling uncertainties, fuzzy logic was included [7], and the results from dynamic systems were extended to other types of fault diagnosis [13,33]. Fuzzy rule-based approach and linguistic equations were applied to functional testing in electronics manufacturing [27]. The overall model-based diagnostical process analysis (MDPA) introduced in [16] has been earlier extended to a case-based reasoning (CBR) type approach in a paper machine application [11]. Resulting systems can be used in various ways suitable for software sensors, risk analysis and detection of sensor failures. Sophisticated trend information can be utilised by temporal reasoning on the recent process history. The MDPA methodology has been tested with simulations, expert knowledge and real data.

Case-based models are generated for large-scale systems, and usually much less than one percent of the original alternatives are needed in the case models, e.g. only from 7 to 22 equations were needed in the paper machine applications. The final set of selected groups can be built in a modular way: some groups may include also four or five variables, and different subgroups can be developed for different subsets of variables. Redundant measurements are included to the same groups only for the fault diagnosis of measurement devices. All these tasks can be performed automatically, but more important is that any intermediate result can be modified on the basis of expertise.

Intelligent system developed for analysing paper web breaks consists of specialised models corresponding different operating conditions. The core of the system is a set of linguistic equation (LE) models developed by data-based techniques from process data of a paper machine. A special system was developed for interactive selection of interesting cases from extensive data sets. Various methods were compared in variable selection stage. The LE modelling was done for variable groups including upto five variables. In order to detect different operating conditions each model set should be accurate in its own case but in the same time its fit to other cases should be much worse. This methodology is suitable for detecting most important variables and variable groups for separate cases.

In the web break indicator, the fuzziness of each equation is calculated by Eq. (18). If the fuzziness is close to zero, the degree of membership for the equation is one. Higher fuzziness reduces this degree. Each equation has also a weight factor whose value depends on the sharpness of the fuzziness distribution in the training case. The degree of membership for each case is obtained by combining the membership degrees and weight factors of individual equations, and the final selection of the active cases and the corresponding web break category is based on fuzzy reasoning.

6. Conclusions

Different combinations of fuzzy logic and neural networks provide various ingredients for smart adaptive applications. Linguistic equation (LE) approach originating from fuzzy logic is an efficient technique for these problems. Insight to the process operation is maintained since all the modules can be assessed by expert knowledge and membership definitions relate measurements to appropriate operating areas. The LE approach increases the performance by combining various specialised models in a case-based approach: models can be generated automatically from data. The system integration leads to a hybrid system: fuzzy set systems move gradually to higher levels, neural networks and evolutionary computing are used for tuning, and the whole system reinforced with efficient statistical analysis, signal processing and mechanistic modelling and simulation.

References

[1] R. Babuška, M. Setnes, U. Kaymak, H.B. Verbruggen, Fuzzy Modelling: a Universal and Transparent Tool, in: Proceedings of TOOLMET'97—Tool Environments and Development Methods for Intelligent Systems, Oulu, 1997, pp. 1–27, Oulun yliopistopaino.

[2] R. Babuška, Fuzzy Modeling and Identification, Kluwer Academic Publisher, Boston, 1998.

