Complex Systems Modelling by Rule Based Networks

ALEXANDER GEGOV
School of Computing
University of Portsmouth
Buckingham Building, Portsmouth PO1 3HE
UNITED KINGDOM
alexander.gegov@port.ac.uk    http://userweb.port.ac.uk/~gegova/

Abstract: - A rule based system is a ‘black box’ model in the form of a single large rule base which does not take into account any interconnections in a complex process. As opposed to this, a rule based network is a ‘white box’ model in the form of multiple smaller rule bases which takes into account all interconnections. This paper introduces briefly the foundations of rule based networks including formal models, basic operations, structural properties and advanced operations. It also highlights feedforward and feedback rule based networks. Finally, a comparative evaluation of rule based networks in relation to rule based systems is presented.

Keywords: Complex processes, rule based models, rule based systems, rule based networks.

1 Introduction
Complex processes are usually described by a number of features such as uncertainty, non-linearity, interactions and dimensionality. These features often represent a serious challenge to the modelling of such processes.

In this context, rule based systems such as expert systems have already proved themseleves as a powerful tool for dealing with complex processes characterised by some type of uncertainty [2], [4], [5]. The most common cause for uncertainty is knowledge about the process that is in some way incomplete, ambiguous or contradictory [1], [6].

Besides this, the use of rule based systems helps with the tackling of non-linearity in complex processes [7], [8], [9]. This is because a rule base is usually capable of representing quite well strongly non-linear input-output relationships in a complex process.

However, in spite of the relative success of rule based systems in capturing uncertainty and non-linearity, there are other features of complexity that may be problematic [3]. For example, the interactions among subprocesses and the high dimensionality in terms of large number of inputs may lead to the deterioration of rule based models. In particular, these models may suffer from inadequate accuracy, low efficiency and poor transparency. These potential drawbacks are typical for rule based models of complex processes that are derived either from expert knowledge or data.

The next section discusses the main types of rule based systems in relation to their ability to capture different features of complexity.

2 Types of Rule Based Systems
The most common type of rule based system consists of a single rule base (SRB) whereby the associated rule based model is a ‘black box’. In this case, a SRB system deals with all process inputs at the same time while not taking into account interactions and dimensionality. A SRB system is usually known as a standard rule based system.

Another type of rule based system consists of multiple rule bases (MRB), whereby the associated rule based model is a ‘white box’. In this case, the MRB system deals with process inputs sequentially in time while taking into account interactions and dimensionality. A MRB system is usually known as a chained rule based system.

A third type rule based system consists of networked rule bases (NRB), whereby the associated rule based model is also a ‘white box’. In this case, a NRB system deals with process inputs sequentially in both time and space while taking into account interactions and dimensionality. A NRB system represents a rule based network. This network is a collection of interconnected SRB systems that are considered as self-standing individual units.
Overall, a NRB system represents a novel extension to both SRB and MRB systems. In particular, a NRB system plays the role of a bridge between SRB and MRB systems that facilitates their use.

In this context, NRB systems provide a basis for extending rule based systems to rule based networks with the purpose of capturing better different features of complexity. The next section introduces three groups of formal models for rule based networks.

3 Formal Models of Rule Based Networks

The first group of models makes use of block schemes and topological expressions.

A block scheme is a visual model that uses blocks and arrows for representing individual rule bases and the interactions among them, respectively. For example, the simplest rule based network has a single node N with no connections, an input x and an output y. This network is equivalent to a rule based system and can be presented by the block scheme in Fig.1.

![Fig.1. Block scheme for a single node network](image)

A topological expression is an analytical equivalent of a block scheme whereby the location of individual rule bases and the interactions among them are specified uniquely in the expression. For example, the topological expression for the rule based network in Fig.1 is given by Equation (1).

\[ [N] (x \mid y) \]  

The second group of models makes use of Boolean matrices and binary relations.

A Boolean matrix is a compressed image of a rule base, whereby the row and column labels are all possible permutations of values of inputs and outputs, and each rule is coded by a 1. For example, if the rule base for the node N from Fig.1 is an identity mapping of the input x to the output y whereby both x and y take three values, then the corresponding Boolean matrix for N is the matrix given in Table 1.

