Methods for Discovering Process Models and Their Properties in Data

Zbigniew Suraj
Chair of Computer Science
University of Rzeszow

Laboratory of Intelligent Decision Support Systems, Institute of Computing Science
Poznan University of Technology
and
Section of Intelligent Decision Support Systems and Soft Computing,
Committee on Computer Science of the Polish Academy of Sciences
January 11, 2011, Poznań
OUTLINE

- Introduction
- Considered Problems
- Data and Knowledge Representations
- Concurrent Models
- Experiments
- Software
- Concluding Remarks and Further Work
INTRODUCTION

• Data Mining and Knowledge Discovery are crucial and current research problems in the modern computer sciences.

• Discovering hidden relationships in data is a main goal of machine learning.

• In a lot of cases, data are generated by concurrent processes. Therefore, discovering concurrent system models is essential from the point of view of understanding the nature of modeled systems as well as explaining their behaviors.
INTRODUCTION

• A notion of concurrent systems can be understood widely.
• In a general case, a concurrent system is a system consisting of some processes, whose local states can coexist together and they are partly independent.
• We can treat systems consisting of economic processes, financial processes, biological processes, genetic processes, meteorological processes, etc. as concurrent systems.
INTRODUCTION

- This lecture concerns methods of concurrent system modeling on the basis of observations or specifications of their behaviors given in the form of different kinds of data tables.

- Data tables can include results of observations or measurements of specific states of concurrent processes. In this case, created models of concurrent systems are useful for analyzing properties of modeled systems, discovering the new knowledge about behaviors of processes, etc.

- Data tables can also include specifications of behaviors of concurrent processes.

- Then, created models can be a tool for verification of those specifications, e.g. during designing concurrent systems. Methods presented in this lecture can be used, i.e., in system designing or analyzing, data analysis, forecasting.
THE AIM OF RESEARCH

To present a general approach to inductive process modelling from data.

This work is a continuation of a research direction concerning relationships between rough set theory and concurrency.
SPECIFICATION OF CONCURRENT SYSTEMS BY INFORMATION SYSTEMS

EXAMPLE: LIGHT CONTROL

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>b</th>
<th>c</th>
</tr>
</thead>
<tbody>
<tr>
<td>$u_1$</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>$u_2$</td>
<td>0</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>$u_3$</td>
<td>0</td>
<td>0</td>
<td>2</td>
</tr>
</tbody>
</table>

- $u_1, u_2, u_3$ - states
- $a, b, c$ - movement directions
- $0, 1, 2$ - light color (red, green, green arrow)
HOW TO DESIGN CONCURRENT SYSTEMS FROM SPECIFICATIONS BY INFORMATION SYSTEMS?


MAIN IDEA

• Dependencies defined by attributes are conditions for coexistence of local states in global states

• One can use the existing methods for generating rules representing such dependencies (e.g., based on reducts or not)

• The set of rules is treated as knowledge representation for a given information system and defines its maximal extension, i.e., the set of global states consistent with all rules

• One can develop algorithms for designing, e.g., Petri Nets defining maximal extensions of information systems (data tables)
FORMALLY:

• Given information system A

• Define a theory Th(A) of A (consisting of a set of rules describing dependencies in A)

• Th(A) defines the maximal extension of A

• Construct a Petri net consistent with Th(A)
ADVANTAGES

• Complex Petri Nets can be generated automatically from their specification by data tables

• Petri Net can be adaptively modified with changes of data
IMPORTANT QUESTIONS

• Which kinds of rules should be used (e.g., also non-deterministic, probabilistic)?

• How to characterize the expressibility of different rule sets?

• How to extend the approach by adding information on transition relation or temporal dependencies?


OUR METHODOLOGY

It is based on:

• rough set theory (Z. Pawlak 1982),
• Boolean reasoning (G. Boole, XIX cent.),


ROUGH SETS

• Rough set theory introduced by prof. Zdzisław Pawlak (1982) from Poland provides advanced and efficient methods of data analysis and knowledge extraction.
THE CREATOR OF ROUGH SETS

More information about publications, software and biographies of distinctive researchers in the rough set theory and applications can be found in the Rough Set Database under address:

http://rsds.univ.rzeszow.pl
Welcome to our service

a bibliographic database on wide aspects of rough sets.

The service has been developed in order to facilitate the creation of bibliography, for various types of publications. At present it is possible to create bibliography in HTML or in BibTex format. In order to broaden the service contents it is possible to append new data using specially dedicated form. After appending data online, the database is updated automatically. If one prefers sending a data file to the database administrator, please be aware that the database is update once a month. Detailed information on how to use the service can be found in Help section. Any comments about our service will be helpful and greatly appreciated. Please post them to the database administrator who permanently carries out work on improving the service and broadening its possibilities.

!!! Important !!!
For proper creating of files in BibTex format it is required to allow Cookies in your Web browser’s options.
Boolean reasoning makes a base for solving a lot of decision and optimization problems. Especially, it plays a special role during generation of decision rules.
DATA AND KNOWLEDGE

- In our approach:
  - the data are represented in the form of data tables (information/decision systems, specialized tables) that are used as the specification of the structure and behaviour of modelled systems,
  - the knowledge about the structure and behaviour of the modelled system is extracted from a given data table and represented in the form of IF...THEN rules.
CONCURRENT MODEL - PETRI NETS

- Petri nets are the graphical and mathematical tool for modeling of different kinds of phenomena, especially those, where actions executed concurrently play a significant role.

- General net theory proposed by C.A. Petri in 1962.

- It is a generalisation of automata theory such that the concept of concurrently occurring events can be expressed.
THE CREATOR OF GENERAL NET THEORY

Carl Adam Petri - German mathematician and computer scientist, a honorary professor at the University of Hamburg.

His PhD thesis:

*Kommunikation mit Automaten (Communication with automata).*
Bonn: Institut für Instrumentelle Mathematik, Schriften des IIM Nr. 2, 1962
WHY DO WE USE COLOURED PETRI NETS?

• In particular, coloured Petri nets have:
  - solid mathematical backgrounds
  - intuitive graphical (visual) representation
  - the possibility to refine models (hierarchical representation)
  - many practical applications
  - available computer tools for the users (e.g. CPN Tools, CPNetwork)
THE CREATOR OF COLOURED PETRI NETS

Kurt Jensen (1950 - ) – Danish computer scientist, professor of Aarhus University.

First article:
THE PROBLEMS

1. The synthesis problem
2. The decomposition problem
3. The reconstruction problem
4. The prediction problem
1. THE SYNTHESIS PROBLEM

**INPUT:** A given data generated by a system of concurrent processes.

**OUTPUT:** A concurrent model of the system discovered and constructed on the base of knowledge extracted from a given data in such a way that model global states are consistent with the extracted knowledge from the data.

An information system can include the knowledge about global states of a given concurrent system, understood as vectors of local states of processes making up the concurrent system, whereas a dynamic information system can include additionally the knowledge about transitions between global states of the concurrent system. Specialized matrices are designed for specifying undesirable states of a given concurrent system (i.e. those states, which cannot hold together) and undesirable transitions between its states.
2. THE DECOMPOSITION PROBLEM

**INPUT:** A given data (an information system) generated by a system of concurrent processes, and concurrent model constructed on the base the given data (as a result of the synthesis algorithm).