[3] S. Chen, C.F.N. Cowan, P.M. Grant, Orthogonal least squares learning algorithm for radial basis function networks, IEEE Transactions on Neural Networks 2 (1991) 302–309.
[4] D. Dubois, H. Prade, L. Ughetto, Fuzzy logic, control engineering and artificial intelligence, in: H.B. Verbruggen, H.-J. Zimmermann, R. Babuska (Eds.), Fuzzy Algorithms for Control, International Series in Intelligent Technologies, Kluwer, Boston, 1999, pp. 17–57.
[5] F.D. Foresee, M.T. Hagan, Gauss–Newton approximation to Bayesian regularization, in: Proceedings of the 1997 International Joint Conference on Neural Networks, 1997, pp. 1930–1935.
[6] R. Isermann, Process fault detection based on modelling and estimation methods—a survey, Automatica (1984) 387–404.
[7] R. Isermann, On the integration of fault detection methods via unified fuzzy symptom representation, in: Proceedings of the First European Congress on Fuzzy and Intelligent Technologies—EUFIT'93, vol. 1, Aachen, 1993, pp. 400–407, Augustinus Buchhandlung.
[8] J.-S.R. Jang, ANFIS: adaptive-network-based fuzzy inference systems, IEEE Transactions on Systems, Man, and Cybernetics 23 (3) (1993) 665–685.
[9] M. Järvensivu, E. Juuso, O. Ahava, Intelligent control of a rotary kiln fired with producer gas generated from biomass, Engineering Applications of Artificial Intelligence 14 (5) (2001) 629–653.
[10] H. Jessen, T. Slawinski, Test and rating strategies for data-based rule generation, in: Reihe Computational Intelligence, CI–39/98, Sonderforschungsbereich 531, Universität Dortmund, 1998.
[11] E. Juuso, T. Ahola, K. Oinonen, Case-based reasoning with linguistic equations for web break sensitivity indicator, in: Preprints of TOOLMET'98 Symposium—Tool Environments and Development Methods for Intelligent Systems, Oulu, 1998, pp. 97–106, Oulu, 1998, Oulun yliopistopaino.
[12] E.K. Juuso, Linguistic equation framework for adaptive expert systems, in: Modelling and Simulation 1992, Proceedings of the 1992 European Simulation Multiconference, York, UK, 1992, pp. 99–103, San Diego, USA, 1992, SCS International.
[13] E.K. Juuso, Fault diagnosis based on linguistic equation framework, in: Preprints of IFAC Symposium on Fault Detection, Supervision and Safety for Technical Processes SAFEPROCESS'94, vol. 1, Espoo, 1994, pp. 374–379, Helsinki, Hakapaino.
[14] E.K. Juuso, Computational intelligence in distributed interactive synthetic environments, in: Simulation in Industry, Proceedings of the Eighth European Simulation Symposium, Simulation in Industry, ESS'96 Genoa, Italy, 1996, pp. 157–162, San Diego, USA, 1996, SCS International.
[15] E.K. Juuso, Linguistic equations in system development for computational intelligence, in: Proceedings of the Fourth European Congress on Intelligent Techniques and Soft Computing—EUFIT'96, Aachen, vol. 1, 1996, pp. 1127–1131, Aachen, 1996, Mainz.
[16] E.K. Juuso, Intelligent methods in diagnostical process analysis, in: Proceedings of XIV IMEKO World Congress, New Measurements—Challenges and Visions, Tampere 1997, vol. VII, pp. 1–6, 1997.
[17] E.K. Juuso, Intelligent dynamic simulation of a lime kiln with linguistic equations, in: ESM'99: Modelling and Simulation: A tool for the Next Millenium, 13th European Simulation Multiconference, Warsaw, Poland, 1999, pp. 395–400, Delft, The Netherlands, 1998, SCS.
[18] E.K. Juuso, Fuzzy control in process industry: The linguistic equation approach, in: H.B. Verbruggen, H.-J. Zimmermann, R. Babuska (Eds.), Fuzzy Algorithms for Control, International Series in Intelligent Technologies, Kluwer, Boston, 1999, pp. 243–300.
[19] E.K. Juuso, Linguistic equations for data analysis: Fuzzequ toolbox, in: Proceedings of TOOLMET 2000 Symposium—Tool Environments and Development Methods for Intelligent Systems, Oulu, 2000, pp. 212–226, Oulun yliopistopaino.