<table>
<thead>
<tr>
<th>x / y</th>
<th>Value 1</th>
<th>Value 2</th>
<th>Value 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value 1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Value 2</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Value 3</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

A binary relation is another way of compressing further the information contained in a Boolean matrix whereby the pairs in the relation are all possible permutations of values for inputs and outputs from the row and column labels of the matrix, and each rule is coded by a pair. For example, the binary relation corresponding to the Boolean matrix in Table 1 is given by Equation (2).

\[ \{(1, 1), (2, 2), (3, 3)\} \]  

The third group of models makes use of incidence and adjacency matrices. In this case, it is necessary to introduce a start node that applies the inputs and an end node that receives the outputs.

An incidence matrix describes the interactions among individual rule bases, whereby the row labels are the rule bases, the column labels are the associated interactions, and the matrix entries are (+1,-1) pairs for existing connections. For example, the incidence matrix for the rule based network in Fig.1 is the matrix given in Table 2.

<table>
<thead>
<tr>
<th>Node / Connection</th>
<th>x</th>
<th>y</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start node</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>N</td>
<td>-1</td>
<td>1</td>
</tr>
<tr>
<td>End node</td>
<td>0</td>
<td>-1</td>
</tr>
</tbody>
</table>

An adjacency matrix is another way of presenting the information contained in an incidence matrix, whereby both the row and column labels are the rule bases, and the matrix entries are 1s for existing connections. For example, the adjacency matrix corresponding to the incidence matrix in Table 2 is the matrix given in Table 3.

<table>
<thead>
<tr>
<th>Node / Node</th>
<th>Start</th>
<th>N</th>
<th>End</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>N</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>End</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
In summary, the models from the first group describe the structure of a rule based network at a high level of abstraction. In this context, both block schemes and topological expressions specify the location of network nodes represented by rule bases and the direction of network connections represented by the interactions among rule bases. On the other hand, the models in the second and the third group describe a rule based network at a lower level of abstraction. In particular, Boolean matrices and binary relations specify the structure of network nodes, whereas incidence and adjacency matrices specify the structure of network connections.

In order to manipulate individual rule bases in a rule based network, it is necessary to use some basic operations. In this context, the next section introduces three groups of basic operations.

4 Basic Operations in Rule Based Networks

The first group of basic operations deals with network nodes in sequence. These operations include horizontal merging and splitting.

In the case of horizontal merging, two operand nodes \( N_{11} \) and \( N_{12} \) are merged into a single product node \( N_{11 \times 12} \) provided that the output vector for the first node \( z_{11,12} \) is fed forward as an input vector for the second node in the form of an intermediate vector. The product node has the same input vector \( x_{11} \) as the input vector for the first operand node, the same output vector \( y_{12} \) as the output vector for the second operand node, whereas the intermediate vector disappears in the operand node.

The operation of horizontal merging is illustrated by the topological expression in Equation (3). In this case, the symbol ‘*’ denotes the operation, whereas the subscripts designate the location of nodes and the associated vectors.

\[
[N_{11}] (x_{11} | z_{11,12}) \ast [N_{12}] (z_{11,12} | y_{12}) = [N_{11 \times 12}] (x_{11} | y_{12})
\]  

Horizontal splitting is the inverse operation of horizontal merging, whereby a single operand node \( N_{11 \times 12} \) is split into two separate product nodes \( N_{11} \) and \( N_{12} \). The first product node has the same input vector \( x_{11} \) as the input vector for the operand node, the second product node has the same output vector as the output vector for the operand node, and an intermediate vector \( z_{11,12} \) appears between the two operand nodes.

The operation of horizontal splitting is illustrated by the topological expression in Equation (4). In this case, the symbol ‘/’ denotes the operation, whereas the subscripts designate the location of nodes and the associated vectors.

\[
[N_{11}/12] (x_{11} \mid y_{12}) = [N_{11}] (x_{11} \mid z_{11,12}) / [N_{12}] (z_{11,12} \mid y_{12})
\]

The second group of basic operations deals with network nodes in parallel. These operations include vertical merging and splitting.