**OUTPUT:** A family of components (subsystems) together with the links (rules) binding those components which are sufficient to build the original concurrent model such that model global states are consistent with the extracted knowledge from the given data.

Decomposition of data tables into smaller subtables connected by suitable rules is also possible. Those subtables make up modules of a system. Local states of processes represented in a given subtable are linked by means of a functional dependency.
3. THE RECONSTRUCTION PROBLEM

**INPUT:** A given data table representing the specification of concurrent process model obtained by the synthesis algorithm, and a new data representing a new specification of the modelled system.

**OUTPUT:** A plan (algorithm) of the reconstruction of a given concurrent process model satisfying the new specification represented by a new data.

In this lecture, the problems of reconstruction of models and prediction of their changes in time are also taken up. Those problems occur as a result of appearing the new knowledge about modeled systems and their behaviors. The new knowledge can be expressed by appearing new global states, new transitions between states, new local states of individual processes or new processes in modeled systems.
4. THE PREDICTION PROBLEM

**INPUT:** A given concurrent model described by temporal data (ordered in time).

**OUTPUT:** A set of prediction rules which can be used to predict future changes of the model.

A prediction method proposed in this lecture points at the character of model changes in time. For representing prediction rules, both prediction matrices and Pawlak’s flow graphs are used.
A DIAGRAM OF PROBLEM SOLVING

DATA REPRESENTATION

Knowledge Representation

Rough set methods

Data

Rules

Structure

Dynamics

Model

CONCURRENT MODEL
Scheme 1: The synthesis problem

A real system

Observations / Measurements / Designing

A description / specification

Information system / Dynamic information system / Decomposed information system / Specialized data tables

Rough set methods

Knowledge

Rules

Transformations

Concurrent model

Coloured Petri nets

Analysis

New knowledge on the system
Scheme 2: The reconstruction problem

A real system at time $t_1$

A description / specification

Knowledge

Concurrent model

Reconstruction

A real system at time $t_2$

A description / specification

Knowledge

New concurrent model
In our approach, the net model can be built on the basis of a decomposed information system $S$ describing a given concurrent system.

If the description of a concurrent system changes (i.e., a new information system $S^*$ appears), we have to reconstruct the net model representing the concurrent system.

The structure of a constructed net is determined on the basis of components of an information system. So, changing reducts and components in $S$ can lead to a change in the structure of a net model. In that case, we would like to know how the reducts and components change when the new information about the system behavior appears.

The idea of the reconstruction of a net model constructed for $S$ can be presented graphically using a block diagram as in Figure.
Fig. RECONSTRUCTION PROBLEM

Information system $S$ -> Reducts of $S$ -> Components of $S$ -> Covering of $S$ -> Net model of $S$

Information system $S^*$ -> Reducts of $S^*$ -> Components of $S^*$ -> Covering of $S^*$ -> Net model of $S^*$

New behaviour requirements
New knowledge

Decomposition

Computing new reducts

Computing new components

Modification of a net structure and/or guard expressions
RECONSTRUCTION PROBLEM
(Remarks)

• The renewed computation of reducts and components of information systems is time-consuming, because algorithms are NP-hard.

• So, it is important to compute new reducts and components in an efficient way, i.e., without the necessity of renewed computations. Some method has been proposed in:


• In the approach presented there, a particular case has been considered, when the new description (in the form of an information system $S^*$) of a modeled system includes one new object (global state) with relation to the old description (in the form of an information system $S$).
Scheme 4: The prediction problem

1. A real system changing in time
   - Observations / Measurements

2. A description
   - Temporal information system

3. Knowledge on system behaviour in consecutive time windows
   - Rough set methods
   - Reducts, components, rules

4. Concurrent models in consecutive time windows
   - Transformation
   - Coloured Petri nets

5. Prediction of model changes
   - Analysis
PREDICTION RULES
(Algorithm)

• Split a given temporal information system into time windows and obtain a set of all the time windows.

• For each time window from the set and each attribute, compute a set of all functional \( \{a\}\)-reducts of \( A - \{a\} \) and obtain a temporal table of functional reducts) whose columns are labeled with attributes from \( A \) whereas rows, with consecutive time windows from \( S \). The cells of such a table contain sets of functional relative reducts.

• For each attribute \( a \) in \( A \), build a temporal decision system. Attributes of this system are labeled with the consecutive time windows (the last attribute is treated as a decision). Each row represents a sequence of sets of functional relative reducts which appeared in consecutive time windows.

• For each attribute \( a \) in \( A \), compute prediction rules from the temporal decision system. In order to represent such rules, use e.g. flow graphs proposed by Z. Pawlak.

DATA REPRESENTATIONS

1. A data table (Pawlak’s information system)
2. A composed data table (dynamic information system)
3. Specialized data tables (forbidden state/transition matrices)
Data Table (1)

Information system $S = (U,A)$

Interpretation: Processes of $S$

<table>
<thead>
<tr>
<th>$U \backslash A$</th>
<th>$a$</th>
<th>$b$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$u_1$</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>$u_2$</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>$u_3$</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>$u_4$</td>
<td>2</td>
<td>0</td>
</tr>
</tbody>
</table>

Interpretation:

A local state of a given process
INTERPRETED DATA TABLE
- A Communication System -

**PROCESSES:**

- Device a sender/receiver
- Device b sender/receiver

**Bus**

**ACTIONS:**

- 0 – sending
- 1 – receiving
- 2 – disconnecting

<table>
<thead>
<tr>
<th>( U \backslash A )</th>
<th>( a )</th>
<th>( b )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( u1 )</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>( u2 )</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>( u3 )</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>( u4 )</td>
<td>2</td>
<td>0</td>
</tr>
</tbody>
</table>

INTERPRETED DATA TABLE
A DATA TABLE
(REMARKS)

Advantages:
- Simple and intuitive specification of concurrent system
- Easy for interpretation into concurrent system concepts

Disadvantages:
- Partial information on dynamics of concurrent systems (only a set of global states). Lack of information about transition relation.
**Composed Data Table (2)**

Dynamic information system $DS = (U, A, E, T, u0)$

An underlying system $S$ of $DS$  

<table>
<thead>
<tr>
<th>$U \backslash A$</th>
<th>$a$</th>
<th>$b$</th>
<th>$U \backslash E$</th>
<th>$e_1$</th>
<th>$e_2$</th>
<th>$e_3$</th>
<th>$e_4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$u_1$</td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$u_4$</td>
</tr>
<tr>
<td>$u_2$</td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$u_3$</td>
</tr>
<tr>
<td>$u_3$</td>
<td>0</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$u_1$</td>
</tr>
<tr>
<td>$u_4$</td>
<td>2</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$u_2$</td>
</tr>
</tbody>
</table>

A sequence: $u_1 \rightarrow u_4 \rightarrow u_2 \rightarrow u_3 \rightarrow u_1$

A transition system $TS$ of $DS$

Interpretation:
- **Global states of $S$**
- **Local processes of $S$**
- **A next global state**
Composed Data Table $(2')$

- **Dynamic information system**
  - weak specification

\[ S_T = (U, A \cup A') \]

### Previous states (conditional attributes)

### Next states (decisions)

<table>
<thead>
<tr>
<th>( U \setminus (A \cup A') )</th>
<th>( a )</th>
<th>( b )</th>
<th>( a' )</th>
<th>( b' )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( u_1 )</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>( u_2 )</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>( u_3 )</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>( u_4 )</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

It represents only positive information about a transition relation \( T \).

A sequence: \( u_1 \rightarrow u_4 \rightarrow u_2 \rightarrow u_3 \rightarrow u_1 \)
### Composed Data Table (2")

- **Dynamic information system**
  - **strong specification**

\[
S_T = (U, A \cup A' \cup \{d\})
\]

#### (conditional attributes)

#### (decision)

![Diagram showing Previous states and Next states]

<table>
<thead>
<tr>
<th>(U \setminus A \cup A' \cup {d})</th>
<th>(a)</th>
<th>(b)</th>
<th>(a')</th>
<th>(b')</th>
<th>(d)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(u_1)</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>(u_2)</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>(u_3)</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>(u_4)</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

It represents positive and negative information about a transition relation \(T\).
Advantages:
- Simple and intuitive specification of concurrent system, easy for interpretation into concurrent system concepts
- Full information on dynamics of concurrent systems (a set of global states together with a transition relation).
- Opportunity for different representations of the transition relation.
- Two structures of concurrent system models are considered: synchronous and asynchronous. A synchronous model enables us to generate the so-called maximal consistent extension of a given information system. Such an extension includes all possible global states consistent with all rules extracted from the original data table. An asynchronous model enables us to find all possible transitions between global states of a given concurrent system, for which only one process changes its local state.

Disadvantages:
- A rise of the tabular representation size.
### Specialized Data Tables (3)

#### Forbidden state matrix \( FSM \)

<table>
<thead>
<tr>
<th></th>
<th>( (a,0) )</th>
<th>( (a,1) )</th>
<th>( (a,2) )</th>
<th>( (b,0) )</th>
<th>( (b,1) )</th>
<th>( (b,2) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( (a,0) )</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>( (a,1) )</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>( (a,2) )</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>( (b,0) )</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>( (b,1) )</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>( (b,2) )</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

**Interpret.:** A local state of process \( b \)

**Interpret.:** A local state of process \( a \)

**Interpret.:** Process \( a = 1 \) can coexist with process \( b = 0 \)
### Specialized Data Table (3)

**Forbidden transition matrix** $FTM$

<table>
<thead>
<tr>
<th></th>
<th>$(a',0)$</th>
<th>$(a',1)$</th>
<th>$(a',2)$</th>
<th>$(b',0)$</th>
<th>$(b',1)$</th>
<th>$(b',2)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$(a,0)$</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$(a,1)$</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>$(a,2)$</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$(b,0)$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$(b,1)$</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>$(b,2)$</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

Interpret.: A current local state of a given process

Interpret.: A next local state of a given process
SPECIALIZED DATA TABLE
(REMARKS)

Advantages:
- Precise information about structure and dynamics of concurrent systems.
- Specialized matrices are designed for specifying undesirable states of a given concurrent system (i.e. those states, which cannot hold together) and undesirable transitions between its states.

Disadvantages:
- A considerable rise of sizes of the tabular representations.
Extracting Knowledge from Data

Information system

Discernibility matrix

Discernibility function

Prime implicants

Reducts

Rules

Boolean reasoning
KNOWLEDGE REPRESENTATIONS

• Deterministic rules

• Inhibitory rules
DETERMINISTIC RULES

IF (expression) THEN (action)
or
... ⇒ attribute = value

• Example: IF (a = 1) THEN (b = 0)

Theory of information system based on deterministic rules can have nonstandard models:

INHIBITORY RULES

• **IF** *(expression) THEN (no action)*

or

\[ \cdots \Rightarrow attribute \neq value \text{ (inhibitory rule)} \]

• **Example**: **IF** *(a = 0) THEN (b \neq 1)*

Inhibitory rules do not allow nonstandard models

Maximal Consistent Extensions (intuitively)

Information system

Rules

Maximal consistent extension of information system
Example: Maximal Consistent Extensions

All true and realizable rules for $S$:

- $a_1 = 1 \rightarrow a_2 = 0$
- $a_1 = 2 \rightarrow a_2 = 0$
- $a_2 = 1 \rightarrow a_1 = 0$
- $a_2 = 2 \rightarrow a_1 = 0$

All true and realizable rules for $S$ are true for $(0, 0)$. 

hidden global state
**Example:** Maximal Consistent Extensions

All true and realizable inhibitory rules for $S$:

- $a_1 = 0 \rightarrow a_2 \neq 0$, $a_2 = 1 \rightarrow a_1 \neq 1$
- $a_1 = 1 \rightarrow a_2 \neq 1$, $a_2 = 1 \rightarrow a_1 \neq 2$
- $a_1 = 1 \rightarrow a_2 \neq 2$, $a_2 = 0 \rightarrow a_1 \neq 0$
- $a_1 = 2 \rightarrow a_2 \neq 1$, $a_2 = 2 \rightarrow a_1 \neq 1$
- $a_1 = 2 \rightarrow a_2 \neq 2$, $a_2 = 2 \rightarrow a_1 \neq 2$

Not all true and realizable inhibitory rules for $S$ are true for $(0, 0)$. 
INHIBITORY RULES
(Remarks)

It can represent essentially more information encoded in information systems than deterministic ones.

This fact was a reason to use inhibitory rules in classifiers as well as in the concurrent system design.
Maximal Consistent Extensions
(Remarks)

• There exist information systems $S = (U, A)$ for which $|\text{Ext}(S) \setminus U| = \exp(|U|)$

• There are no polynomial algorithms for construction of the set $\text{Ext}(S)$

• There exists polynomial algorithm which for a given object $v \in V$ recognizes if $v$ belongs to $\text{Ext}(S)$ or not

A given information system $S$:

$S = (U, \{X \cup Y\})$

Decomposition w.r.t. a reduct $R$ of the system $S$

A normal component of $S$:

$S_i = (U_i, X_i \cup Y_i)$

$Y_i = \{a \in A : X_i \text{ is a reduct w.r.t. } a$ and $X_i \rightarrow \{a\}\}$

Degenerated component:

$S_i = (U_i, \{a\})$ and the attribute $a$ does not appear in any normal component
Covering of Information Systems

C-covering of $S = (U, A)$ with links $C$:

A set of components:

$$S_1 = (U_1, X_1 \cup Y_1)$$

$$...$$

$$S_k = (U_k, X_k \cup Y_k)$$

where:

$$X_1 \cup ... \cup X_k \cup Y_1 \cup ... \cup Y_k = A$$
Links between Components

A set of links $C$ includes:

- rules corresponding dependencies between attribute values of component $S_i$, where $i = 1,...,k$, called **internal links** of component $S_i$,

- rules corresponding dependencies between attribute values of component $S_i$ and attribute values from outside of $S_i$, where $i = 1,...,k$, called **external links** of component $S_i$.
Example: Decomposition of Information Systems

A given information system $S = (U, A)$ with $A = \{a, b, c\}$:

<table>
<thead>
<tr>
<th>$U \setminus A$</th>
<th>$a$</th>
<th>$b$</th>
<th>$c$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$u_1$</td>
<td>0</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>$u_2$</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>$u_3$</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>$u_4$</td>
<td>1</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

Reducts of $S$:
- $R1 = \{b, c\}$, $A - R1 = \{a\}$
- $R2 = \{a, c\}$

Decomposition w.r.t. reduct $R1$:

**Normal component $S_1$**

<table>
<thead>
<tr>
<th>$U' \setminus A$</th>
<th>$b$</th>
<th>$a$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$u_1$</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>$u_2$</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>$u_3$</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

**Degenerated component $S_2$**

<table>
<thead>
<tr>
<th>$U'' \setminus A$</th>
<th>$c$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$u_1$</td>
<td>1</td>
</tr>
<tr>
<td>$u_3$</td>
<td>0</td>
</tr>
<tr>
<td>$u_4$</td>
<td>2</td>
</tr>
</tbody>
</table>
Descriptive Set of Attributes

A set of attributes $B \subseteq A$ is called a descriptive set for $S$ if there exists a set of rules $Q \subseteq Rul(S)$ constructed over the attributes from $B$ only such that $Ext(S)$ coincides with the set of all tuples (objects) from $V$ for which all rules from $Q$ are true.
Irreducible Descriptive Set

• A descriptive set $B$ for $S$ is called irreducible if each proper subset of $B$ is not a descriptive set for $S$. 
Remarks

• We showed that for any information system $S$ there exists only one irreducible descriptive set of attributes, and create a polynomial algorithm for this set construction.

• We proposed a polynomial algorithm recognizing if there exists a cover of the irreducible descriptive set by reducts of $S$.

• The obtained results will be useful in applications of information systems connected with analysis and design of concurrent systems.


Concurrent Models
- coloured Petri nets -

Definition of CP-net:

\[ CPN = (\Sigma, P, T, A, N, C, G, E, I) \]

\( \Sigma \) – a nonempty finite set of types

\( P \) – a finite set of places, \( T \) – a finite set of transitions

\( A \) – a finite set of arcs, \( N \) – a node function

\( C \) – a colour function, \( G \) – a guard function

\( E \) – an arc expression, \( I \) – an initialization function
Example: Coloured Petri Nets

```
color A = with a1 | a2 | a3;
color B = with b1 | b2;

var x1, x3, y1 : A;
var x2, x4, y4 : B;
```
Solving Synthesis Problem

Input
- Observations
- Measurements
- Designer specification

Data tables

Solution
- Permissible states
- Permissible transitions between states

Net models

Knowledge
- Rules
- Specialized data tables
- * positive rules
  * inhibitor rules
  * transition rules etc
- * information systems
  * dynamic information systems
  * decomposed information systems
  * forbidden state matrix, forbidden transition matrix

* asynchronous in the form of nets with inhibitor expressions
* asynchronous in the form of coloured Petri nets
* synchronous in the form of coloured Petri nets

67
TRANSFORMATION OF DATA INTO CONCURRENT MODEL

Input: A given data table and a set of rules extracted from the data table.

Output: A resulting concurrent model (coloured Petri net).

Step 1. Construct a net representing the set of processes of a given data table.

Step 2. Add to the net obtained in Step 1 a net defined by the set of rules of a given data table, corresponding to all nontrivial dependencies (connections) between the values of attributes belonging to different processes of the data table.

The connections between processes represent constraints which must be satisfied when these processes in the system.

Step 3. Describe the elements (places, transitions and arcs) of the net defined in steps 1-2 according to the definition of a coloured Petri net.
A Scheme of the Transformation

Description

Information system/decomposed system/specialized tables

Set of all minimal rules

Boolean expressions

Net model in the form of CP-net

Net structure

Guard expressions

Concurrent model
Methods for Constructing Model

Information system

Computing reducts

Computing rules for each reduct separately

Computing all inhibitor rules

Creating asynchronous net model

Asynchronous model in the form of a CP-net

Creating synchronous net model

Synchronous model in the form of a CP-net

Rough set methods (discernibility matrix, discernibility function)
Methods for Generating Rules

Computing rules with respect to reducts

For each reduct we compute *internal* and *external* rules.

*Internal rules.* For each attribute $a$ from $R$ we compute rules corresponding to a dependency:

$$(R - \{a\}) \rightarrow \{a\}$$

*External rules.* For each attribute $a$ from outside $R$ we compute rules corresponding to a dependency:

$$R \rightarrow \{a\}$$
Methods for Generating Rules

Computing rules directly from a data table

For each attribute \( a \) of \( S=(U,A) \) we compute rules corresponding to a dependency:

\[
(A - \{a\}) \rightarrow \{a\}
\]
Methods for Constructing Model

Dynamic information system - weak specification

Computing underlying inhibitor rules

Creating synchronous net model

Synchronous model in the form of a CP-net

Dynamic information system - strong specification

Computing transition inhibitor rules

Creating asynchronous net model

Asynchronous model in the form of a CP-net

Rough set methods (discernibility matrix, discernibility function)
Computing a Guard Expression

The set of all minimal rules in \( S \)

\[ \iff \neg x \lor y \]

\[ \iff x \lor x \iff x, \quad x \land x \iff x \]

The Boolean expression (minimal disjunctive normal form)

The guard expression
Example 1

IF $a_1$ THEN $b_0$
IF $a_2$ THEN $b_0$
IF $b_1$ THEN $a_0$
IF $b_2$ THEN $a_0$

$(a_0 \text{ AND } b_0) \text{ OR } (a_0 \text{ AND } b_1) \text{ OR } (a_0 \text{ AND } b_2) \text{ OR } (a_1 \text{ AND } b_0) \text{ OR } (a_2 \text{ AND } b_0)$

SYNONRONOUS CONCURRENT MODEL:

<table>
<thead>
<tr>
<th>$A \cup A$</th>
<th>$a$</th>
<th>$b$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$u_1$</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>$u_2$</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>$u_3$</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>$u_4$</td>
<td>2</td>
<td>0</td>
</tr>
</tbody>
</table>
Example 2

<table>
<thead>
<tr>
<th>UA</th>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>u1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>u2</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>u3</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>u4</td>
<td>2</td>
<td>0</td>
</tr>
</tbody>
</table>

IF a1 THEN b0
IF a2 THEN b0
IF b1 THEN a0
IF b2 THEN a0

(a0 AND b0) OR (a0 AND b1) OR (a0 AND b2)
OR (a1 AND b0) OR (a2 AND b0)

ASYNCHRONOUS CONCURRENT MODEL:

```
color a = with a0 | a1 | a2
color b = with b0 | b1 | b2
var xa, ya : a;
var xb, yb : b;
```
The Synthesis Problem (with decomposition)

- Information system
- Decomposition
- System components
- Internal and external links for components
- Boolean expressions
- Model in the form of CP-net
- Net structure
- Guard expression
**NET MODEL RECONSTRUCTION**

Information system (old) $S$ → Decomposition → Components and linkings → Net model 1 → Modification of a net structure and/or guard expressions → Comparison → Components and linkings → Decomposition → Information system (new) $S^*$
A temporal information system

Split a given system into time windows

Compute functional relative reducts

Construct decision systems (attribute values: families of functional relative reducts)

Construct prediction matrix or flow graphs

Prediction rules

PREDICTION OF MODEL PROPERTY CHANGE
PREDICTION OF MODEL PROPERTY CHANGE

A Pawlak’s flow graph expressive prediction rules
EXAMPLE 1: A GENETIC SYSTEM

<table>
<thead>
<tr>
<th>( U \setminus A )</th>
<th>( g1 )</th>
<th>( g2 )</th>
<th>( g3 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( u_1 )</td>
<td>A</td>
<td>G</td>
<td>C</td>
</tr>
<tr>
<td>( u_2 )</td>
<td>C</td>
<td>A</td>
<td>G</td>
</tr>
<tr>
<td>( u_3 )</td>
<td>C</td>
<td>G</td>
<td>A</td>
</tr>
<tr>
<td>( u_4 )</td>
<td>C</td>
<td>C</td>
<td>C</td>
</tr>
<tr>
<td>( u_5 )</td>
<td>A</td>
<td>G</td>
<td>A</td>
</tr>
<tr>
<td>( u_6 )</td>
<td>C</td>
<td>G</td>
<td>C</td>
</tr>
<tr>
<td>( u_7 )</td>
<td>A</td>
<td>C</td>
<td>A</td>
</tr>
<tr>
<td>( u_8 )</td>
<td>A</td>
<td>A</td>
<td>C</td>
</tr>
<tr>
<td>( u_9 )</td>
<td>A</td>
<td>C</td>
<td>G</td>
</tr>
<tr>
<td>( u_{10} )</td>
<td>G</td>
<td>C</td>
<td>A</td>
</tr>
<tr>
<td>( u_{11} )</td>
<td>G</td>
<td>C</td>
<td>C</td>
</tr>
</tbody>
</table>

\[ U = \{ u_1, \ldots, u_{11} \} \]

\[ A = \{ g_1, g_2, g_3 \} \]

\[ V_{g_1} = V_{g_2} = V_{g_3} = \{ A, G, C \} \]

Attribute values represent allele (states of genes):

- \( A \) – Adenine
- \( C \) – Cytosine
- \( G \) – Guanine

Global states are interpreted as chromosomes (gene chains), attributes as genes.
PROBLEMS

Define on the base of the knowledge extracted from a given data table coming from observations:

- all global states consistent with the knowledge extracted from a given data table,

- all possible transitions between global states of the system defining pointed mutation (only one of genes changes its value).
SYNCHRONOUS CONCURRENT MODEL

\[
\begin{align*}
&g_1 \xrightarrow{\text{pg}_1} 1'g_1_A \\
&g_2 \xrightarrow{\text{pg}_2} 1'g_2_G \\
&g_3 \xrightarrow{\text{pg}_3} 1'g_3_C \\
&1'g_1_A \xrightarrow{1'xg_1} 1'g_2_G \\
&1'g_2_G \xrightarrow{1'xg_2} 1'g_3_C \\
&1'g_3_C \xrightarrow{1'xg_3} 1'g_1_A \\
&\text{[\not (((yg_2=g_2_A)\land also(yg_1=g_1_G))} \\
&\lor else (((yg_2=g_2_G)\land also(yg_1=g_1_G))) \\
&\lor else (((yg_1=g_1_C)\land also(yg_3=g_3_C)\land also(yg_2=g_2_A))) \\
&\lor else (((yg_1=g_1_G)\land also(yg_3=g_3_G)\land also(yg_2=g_2_C))) \\
&\lor else (((yg_1=g_1_C)\land also(yg_3=g_3_G)\land also(yg_2=g_2_C))) \\
&\lor else (((yg_2=g_2_G)\land also(yg_3=g_3_G)\land also(yg_1=g_1_C))) \\
&\lor else (((yg_2=g_2_C)\land also(yg_3=g_3_C)\land also(yg_1=g_1_C))) \\
&\lor else (((yg_2=g_2_C)\land also(yg_3=g_3_A)\land also(yg_1=g_1_C))) \\
&\lor else (((yg_2=g_2_A)\land also(yg_3=g_3_A)\land also(yg_1=g_1_C))) \\
&\lor else (((yg_2=g_2_A)\land also(yg_3=g_3_G)\land also(yg_1=g_1_A))) \\
&\lor else (((yg_3=g_3_A)\land also(yg_2=g_2_A)\land also(yg_1=g_1_A))) \\
\}]
\end{align*}
\]
ANALYSIS RESULTS OF THE GENETIC SYSTEM

<table>
<thead>
<tr>
<th>( U \setminus A )</th>
<th>( g1 )</th>
<th>( g2 )</th>
<th>( g3 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( u_1 )</td>
<td>A</td>
<td>G</td>
<td>C</td>
</tr>
<tr>
<td>( u_2 )</td>
<td>C</td>
<td>A</td>
<td>G</td>
</tr>
<tr>
<td>( u_3 )</td>
<td>C</td>
<td>G</td>
<td>A</td>
</tr>
<tr>
<td>( u_4 )</td>
<td>C</td>
<td>C</td>
<td>C</td>
</tr>
<tr>
<td>( u_5 )</td>
<td>A</td>
<td>G</td>
<td>A</td>
</tr>
<tr>
<td>( u_6 )</td>
<td>C</td>
<td>G</td>
<td>C</td>
</tr>
<tr>
<td>( u_7 )</td>
<td>A</td>
<td>C</td>
<td>A</td>
</tr>
<tr>
<td>( u_8 )</td>
<td>A</td>
<td>A</td>
<td>C</td>
</tr>
<tr>
<td>( u_9 )</td>
<td>A</td>
<td>C</td>
<td>G</td>
</tr>
<tr>
<td>( u_{10} )</td>
<td>G</td>
<td>C</td>
<td>A</td>
</tr>
<tr>
<td>( u_{11} )</td>
<td>G</td>
<td>C</td>
<td>C</td>
</tr>
<tr>
<td>( u_{12} )</td>
<td>A</td>
<td>C</td>
<td>C</td>
</tr>
</tbody>
</table>

New global state consistent with all rules generated from the given genetic system.
TRANSITION GRAPH OF ASYNCHRONOUS TRANSITIONS BETWEEN GLOBAL STATES
(only one process changes its local state)

New state u12 and its relationships with another states
Our task is to design a traffic signals control for this crossroads.

EXAMPLE 2: TRAFFIC SIGNALS

\[ A = \{a, b, c\} \]
\[ V = V_a = V_b = V_c = \{1, 2, 3, 4\} \]
\[ DESC(A, V) = \{(a,1), (a,2), (a,3), (a,4), (b,1), (b,2), (b,3), (b,4), (c,1), (c,2), (c,3), (c,4)\} \]

The meaning of attribute values:
1 – red
2 – green arrow (left turn)
3 – green arrow (right turn)
4 - green
# THE FORBIDDEN STATE MATRIX

<table>
<thead>
<tr>
<th></th>
<th>(a, 1)</th>
<th>(a, 2)</th>
<th>(a, 3)</th>
<th>(a, 4)</th>
<th>(b, 1)</th>
<th>(b, 2)</th>
<th>(b, 3)</th>
<th>(b, 4)</th>
<th>(c, 1)</th>
<th>(c, 2)</th>
<th>(c, 3)</th>
<th>(c, 4)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a, 1)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>(a, 2)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>(a, 3)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>(a, 4)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>(b, 1)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>(b, 2)</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>(b, 3)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>(b, 4)</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>(c, 1)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>(c, 2)</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>(c, 3)</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>(c, 4)</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
### The Forbidden Transition Matrix

<table>
<thead>
<tr>
<th></th>
<th>(a, 1)</th>
<th>(a, 2)</th>
<th>(a, 3)</th>
<th>(a, 4)</th>
<th>(b, 1)</th>
<th>(b, 2)</th>
<th>(b, 3)</th>
<th>(b, 4)</th>
<th>(c, 1)</th>
<th>(c, 2)</th>
<th>(c, 3)</th>
<th>(c, 4)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a, 1)</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>(a, 2)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>(a, 3)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>(a, 4)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>(b, 1)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>(b, 2)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>(b, 3)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>(b, 4)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>(c, 1)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>(c, 2)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>(c, 3)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>(c, 4)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
[not( ((ya=a2) andalso (yb=b2)) orelse ((ya=a2) andalso (yb=b4)) orelse ((ya=a2) andalso (yc=c2))
orelse ((ya=a2) andalso (yc=c3)) orelse ((ya=a2) andalso (yc=c4)) orelse ((ya=a3) andalso (yb=b2))
orelse ((ya=a3) andalso (yb=b4)) orelse ((ya=a4) andalso (yb=b2)) orelse ((ya=a4) andalso (yb=b4))
orelse ((ya=a4) andalso (yc=c2)) orelse ((ya=a4) andalso (yc=c3)) orelse ((ya=a4) andalso (yc=c4))
orelse ((yb=b2) andalso (yc=c2)) orelse ((yb=b2) andalso (yc=c4)) orelse ((yb=b3) andalso (yc=c2))
orelse ((yb=b3) andalso (yc=c4)) orelse ((yb=b4) andalso (yc=c2)) orelse ((yb=b4) andalso (yc=c4))
orelse ((xa=a1) andalso (ya=a1)) orelse ((xb=b1) andalso (yb=b1)) orelse ((xc=c1) andalso (yc=c1))]]

```
color a = with a1 | a2 | a3 | a4;
color b = with b1 | b2 | b3 | b4;
color c = with c1 | c2 | c3 | c4;
var xa, ya : a; var xb, yb : b; var xc, yc : c;
```
THE REACHABLE STATES OF THE DESIGNED SYSTEM

<table>
<thead>
<tr>
<th>$U \setminus A$</th>
<th>$a$</th>
<th>$b$</th>
<th>$c$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$u_1$</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>$u_2$</td>
<td>1</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>$u_3$</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$u_4$</td>
<td>1</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>$u_5$</td>
<td>1</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>$u_6$</td>
<td>4</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$u_7$</td>
<td>1</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>$u_8$</td>
<td>1</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>$u_9$</td>
<td>4</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>$u_{10}$</td>
<td>3</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$U \setminus A$</th>
<th>$a$</th>
<th>$b$</th>
<th>$c$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$u_{11}$</td>
<td>3</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>$u_{12}$</td>
<td>3</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>$u_{13}$</td>
<td>1</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>$u_{14}$</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$u_{15}$</td>
<td>3</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>$u_{16}$</td>
<td>1</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>$u_{17}$</td>
<td>2</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>$u_{18}$</td>
<td>1</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>$u_{19}$</td>
<td>3</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>$u_{20}$</td>
<td>1</td>
<td>3</td>
<td>1</td>
</tr>
</tbody>
</table>
EXAMPLE 2: ECONOMIC PROCESSES

Economic processes (exchange rates, oil price)

<table>
<thead>
<tr>
<th>$U \setminus A$</th>
<th>$dollar$</th>
<th>$euro$</th>
<th>$oil$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$u_1$</td>
<td>-1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>$u_2$</td>
<td>0</td>
<td>0</td>
<td>-1</td>
</tr>
<tr>
<td>$u_3$</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$u_4$</td>
<td>-1</td>
<td>1</td>
<td>-1</td>
</tr>
<tr>
<td>$u_5$</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>$u_6$</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$u_7$</td>
<td>1</td>
<td>1</td>
<td>-1</td>
</tr>
<tr>
<td>$u_8$</td>
<td>1</td>
<td>0</td>
<td>-2</td>
</tr>
<tr>
<td>$u_9$</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$u_{10}$</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>$u_{11}$</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

$U = \{ u_1, \ldots, u_{11} \}$

$A = \{ dollar, euro, oil \}$

$DESC(A,V) = \{ (dollar,-1), (dollar,0), (dollar,1), (euro,0), (euro,1), (oil,-2), (oil,-1), (oil,0), (oil,1) \}$
Objects represent data from consecutive days.

The meaning of local states of processes:
- 0 – no change, i.e. [-0.5%,0.5%]
- -1 – decrease [-1.5%,-0.5%)
- -2 – decrease [-2.5%,-1.5%)
- 1 – increase (0.5%,1.5%]
- 2 – increase (1.5%,2.5%]

The problem is the following:

On the basis of possessed knowledge (deriving from observations) determine:
- all states in which the processes can hold
- all transitions between states which can appear
INFORMATION SYSTEM: A WEAK SPECIFICATION

<table>
<thead>
<tr>
<th>$U \setminus A$</th>
<th>dollar</th>
<th>euro</th>
<th>oil</th>
<th>dollar'</th>
<th>euro'</th>
<th>oil'</th>
</tr>
</thead>
<tbody>
<tr>
<td>$u_1$</td>
<td>-1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>-1</td>
</tr>
<tr>
<td>$u_2$</td>
<td>0</td>
<td>0</td>
<td>-1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$u_3$</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>-1</td>
<td>1</td>
<td>-1</td>
</tr>
<tr>
<td>$u_4$</td>
<td>-1</td>
<td>1</td>
<td>-1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>$u_5$</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$u_6$</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>-1</td>
</tr>
<tr>
<td>$u_7$</td>
<td>1</td>
<td>1</td>
<td>-1</td>
<td>1</td>
<td>0</td>
<td>-2</td>
</tr>
<tr>
<td>$u_8$</td>
<td>1</td>
<td>0</td>
<td>-2</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$u_9$</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>$u_{10}$</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>
## RULES WITH PARAMETERS

<table>
<thead>
<tr>
<th>Antecedent</th>
<th>Consequent</th>
<th>Support</th>
<th>Strength</th>
<th>Coverage</th>
</tr>
</thead>
<tbody>
<tr>
<td>euro=0 and oil=-1</td>
<td>not dollar=1</td>
<td>1</td>
<td>0.1111</td>
<td>0.3333</td>
</tr>
<tr>
<td>euro=0 and oil=-1</td>
<td>not dollar=0</td>
<td>1</td>
<td>0.1111</td>
<td>0.2500</td>
</tr>
<tr>
<td>euro=1 and oil=1</td>
<td>not dollar=0</td>
<td>1</td>
<td>0.1111</td>
<td>0.2500</td>
</tr>
<tr>
<td>euro=1 and oil=1</td>
<td>not dollar=1</td>
<td>1</td>
<td>0.1111</td>
<td>0.2500</td>
</tr>
<tr>
<td>oil=-2</td>
<td>not dollar=0</td>
<td>1</td>
<td>0.1111</td>
<td>0.2500</td>
</tr>
<tr>
<td>oil=-2</td>
<td>not dollar=1</td>
<td>1</td>
<td>0.1111</td>
<td>0.2500</td>
</tr>
<tr>
<td>dollar=-1 and oil=1</td>
<td>not euro=1</td>
<td>1</td>
<td>0.1111</td>
<td>0.1667</td>
</tr>
<tr>
<td>dollar=0</td>
<td>not euro=0</td>
<td>2</td>
<td>0.2222</td>
<td>0.3333</td>
</tr>
<tr>
<td>dollar=-1 and oil=1</td>
<td>not euro=1</td>
<td>1</td>
<td>0.1111</td>
<td>0.3333</td>
</tr>
<tr>
<td>dollar=1 and oil=1</td>
<td>not euro=1</td>
<td>1</td>
<td>0.1111</td>
<td>0.3333</td>
</tr>
<tr>
<td>dollar=1 and oil=-1</td>
<td>not euro=0</td>
<td>1</td>
<td>0.1111</td>
<td>0.3333</td>
</tr>
<tr>
<td>oil=-2</td>
<td>not euro=0</td>
<td>1</td>
<td>0.1111</td>
<td>0.1667</td>
</tr>
<tr>
<td>dollar=-1 and euro=0</td>
<td>not oil=0</td>
<td>1</td>
<td>0.1111</td>
<td>0.3333</td>
</tr>
<tr>
<td>dollar=-1 and euro=0</td>
<td>not oil=-2</td>
<td>1</td>
<td>0.1111</td>
<td>0.3333</td>
</tr>
<tr>
<td>dollar=-1 and euro=0</td>
<td>not oil=-1</td>
<td>1</td>
<td>0.1111</td>
<td>0.3333</td>
</tr>
<tr>
<td>dollar=-1 and euro=1</td>
<td>not oil=0</td>
<td>1</td>
<td>0.1111</td>
<td>0.3333</td>
</tr>
<tr>
<td>dollar=-1 and euro=1</td>
<td>not oil=1</td>
<td>1</td>
<td>0.1111</td>
<td>0.3333</td>
</tr>
<tr>
<td>dollar=-1 and euro=1</td>
<td>not oil=-2</td>
<td>1</td>
<td>0.1111</td>
<td>0.3333</td>
</tr>
</tbody>
</table>
## INHIBITOR TRANSITION RULES

<table>
<thead>
<tr>
<th>Antecedent</th>
<th>Consequent</th>
<th>Support</th>
<th>Strength</th>
<th>Coverage</th>
</tr>
</thead>
<tbody>
<tr>
<td>dollar=-1</td>
<td>not dollar=1</td>
<td>2</td>
<td>0.2000</td>
<td>0.4000</td>
</tr>
<tr>
<td>dollar=-1</td>
<td>not dollar=-1</td>
<td>2</td>
<td>0.2000</td>
<td>0.4000</td>
</tr>
<tr>
<td>dollar=0 and oil=-1</td>
<td>not dollar=0</td>
<td>1</td>
<td>0.1000</td>
<td>0.2500</td>
</tr>
<tr>
<td>dollar=0 and oil=-1</td>
<td>not dollar=-1</td>
<td>1</td>
<td>0.1000</td>
<td>0.2500</td>
</tr>
<tr>
<td>euro=0 and oil=-1</td>
<td>not dollar=0</td>
<td>1</td>
<td>0.1000</td>
<td>0.2500</td>
</tr>
<tr>
<td>euro=0 and oil=-1</td>
<td>not dollar=-1</td>
<td>1</td>
<td>0.1000</td>
<td>0.2500</td>
</tr>
<tr>
<td>dollar=1 and oil=0</td>
<td>not dollar=1</td>
<td>1</td>
<td>0.1000</td>
<td>1.0000</td>
</tr>
<tr>
<td>dollar=1 and oil=0</td>
<td>not dollar=0</td>
<td>1</td>
<td>0.1000</td>
<td>1.0000</td>
</tr>
<tr>
<td>dollar=1 and euro=1</td>
<td>not dollar=0</td>
<td>2</td>
<td>0.2000</td>
<td>0.5000</td>
</tr>
<tr>
<td>dollar=1 and euro=1</td>
<td>not dollar=-1</td>
<td>2</td>
<td>0.2000</td>
<td>0.5000</td>
</tr>
<tr>
<td>dollar=1 and oil=1</td>
<td>not dollar=0</td>
<td>1</td>
<td>0.1000</td>
<td>0.2500</td>
</tr>
<tr>
<td>dollar=1 and oil=1</td>
<td>not dollar=-1</td>
<td>1</td>
<td>0.1000</td>
<td>0.2500</td>
</tr>
<tr>
<td>euro=1 and oil=1</td>
<td>not dollar=0</td>
<td>1</td>
<td>0.1000</td>
<td>0.2500</td>
</tr>
<tr>
<td>euro=1 and oil=1</td>
<td>not dollar=-1</td>
<td>1</td>
<td>0.1000</td>
<td>0.2500</td>
</tr>
<tr>
<td>dollar=1 and oil=-1</td>
<td>not dollar=0</td>
<td>1</td>
<td>0.1000</td>
<td>0.2500</td>
</tr>
<tr>
<td>dollar=1 and oil=-1</td>
<td>not dollar=-1</td>
<td>1</td>
<td>0.1000</td>
<td>0.2500</td>
</tr>
<tr>
<td>oil=-2</td>
<td>not dollar=1</td>
<td>1</td>
<td>0.1000</td>
<td>0.2000</td>
</tr>
<tr>
<td>oil=-2</td>
<td>not dollar=-1</td>
<td>1</td>
<td>0.1000</td>
<td>0.2000</td>
</tr>
<tr>
<td>dollar=0 and oil=0</td>
<td>not dollar=1</td>
<td>1</td>
<td>0.1000</td>
<td>0.2000</td>
</tr>
<tr>
<td>dollar=0 and oil=0</td>
<td>not dollar=-1</td>
<td>1</td>
<td>0.1000</td>
<td>0.2000</td>
</tr>
<tr>
<td>dollar=1 and oil=0</td>
<td>not dollar=-1</td>
<td>2</td>
<td>0.2000</td>
<td>0.2857</td>
</tr>
<tr>
<td>oil=-1</td>
<td>not euro=1</td>
<td>3</td>
<td>0.3000</td>
<td>0.4286</td>
</tr>
</tbody>
</table>
NET MODEL

\[
\text{not}((\text{ydollar} = \text{dollar}_0) \text{ andalso } (\text{yoil} = \text{oil}_2)) \\
\text{orelse } ((\text{ydollar} = \text{dollar}_1) \text{ andalso } (\text{yoil} = \text{oil}_2)) \\
\text{orelse } ((\text{yeuro} = \text{euro}_1) \text{ andalso } (\text{ydollar} = \text{dollar}_0)) \\
\text{orelse } ((\text{yeuro} = \text{euro}_1) \text{ andalso } (\text{yoil} = \text{oil}_0)) \\
\text{orelse } ((\text{yeuro} = \text{euro}_1) \text{ andalso } (\text{yoil} = \text{oil}_1)) \\
\text{orelse } ((\text{yeuro} = \text{euro}_1) \text{ andalso } (\text{yoil} = \text{oil}_2)) \\
\text{orelse } ((\text{ydollar} = \text{dollar}_1) \text{ andalso } (\text{yoil} = \text{oil}_1) \text{ andalso } (\text{yeuro} = \text{euro}_0)) \\
\text{orelse } ((\text{yeuro} = \text{euro}_1) \text{ andalso } (\text{yoil} = \text{oil}_1) \text{ andalso } (\text{ydollar} = \text{dollar}_1)) \\
\text{orelse } ((\text{yeuro} = \text{euro}_0) \text{ andalso } (\text{yoil} = \text{oil}_1) \text{ andalso } (\text{ydollar} = \text{dollar}_1)) \\
\text{orelse } ((\text{yeuro} = \text{euro}_0) \text{ andalso } (\text{yoil} = \text{oil}_1) \text{ andalso } (\text{ydollar} = \text{dollar}_1)) \\
\text{orelse } ((\text{yoil} = \text{oil}_0) \text{ andalso } (\text{yeuro} = \text{euro}_0) \text{ andalso } (\text{ydollar} = \text{dollar}_1))
\]

\begin{verbatim}
color dollar = with dollar_1 | dollar_0 | dollar_1; 
color euro = with euro_1 | euro_0; 
color oil = with oil_0 | oil_1 | oil_2 | oil_1;

var x dollar : dollar; var y dollar : dollar; 
var x euro : euro; var y euro : euro; 
var x oil : oil; var y oil : oil;
\end{verbatim}
# ANALYSIS RESULTS

New transitions between global states consistent with all transition rules extracted from a dynamic information system $DS$
In the today’s computer science development, the usefulness of proposed methods and algorithms for real-life data is conditioned by existing suitable computer tools automating computing processes. Therefore, in the lecture the ROSECON system is presented.

<table>
<thead>
<tr>
<th>Tool</th>
<th>Use</th>
</tr>
</thead>
<tbody>
<tr>
<td>ROSETTA</td>
<td>Data analysis (among others: computation of reducts, rules, classification, discretization)</td>
</tr>
<tr>
<td>RSES</td>
<td>Data analysis (among others: computation of reducts, rules, classification, construction of composed classifiers)</td>
</tr>
<tr>
<td>DESIGN/CPN</td>
<td>Design, analysis and simulation of coloured Petri net models</td>
</tr>
<tr>
<td>CPNetwork</td>
<td>Design and simulation of coloured Petri net models</td>
</tr>
<tr>
<td>ROSECON</td>
<td>Discovering concurrent models from data tables</td>
</tr>
</tbody>
</table>

http://rds.univ.rzeszow.pl (Software)
ROSECON

ROSECON is a computer tool supporting users in automatized discovering net models from data tables as well as predicting their changes in time.

ROSECON can be used to support automated:

- discovering concurrent models from experimental data,
- predicting model changes for temporal data,
- data analysis by using rough set methods
ROSECON ARCHITECTURE

MS Excel  
Matlab

Editing module

information system / dynamic information system

Statistics module

Discovering module

Rough set analysing module

Petri net analysing module

concurrent model

DESCON module

discernibility matrix, reducts, rules, components, coverings, extensions, rule coefficients

occurrence graph

Design/CPN  
CPN Tools
CONCLUDING REMARKS

• The presented methodology can be applied for automatic feature extraction. *The processes and connections between them can be interpreted as new features of the modelled system.*

• Constructed models can be useful for designers and analysts to:
  - determine some properties (concerning structures and behaviours) of modelled systems
  - extract new knowledge about systems
  - verify their descriptions or specifications
FURTHER WORKS

• to consider the prediction problem of property changing net models in non-stationary data systems

• to discover modular and hierarchical concurrent models

• discovery of dynamical models from data based on rough granular calculus of changes and interactions
A NEW SYNTHESIS PROBLEM

• Given the hierarchical information system A
• Define theory Th(A) of A (consisting of a set of rules describing temporal and spatial dependencies in A)
• Th(A) defines the maximal extension of A
• Construct a concurrent system consistent with Th(A)
More information about „Discovering Concurrent Models from Data” can be found in collective book, dedicated to Professor Zdzisław Pawlak
This book offers the most comprehensive coverage of key rough computing research, surveying a full range of topics from granular computing to systems theory.

In particular:

**Chapter XII: Rough Sets for Discovering Concurrent System Models from Data Tables**

*by Krzysztof Pancerz and Zbigniew Suraj*
More information about inhibitory rules can be found in our book
The series Studies in Computational Intelligence (SCI) publishes new developments and advances in the various areas of computational intelligence - quickly and with a high quality. The format is to cover the theory, applications, and design methods of computational intelligence, as embodied in the fields of engineering, computer science, physics and life science, as well as the methodologies behind them.

The series contains monographs, lecture notes, and edited volumes in computational intelligence spanning the areas of neural networks, connectionist systems, genetic algorithms, evolutionary computation, artificial intelligence, cellular automata, self-organizing systems, software engineering, parallel and distributed systems, and high-performance computing systems. Critical to both contributors and readers are the short publication time and world-wide distribution - this provides a rapid and broad dissemination of research results.

This monograph is devoted to a theoretical and experimental study of inhibitory decision and association rules. Inhibitory rules contain on the right-hand side a relation of the kind "attribute does not equal value". The use of inhibitory rules instead of deterministic (standard) rules allows us to describe more completely information encoded in decision or information systems and to design classifiers of high quality.

The most important feature of this monograph is that it includes an advanced mathematical analysis of problems on inhibitory rules. We consider algorithms for construction of inhibitory rules, based on minimal complexity of inhibitory rules, and algorithms for construction of the set of all minimal inhibitory rules. We also discuss results of experiments with standard and fuzzy classifiers based on inhibitory rules. These results show that inhibitory decision and association rules can be used in data mining and knowledge discovery both for knowledge representation and for prediction. Inhibitory rules can be also used under the analysis and design of concurrent systems.

The results obtained in the monograph can be useful for researchers in such areas as machine learning, data mining, and knowledge discovery, especially for those who are working in rough set theory, text theory, and logical analysis of data (LAD). The monograph can be used under the creation of courses for graduate students and for Ph. D. students.


Available online
springerlink.com

Springer
Thank you!