[20] E.K. Juuso, P. Balsa, K. Leiviska, Linguistic equation controller applied to a solar collectors field, in: Proceedings of the European Control Conference-ECC'97, vol. 5, Brussels, 1–4 July 1997, TH-E I4, paper 267 (CD-ROM), p. 6.
[21] E.K. Juuso, P. Balsa, L. Valenzuela, Multilevel linguistic equation controller applied to a 1 mw solar power plant, in: Proceedings of the ACC'98, vol. 6, Philadelphia, PA, 24–26 June 1998, pp. 3891–3895.
[22] E.K. Juuso, D. Schauten, T. Slawinski, H. Kiendl, Combination of linguistic equations and the fuzzy-ROSA method in dynamic simulation of a solar collector field, in: Proceedings of TOOLMET 2000 Symposium—Tool Environments and Development Methods for Intelligent Systems, Oulu, 2000, pp. 63–77, Oulun yliopistopaino.
[23] S. Juuso, E.K. Juuso, Modelling and forecasting with Kohonen networks and linguistic equations, in: Proceedings of EUFIT'99, BA-4-5 (CD-ROM), 1999, p. 7.
[24] S. Kivikunnas, M. Bergonzini-Corradini, E.K. Juuso, Fuzzy conversion estimation in fermentation control, in: Proceedings of the TOOLMET'96 Workshop, Oulu, Finland, 1996, pp. 177–186, UOulu, Control Eng. Lab.
[25] S. Kivikunnas, K. Ibatici, E.K. Juuso, Process Trend Analysis and Fuzzy Reasoning in Fermentation Control, in: Proceedings of IWISP'96—Third International Workshop on Image and Signal Processing on the Theme of Advances in Computational Intelligence, 1996, Manchester, UK, pp. 137–140.
[26] T. Kohonen, Self-Organizing Maps, Springer, Berlin, 1995.
[27] K. Komulainen, M. Heikkinen, T. Frantti, E. Juuso, K. Leiviskä, Utilization of failure information in functional testing, in: Proceedings of TOOLMET'97—Tool Environments and Development Methods for Intelligent Systems, Oulu, 1997, pp. 109–115, Oulun yliopistopaino.
[28] A. Krone, H. Kiendl, Automatic generation of positive and negative rules for two-way fuzzy controllers, in: H.-J. Zimmermann (Ed.), Proceedings of the Second European Congress on Intelligent Technologies and Soft Computing-EUFIT'94, vol. 1, Aachen, 21–23 September 1994, pp. 438–447, Augustinus Buchhandlung.
[29] A. Krone, U. Schwane, Generating fuzzy rules from contradictory data of different control strategies and control performances, in: Proceedings of the Fuzz-IEEE'96, New Orleans, USA, 1996, pp. 492–497.
[30] J. Lampinen, Advances in neural network modeling, in: Proceedings of TOOLMET'97—Tool Environments and Development Methods for Intelligent Systems, Oulu, 1997, pp. 28–35, Oulun yliopistopaino.
[31] K. Leiviskä, E. Juuso, A. Isokangas, Intelligent modelling of continuous pulp cooking, in: K. Leiviskä (Ed.), Industrial Applications of Soft Computing, Springer, Heidelberg, 2001, pp. 147–158.
[32] S. Lotvonen, S. Kivikunnas, E. Juuso, Tuning of a fuzzy system with genetic algorithms and linguistic equations, in: Proceedings of the Fifth European Congress on Intelligent Techniques and Soft Computing —EUFIT'97, vol. 2, Aachen, 1997, pp. 1289–1293, Mainz.
[33] M. Ulieru, From Boolean probabilistic to fuzzy possibilistic fault tree processing in diagnostic decision making, in: Proceedings of the First European Congress on Fuzzy and Intelligent Technologies—EUFIT'93, vol. 1, Aachen, 1993, pp. 408–414, Augustinus Buchhandlung.
[34] W. Pedrycz, An identification algorithm in fuzzy relational systems, Fuzzy Sets and Systems 13 (1984) 153–167.
[35] D.E. Rummelhart, G.E. Hinton, R.J. Williams, Learning internal representations by error propagation, in: D.E. Rummelhart, J. McClelland (Eds.), Parallel Data Processing, M.I.T. Press, Cambridge, MA, 1986, pp. 318–362.
[36] H. Szczerbicka, Are genetic algorithms a panacea for optimization of simulation models, in: Proceedings of the Conference on Modelling and Simulation 1994, San Diego, USA, pp. 38–46, SCS International.

[37] T. Takagi, M. Sugeno, Fuzzy identification of systems and its applications to modelling and control, IEEE Transactions on Systems, Man, and Cybernetics 15 (1) (1985) 116–132.
[38] L.-X. Wang, Adaptive Fuzzy Systems and Control. Design and Stability Analysis, Prentice Hall, New Jersey, 1994.
[39] L.-X. Wang, J.M. Mendel, Generating fuzzy rules by learning from examples, IEEE Transactions on Systems, Man, and Cybernetics 22 (1992) 1414–1427.
[40] L.A. Zadeh, Knowledge representation in fuzzy logic, IEEE Transactions on Knowledge and Data Engineering 1 (1) (1989) 89–100.