In the case of vertical merging, two separate operand nodes \( N_{11} \) and \( N_{21} \) are merged into a single product node \( N_{11+21} \). The input vector \( x_{11}, x_{21} \) for the product node is the union of input vectors \( x_{11} \) and \( x_{21} \) for the operand nodes, whereas the output vector \( y_{11}, y_{21} \) for the product node is the union of the output vectors \( y_{11} \) and \( y_{21} \) for the operand nodes.

The operation of vertical merging is illustrated by the topological expression in Equation (5). In this case, the symbol ‘+’ denotes the operation, whereas the subscripts designate the location of nodes and the associated vectors.

\[
[N_{11}] (x_{11} \mid y_{11}) + [N_{21}] (x_{21} \mid y_{21}) = [N_{11+21}] (x_{11}, x_{21} \mid y_{11}, y_{21})
\]

Vertical splitting is the inverse operation of vertical merging, whereby a single operand node \( N_{11-21} \) is split into two separate product nodes \( N_{11} \) and \( N_{21} \). The input vectors \( x_{11} \) and \( x_{21} \) for the product nodes partition the input vector \( x_{11}, x_{21} \) for the operand node, whereas the output vectors \( y_{11} \) and \( y_{21} \) for the product nodes partition the output vector \( y_{11}, y_{21} \) for the operand node.

The operation of vertical splitting is illustrated by the topological expression in Equation (6). In this case, the symbol ‘-’ denotes the operation, whereas the subscripts designate the location of nodes and the associated vectors.

\[
[N_{11-21}] (x_{11}, x_{21} \mid y_{11}, y_{21}) = [N_{11}] (x_{11} \mid y_{11}) - [N_{21}] (x_{21} \mid y_{21})
\]

The third group of basic operations also deals with network nodes in parallel. These operations include output merging and splitting.

In the case of output merging, two operand nodes \( N_{11} \) and \( N_{21} \) with a common input vector \( x_{11,21} \) are merged into a single product node \( N_{11;21} \). In this case, the input vector for the product node is the
same as the common input vector for the operand nodes, whereas the output vector \( y_{1i}, y_{2i} \) for the product node is the union of the output vectors \( y_{1i} \) and \( y_{2i} \) for the operand nodes.

The operation of output merging is illustrated by the topological expression in Equation (7). In this case, the symbol ‘;’ denotes the operation, whereas the subscripts designate the location of nodes and the associated vectors.

\[
[N_{11}] (x_{11,21} | y_{11}) ; [N_{21}] (x_{11,21} | y_{21}) = \ [N_{11;21}] (x_{11,21} | y_{11}, y_{21})
\]  

Output splitting is the inverse operation of output merging, whereby a single operand node \( N_{11;21} \) is split into two product nodes \( N_{11} \) and \( N_{21} \). The product nodes have a common input vector \( x_{11,21} \) that is the same as the input vector to the operand node, whereas the output vectors \( y_{11} \) and \( y_{21} \) for product nodes partition the output vector \( y_{11}, y_{21} \) for the operand node.

The operation of output splitting is illustrated by the topological expression in Equation (8). In this case, the symbol ‘;’ denotes the operation, whereas the subscripts designate the location of nodes and the associated vectors.

\[
[N_{11;21}] (x_{11,21} | y_{11}, y_{21}) = \ [N_{11}] (x_{11,21} | y_{11}) ; [N_{21}] (x_{11,21} | y_{21})
\]  

In summary, the operations from the first group deal with nodes located in the same level of the network structure, whereas the operations from the second and the third group deal with nodes located in the same layer of this structure. In this context, levels and layers represent rows and columns, respectively, in the underlying grid structure of the rule based network.

The presented basic operations have some structural properties, which facilitate the manipulation of individual rule bases in a rule based network. The next section introduces three groups of structural properties.

## 5 Properties of Basic Operations

The first group of structural properties relates to the basic operations of horizontal merging and splitting of network nodes.

In particular, horizontal merging is associative in that it can be applied in an arbitrary order for more than two operand nodes. For example, in the case of three operand nodes \( A, B, C \) that are merged into a product node \( D \), it does not make any difference as to whether the product from merging \( A \) and \( B \) is merged with \( C \) or \( A \) is merged with the product from merging \( B \) and \( C \). This property is illustrated by Equation (9).

\[
A * B * C = (A * B) * C = A * (B * C) = D
\]  

Similarly, horizontal splitting is associative in that an operand node \( D \) can either be split into three product nodes \( A, B, C \) by deriving \( A \) before \( B \) and \( C \) or by deriving \( C \) before \( A \) and \( B \). This property is illustrated by Equation (10).

\[
D = A / (B / C) = (A / B) / C = A / B / C
\]  

The second group of structural properties relates to the basic operations of vertical merging and splitting of network nodes.

In particular, vertical merging is associative in that it can be applied in an arbitrary order for more than two operand nodes. For example, in the case of three operand nodes \( A, B, C \) that are merged into a product node \( D \), it does not make any difference as to whether the product from merging \( A \) and \( B \) is merged with \( C \) or \( A \) is merged with the product from merging \( B \) and \( C \). This property is illustrated by Equation (11).

\[
A + B + C = (A + B) + C = A + (B + C) = D
\]  

Similarly, vertical splitting is associative in that an operand node \( D \) can either be split into three product nodes \( A, B, C \) by deriving \( A \) before \( B \) and \( C \) or by deriving \( C \) before \( A \) and \( B \). This property is illustrated by Equation (12).

\[
D = A - (B - C) = (A - B) - C = A - B - C
\]  

The third group of structural properties relates to the basic operations of output merging and splitting of network nodes.

In particular, output merging is associative in that it can be applied in an arbitrary order for more than two operand nodes. For example, in the case of three operand nodes \( A, B, C \) that are merged into a product node \( D \), it does not make any difference as to whether the product from merging \( A \) and \( B \) is merged with \( C \) or \( A \) is merged with the product from merging \( B \) and \( C \). This property is illustrated by Equation (13).

\[
A ; B ; C = (A ; B) ; C = A ; (B ; C) = D
\]
Similarly, vertical splitting is associative in that an operand node D can either be split into three product nodes A, B, C by deriving A before B and C or by deriving C before A and B. This property is illustrated by Equation (14).

\[ D = A : (B : C) = (A : B) : C = A : B : C \]  

(14)

In summary, the structural properties of basic operations can be extended to an arbitrary number of operand nodes for any type of merging and an arbitrary number of product nodes for any type of splitting. These properties underpin the manipulation of individual rule bases in a rule based network. However, in order to facilitate this process further, two groups of advanced operations of network nodes are introduced in the next section.

6 Advanced Operations in Rule Based Networks

The first group of advanced operations includes modifications in nodes that allow the application of specific basic operations to these nodes. These operations are described briefly below.

In the case of two nodes with a common input whereby one of the nodes has an additional input, the other node can be augmented by this input such that an output merging operation can be applied to the two nodes. Also, when the intermediate variables between two nodes in sequence are crossing their paths, these variables can be permuted to avoid any crossings such that a horizontal merging operation can be applied to the two nodes. In the case when some outputs from a node are mapped to some of its inputs by means of identity feedback, the node can be represented equivalently as a node without feedback.

The second group of advanced operations includes ways of identifying an unknown node when all other individual node models in the network and the equivalent network model are specified. These operations usually involve the solution of systems of Boolean equations that relate to horizontal, vertical or output merging.

In summary, the advanced operations from the first group can be used for network analysis, whereas the ones from the second group can be used for network synthesis. In this sense, analysis relates to the simulation of fully specified rule based networks, whereas synthesis is concerned with the design of partially specified networks.

The next two sections illustrate the theoretical preliminaries introduced so far on different types of feedforward and feedback networks. The subsequent section evaluates rule based networks in comparative terms with respect to rule based systems.

7 Feedforward Rule Based Networks

Feedforward rule based networks are networks that may include only feedforward connections, i.e. connections from a particular node to another node in a subsequent layer. There are several types of feedforward networks, which are described briefly below.

The simplest type of feedforward rule based network has a single level and a single layer. This network is equivalent to a rule based system. A more complex type of feedforward rule based network has a single level within multiple layers. This network is equivalent to a queue of rule based systems. Similar type of complexity can be attributed to a feedforward rule based network that has multiple levels within a single layer. This network is equivalent to a stack of rule based systems. The most complex type of feedforward rule based network is the one with multiple levels and multiple layers. This network is equivalent to a grid of rule based systems.

8 Feedback Rule Based Networks

Feedback rule based networks are networks that may include not only feedforward but also feedback connections, i.e. connections from a particular node to the same or another node in the same or preceding layer. There are several types of feedback networks, which are described briefly below.

The simplest type of feedback rule based network has single local feedback. This feedback embraces only one individual node in the network. A more complex type of feedback rule based network has multiple local feedback. This feedback embraces by separate loops at least two individual nodes in a single level or a single layer. Similar type of complexity can be attributed to a feedback rule based network that has single global feedback. This feedback embraces by a single loop a set of at least two adjacent nodes in a single level or a single layer. The most complex type of feedback rule based network is the one that has multiple global feedback. This feedback embraces by separate loops at least two sets of at least two adjacent nodes each, whereby each set of nodes may be in a single level within multiple layers or in a single layer within multiple levels.
9 Evaluation of Rule Based Networks
The evaluation of the capability of rule based networks to model complex processes can be done for a particular type of rule bases. For example, rule based networks with fuzzy rule bases that are referred to here as fuzzy networks can be evaluated by either precise or approximate metrics. The functional metrics is based on precise functional indicators such as accuracy, efficiency and transparency. The linguistic metrics is based on approximate linguistic indicators such as distance and equivalence. These two metrics and the associated indicators are described briefly below.

The accuracy of fuzzy network models derived from data can be estimated precisely by the absolute mean error for all network outputs. In this case, the accuracy is obtained by taking the absolute value of the difference between the values of each data point and the corresponding simulated output, summing this difference for all data points and all outputs, and dividing it by overall number of data points.

The accuracy of fuzzy network models derived from expert knowledge can be estimated approximately by the overall number of defuzzifications. The assumption here is that each defuzzification deteriorates the model performance by approximating the actual output.

The efficiency of fuzzy network models can be estimated approximately by the overall number of rules. This can be obtained by calculating the number of rules for each node, multiplying this number by the number of outputs for the node and then summing these numbers for all nodes in the fuzzy network.

The transparency of fuzzy network models can be estimated approximately by mean of dividing the sum of nodes and intermediate variables by the sum of inputs and outputs. The assumption here is that each additional intermediate variable or node improves the transparency by taking into account some connections.

The comparative evaluation of fuzzy networks with respect to SRB and MRB systems shows some interesting results. In terms of accuracy, fuzzy networks usually perform much better than MRB systems but slightly worse than SRB systems. As far as efficiency is concerned, fuzzy networks are equivalent to SRB systems but more complex than MRB systems. And finally, the transparency of fuzzy networks is equivalent to that of MRB systems but higher than that of SRB systems.

These results show that fuzzy networks compare quite well with SRB and MRB systems. In particular, fuzzy networks are superior to SRB systems in terms of transparency and to MRB systems in terms of accuracy. At the same time, the inferiority of fuzzy networks to SRB systems in terms of accuracy can be reduced by introducing more values for the intermediate variables. This procedure does not affect the efficiency of fuzzy networks. On the other hand, the inferiority of fuzzy networks to MRB systems in terms of efficiency can be reduced by optimising the number of rules for the individual nodes. This procedure does not affect the transparency of fuzzy networks either.

10 Conclusion
The results presented in this paper highlight a new trend in artificial intelligence – the extension of rule based systems to rule based networks.

In this context, a rule based network is a generalisation of a rule based system that allows interactions among subprocesses in a complex process to be taken into account by the network model. Also, a rule based network facilitates the identification of the rule based model for a complex process with a large number of inputs. This is due to the possibility for considering initially some subsets of inputs. Once these subsets have been identified, the associated individual rule bases are manipulated to form an equivalent single rule base.

References: