Discovery of Concurrent Data Models from Experimental Tables: A Rough Set Approach

Andrzej Skowron[1*] and **Zbigniew Suraj**[2*]

[1]Institute of Mathematics, Warsaw University
Banacha 2, 02-097 Warsaw, Poland

[2]Institute of Mathematics, Pedagogical University
Rejtana 16A, 35-310 Rzeszów, Poland
email: skowron, suraj@mimuw.edu.pl

Abstract

The main objective of machine discovery is the determination of relations between data and of data models. In the paper we describe a method for discovery of data models represented by concurrent systems from experimental tables. The basic step consists in a determination of rules which yield a decomposition of experimental data tables; the components are then used to define fragments of the global system corresponding to a table. The method has been applied for automatic data models discovery from experimental tables with Petri nets as models for concurrency.
Key words: data mining, system decomposition, rough sets, concurrent models

Introduction

The aim of this paper is to present an approach to the decomposition of information systems. Our approach can be applied to the discovery of data models in the form of concurrent systems.

Decomposition of large experimental data tables can be treated as one of the fundamental tools in data mining. It is usually imposed by the high computational complexity of the search for relations between data on one hand and/or the structure of the process of data models discovery on the other (see e.g. (Żytkow 1991), (Piatetsky-Shapiro & Frawley 1991), (Shrager & Langley 1990), (Kodratoff & Michalski 1990)).

Our approach is based on rough set theory (Pawlak 1991) and boolean reasoning (Brown 1990). It consists of three levels. First we show how experimental data tables are represented by information systems (Pawlak 1991). Next we discuss how any information system S can be decomposed (with respect to any of its reduct) into components linked by some connections which allow to preserve some constraints. Any component represents in a sense the strongest functional module of the system. The connections between components represent constraints which must be satisfied when these functional modules coexist in the system. The components together with the connections define a so called covering of S. Finally, we use the coverings of the information system S to construct its concurrent model in the form of a marked Petri net (N_S, M_S) (Skowron & Suraj 1994) with the following property: the reachability set $R(N_S, M_S)$ is in one-to-one correspondence with the set of all global states consistent with all decision rules valid in S (and having examples in S).

In the paper we investigate the decomposition problems which can now be roughly defined as follows:

Component Extracting Problem:
Input: An information system S.
Output: All components of S.

Covering Problem:
Input: An information system S.
Output: The set of all coverings of S.

Our approach can be applied for automatic feature extraction and for control design of systems represented by experimental tables.

Information Systems

In the paper we represent experimental data by information systems (Pawlak 1991).

An *information system* is a pair $S = (U, A)$, where U is a nonempty finite set of objects, called the *universe*, A is a nonempty finite set of *attributes*, i.e. $a : U \to V_a$ for $a \in A$, where V_a is called the *value set* of a.

The set $V = \bigcup_{a\in A} V_a$ is said to be the *domain* of A.

If $S = (U, A)$, then the system $S' = (U', A')$ such that $U \subseteq U'$, $A' = \{a' : a \in A\}$, $a'(u) = a(u)$ for $u \in U$ and $V_a = V_{a'}$ for $a \in A$ will be called a *U'-extension* of S (or an extension of S, in short). S is also called a *restriction* of S'. If $S = (U, A)$ then $S' = (U, B)$ such that $A \subseteq B$ will be referred to as a *B-extension* of S.

EXAMPLE 1 (Pawlak 1992). Let us consider an information system $S = (U, A)$ such that $U = \{u_1, u_2, u_3, u_4, u_5\}$, $A = \{a, b, c, d, e\}$ and the values of the attributes are defined as in Table 1. □

*This work has been supported by the grant #8 S503 019 06 from the State Committee for Scientific Research in Poland

$U\backslash A$	a	b	c	d	e
u_1	1	0	2	1	0
u_2	0	0	1	2	1
u_3	2	0	2	1	0
u_4	0	0	2	2	2
u_5	1	1	2	1	0

Table 1. An example of an information system

Let $S = (U, A)$ be an information system. With any subset of attributes $B \subseteq A$ we associate a binary relation $ind(B)$, called *indiscernibility relation*, which is defined by: $ind(B) = \{(u, u') \in U \times U : \text{for every } a \in B, a(u) = a(u')\}$.

Any information system $S = (U, A)$ determines an *information function*, $Inf_A : U \to P(A \times V)$ defined by $Inf_A(u) = \{(a, a(u)) : a \in A\}$, where $V = \bigcup_{a\in A} V_a$ and $P(X)$ denotes the powerset of X. The set $\{Inf_A(u) : u \in U\}$ will be denoted by $INF(S)$.

Hence, $u\, ind(A) u'$ iff $Inf_A(u) = Inf_A(u')$.

The values of an information function will be sometimes represented by vectors of the form $(v_1, \ldots, v_m)$ with $v_i \in V_{a_i}$ for $i = 1, \ldots, m$, where $m = |A|$. Such vectors are called *information vectors* (over V and A).

Let $S = (U, A)$ be an information system, where $A = \{a_1, ..., a_m\}$. Pairs (a, v) with $a \in A$, $v \in V$ are called *descriptors*. Instead of (a, v) we also write $a = v$ or a_v.

The set of *terms* over A and V is the least set containing descriptors (over A and V) and closed with respect to the classical propositional connectives: $\neg, \vee, \wedge$. The meaning $\| \tau \|_S$ (or in short $\| \tau \|$) of a term τ in S is defined inductively as follows:

$$\| (a, v) \| = \{u \in U : a(u) = v\} \text{ for } a \in A \text{ and } v \in V_a;$$
$$\| \tau \vee \tau' \| = \| \tau \| \cup \| \tau' \|; \| \tau \wedge \tau' \| = \| \tau \| \cap \| \tau' \|;$$
$$\| \neg\tau \| = U - \| \tau \| .$$

Decision Rules

Decision rules express some of the relationships between attributes of information systems.

Let $S = (U, A)$ be an information system and let $B, C \subseteq A$. We say that the *set C depends* on B in S in *degree* $k (0 \leq k \leq 1)$, symbolically $B \underset{S,k}{\to} C$, if $k = \frac{card(POS_B(C))}{card(U)}$, where $POS_B(C)$ is the *B-positive region of* C in S (Pawlak 1991).

If $k = 1$ we write $B \underset{S}{\to} C$ instead of $B \underset{S,k}{\to} C$. In this case $B \underset{S}{\to} C$ means that $ind(B) \subseteq ind(C)$. If the right hand side of a dependency consists of one attribute only, we say the dependency is *elementary*.

A decision rule over A and V is any expression of the following form: (1) $a_{i_1} = v_{i_1} \wedge \ldots \wedge a_{i_r} = v_{i_r} \Rightarrow a_p = v_p$, where $a_p, a_{i_j} \in A$, v_p, $v_{i_j} \in V_{a_{i_j}}$ for $1 \leq j \leq r$.

A decision rule of the form (1) is called *trivial* if $a_p = v_p$ also appears on the left hand side of the rule. It is *true* in S if $\| a_{i_1} = v_{i_1} \wedge \ldots \wedge a_{i_r} = v_{i_r} \| \subseteq \| a_p = v_p \|$, which is denoted by (2) $a_{i_1} = v_{i_1} \wedge \ldots \wedge a_{i_r} = v_{i_r} \underset{S}{\Rightarrow} a_p = v_p$. A rule of type (2) has an example in S if $\| a_{i_1} = v_{i_1} \wedge \ldots \wedge a_{i_r} = v_{i_r} \|_S \neq \emptyset$. By $D(S)$ we denote the set of all true decision rules which have examples in S.

Let $R \subseteq D(S)$. An information vector $\boldsymbol{v} = (\boldsymbol{v}_1, \ldots, \boldsymbol{v}_m)$ is *consistent* with R iff for any decision rule $a_{i_1} = v_{i_1} \wedge \ldots \wedge a_{i_r} = v_{i_r} \Rightarrow a_p = v_p$ in R if $\boldsymbol{v}_{i_j} = v_{i_j}$ for $j = 1, \ldots, r$ then $v_p = \boldsymbol{v}_p$. The set of all information vectors consistent with R is denoted by $CON(R)$.

Let $S' = (U', A')$ be a U'-extension of $S = (U, A)$. We say that S' is a *consistent extension* of S iff $D(S) \subseteq D(S')$. S' is a *maximal* consistent extension of S iff S' is a consistent extension of S and any consistent extension S'' of S is a restriction of S'.

Reduction of Attributes

Let $S = (U, A)$ be an information system. A subset $B \subseteq A$ is a *reduct* of S (Pawlak 1991) iff $ind(B) = ind(A)$ and $ind(B') \neq ind(A)$ for any $B' \subset B$. The set of all reducts in S is denoted by $RED(S)$.

Let $S = (U, A)$ be an information system and let us assume that $U = \{u_1, \ldots, u_n\}$, and $A = \{a_1, \ldots, a_m\}$. By $M(S)$ we denote an $n \times n$ matrix (c_{ij}), called the *discernibility matrix* of S, such that $c_{ij} = \{a \in A : a(u_i) \neq a(u_j)\}$ for $i, j = 1, \ldots, n$ (Skowron & Rauszer 1992).

With any discernibility matrix $M(S)$ we can associate a *discernibility function* $f_{M(S)}$. The *discernibility function* $f_{M(S)}$ for an information system S is a boolean function of m propositional variables $\boldsymbol{a}_1, \ldots, \boldsymbol{a}_m$ (where $a_i \in A$ for $i = 1, \ldots, m$) defined as the conjunction of all expressions $\bigvee c_{ij}$, where $\bigvee c_{ij}$ is the disjunction of all the elements of $\boldsymbol{c}_{ij} = \{\boldsymbol{a} : a \in c_{ij}\}$, where $1 \leq j < i \leq n$ and $c_{ij} \neq \emptyset$. In the sequel we write a instead of $\boldsymbol{a}$ when no confusion can arise.

PROPOSITION 1 (Skowron & Rauszer 1992). *Let $S = (U, A)$ be an information system and let $f_{M(S)}$ be a discernibility function for S. Then all prime implicants (Wegner 1987) of the function $f_{M(S)}$ correspond to all reducts of S.*

In order to construct a concurrent model for a given information system all reducts of the system have to be computed first (Skowron & Rauszer 1992).

PROCEDURE for computing $RED(S)$:

Step 1. Compute the discernibility matrix for the system S.

Step 2. Compute the discernibility function $f_{M(S)}$ associated with the discernibility matrix $M(S)$.

Step 3. Compute the minimal disjunctive normal form of the discernibility function $f_{M(S)}$. (The normal form of the function yields all reducts).

EXAMPLE 2. Applying the above procedure to the system in Example 1, we obtain two reducts: $R_1 = \{a, b, c\}$ and $R_2 = \{a, b, e\}$ of the system. □

PROPOSITION 2 (Pawlak 1992). *Let $S = (U, A)$ be an information system, $R \in RED(S)$, and $R \subset A$. Let $f_{M(S')}$ be a relative discernibility function for the system $S' = (U, R \cup \{a^*\})$, where $a^* \in A - R$. Then all prime implicants of the function $f_{M(S')}$ correspond to all $\{a^*\}$ - reducts of S'.*

The next example illustrates how to find all dependencies among attributes.

EXAMPLE 3. Let us consider again the system S from Example 1. We have $\{a, b, c\} \underset{S}{\rightarrow} \{d, e\}$ and $\{a, b, e\} \underset{S}{\rightarrow} \{c, d\}$, so $\{a, b, c\} \underset{S}{\rightarrow} \{d\}$, $\{a, b, c\} \underset{S}{\rightarrow} \{e\}$, $\{a, b, e\} \underset{S}{\rightarrow} \{c\}$, $\{a, b, e\} \underset{S}{\rightarrow} \{d\}$. The set of rules corresponding to all nontrivial dependencies within the reduct R_2 has the form: $a_1 \vee a_2 \vee b_1 \underset{S}{\Rightarrow} e_0, a_0 \vee a_2 \vee e_1 \vee e_2 \underset{S}{\Rightarrow} b_0, e_1 \vee e_2 \underset{S}{\Rightarrow} a_0$, $b_1 \underset{S}{\Rightarrow} a_1$, while the set of decision rules corresponding to all nontrivial dependencies of R_2 with c, d has the following form: $a_1 \vee a_2 \vee b_1 \vee e_0 \vee e_2 \underset{S}{\Rightarrow} c_2$, $e_1 \underset{S}{\Rightarrow} c_1$, $a_1 \vee a_2 \vee b_1 \vee e_0 \underset{S}{\Rightarrow} d_1$, $a_0 \vee e_1 \vee e_2 \underset{S}{\Rightarrow} d_2$. These rules can be generated by applying the method presented in (Skowron 1993). □

Decomposition of Information Systems

This section introduces concepts and notation related to the decomposition of information systems as well as a method for constructing components and coverings of a given information system with respect to its reducts.

Let $S = (U, A)$ be an information system. An information system S is said to be *covered with constraints C (or C-covered, in short) by information systems* $S_1 = (U_1, A_1), \ldots, S_k = (U_k, A_k)$ if $INF(S') = \{Inf_{A_1}(u_1) \cup \ldots \cup Inf_{A_k}(u_k) \;:\; Inf_{A_1}(u_1) \cup \ldots \cup Inf_{A_k}(u_k) \in CON(C)$ and $u_i \in U_i$ for $i = 1, \ldots, k\}$, where S' is a maximal consistent extension of S and C is a set of decision rules. The pair $(\{S_1, \ldots, S_k\}, C)$ is called a *C-covering* of S (or a covering of S, in short). The sets $S_1, \ldots, S_k$ are its *components* and C is the set of *constraints* (connections).

EXAMPLE 4. Let us consider the information system S from Example 1. It is easy to see that the information systems $S_1 = (U_1, A_1)$, $S_2 = (U_2, A_2)$ and $S_3 = (U_3, A_3)$ represented by Tables 2, 3 and 4, respectively, and the set of constraints C containing decision rules $a_1 \vee a_2 \underset{S}{\Rightarrow} e_0$, $e_1 \vee e_2 \underset{S}{\Rightarrow} a_0$, $a_0 \vee a_2 \underset{S}{\Rightarrow} b_0$, $b_1 \underset{S}{\Rightarrow} a_1$, $b_1 \underset{S}{\Rightarrow} e_0$, and $e_1 \vee e_2 \underset{S}{\Rightarrow} b_0$ yield a C-covering of S.

$U_1\backslash A_1$	a	d
u_1	1	1
u_2	0	2
u_3	2	1

Table 2. The information system S_1

$U_2\backslash A_2$	c	d	e
u_1	2	1	0
u_2	1	2	1
u_4	2	2	2

Table 3. The information system S_2

$U_3\backslash A_3$	b
u_1	0
u_5	1

Table 4. The information system S_3

□

PROPOSITION 3. *Every information system has at least one covering.*

If $S = (U, A)$, then the system $S = (U', A')$ such that $U' \subseteq U$, $A' = \{a' : a \in B \subseteq A\}$, $a'(u) = a(u)$ for $u \in U'$ and $V_{a'} = V_a$ for $a \in A$ is said to be a *subsystem* of S.

EXAMPLE 5. Every information system in Example 4 is a subsystem of the system S from Example 1. □

Let $S = (U, A)$ be an information system and let $R \in RED(S)$. An information system $S' = (U', A')$ is a *normal component* of S (with respect to R) iff the following conditions are satisfied: (i) S' is a subsystem of S, (ii) $A' = B \cup C$, where B is a minimal (with respect to $\subseteq$) subset of R such that $B \underset{S}{\rightarrow} \{a\}$ for some $a \in A - R$ and C is the set of all attributes a with the above property.

The set of all normal components of S (with respect to R) is denoted by $COMP_R(S)$.

EXAMPLE 6. The subsystems S_1, S_2, S_3 are normal components of S from Example 1 (with respect to the reduct R_2). A more detailed explanation of this fact is included in Example 7. □

PROPOSITION 4. *Every information system has at least one normal component (with respect to any of its reduct).*

Let $S' \in COMP_R(S)$ and $S' = (U', B_{S'} \cup C_{S'})$. By X_R we denote the set of all attributes which simultaneously occur in normal components of S (with respect to R) and in the reduct R, i.e. $X_R = \bigcup_{S' \in COMP_R(S)} B_{S'}$.

Let X_R be a set defined for S and R as above. We say that a subsystem $S' = (U', A')$ of S is a *degenerated component* of S (with respect to R) iff $A' = \{a\}$ for some $a \in R - X_R$. We denote this fact by $\{a\} \underset{S}{\rightarrow} \emptyset$ (the empty set).

In the sequel a component (with respect to a reduct) will be assumed to be either a normal component or a degenerated component (with respect to the reduct).

PROPOSITION 5. *Let $S = (U, A)$ be an information system and let R be its reduct. Then the information system S consists of $|R - X_R|$ degenerated components (with respect to R).*

Let $S = (U, A)$ be an information system, $R \in RED(S)$.

We say that S is *R-decomposable into components* or that S is *C-coverable by components* (with respect to R) iff there exist components $S_1 = (U_1, B_1 \cup C_1), \ldots$, $S_k = (U_k, B_k \cup C_k)$ of S (with respect to R) with a set of constraints C such that $B_1 \cup \ldots \cup B_k = R$ and $C_1 \cup \ldots \cup C_k = A - R$, yielding a C-covering of S.

The set of constraints (connections) C includes: (i) decision rules corresponding to nontrivial dependencies between attributes in B_i $(i = 1, ..., k)$ called *internal linkings* (*the internal connections*) within the component S_i of S, (ii) decision rules corresponding to nontrivial dependencies between attributes in $B_i (i = 1, ..., k)$ and those in the set $A - A_i$, where $A_i = B_i \cup C_i$ called *external linkings* (*the external connections*) with the outside of S_i.

THEOREM 1. *Every information system is C-coverable by components (with respect to any its reduct), where C is the set of all internal and external linkings of S.*

We obtained a constructive method of information systems (data tables) decomposition into functional modules interconnected by external linkings. One can observe a similarity of our data models to those used in general system theory and control design.

PROPOSITION 6. *Let R be a reduct of an information system S. Then S has at least one C-covering by components (with respect to R), where C is the set of all internal and external linkings of S.*

We denote by $COVER_R(S)$ the family of all C-coverings of S (with respect to R), where C is the set of all internal and external linkings of S.

Procedures for Computing Components and Coverings

All normal components of a given information system $S = (U, A)$ (with respect to a reduct $R \in RED(S)$) can be obtained by the following procedure:

PROCEDURE for computing $COMP_R(S)$:

Input: An information system $S = (U, A)$,
a reduct $R \in RED(S)$.
Output: Components of S (with respect to R), i.e.
the set $COMP_R(S)$.

Step 1. Compute all dependencies of the form: $R \underset{S}{\rightarrow} \{a\}$, for any $a \in A - R$.

Step 2. Compute the discernibility function $f_{M(S')}$ for each subsystem $S' = (U, R \cup \{a\})$ with $a \in A - R$. In this step we compute the so called $\{a\}$ - reducts of R, for $a \in A - R$ (Pawlak 1991).

Step 3. For all dependencies of the form $B \underset{S}{\rightarrow} \{a_{i_1}\}$, ..., $B \underset{S}{\rightarrow} \{a_{i_k}\}$, where B is any subset of R obtained in Step 2, construct a dependency $B \underset{S}{\rightarrow} C$, where $C = \{a_{i_1}\} \cup \ldots \cup \{a_{i_k}\}$. The set C is the maximal subset in $A - R$ such that the dependency $B \underset{S}{\rightarrow} C$ is true. Now the subsystem $S'' = (U', B \cup C)$ of S defines a normal component of S (with respect to R).

The complexity of the computation of $RED(S)$ is high. Nevertheless, the existing procedures and heuristics are sufficient for the computation of reducts in many applications (see e.g. (Bazan, Skowron & Synak 1994)).

EXAMPLE 7. Let us perform the procedure for the computation $COMP_{R_2}(S)$ for the information system S of Example 1 and its reduct R_2.

Step 1. The following elementary dependencies are valid in the system S for the reduct $R_2 : \{a, b, e\} \underset{S}{\rightarrow} \{c\}$, $\{a, b, e\} \underset{S}{\rightarrow} \{d\}$.

Step 2. We compute the minimal subsets of R_2 on which the sets $\{c\}$ and $\{d\}$ depend, i.e. we compute the relative reducts (cf. (Pawlak 1992)) of the left hand sides of the above dependencies. To reduce the first elementary dependency we consider the information system $S_1 = (U, B \cup \{c\})$ with $B = \{a, b, e\}$. Hence, $f_{M(S_1)}(a, b, e) = e$. Thus $\{a, b, e\} \underset{S}{\rightarrow} \{c\}$ can be simplified to $\{e\} \underset{S}{\rightarrow} \{c\}$.

We reduce the second dependency in a similar way. As a consequence, $\{a, b, e\} \underset{S}{\rightarrow} \{d\}$ can be reduced either to $\{a\} \underset{S}{\rightarrow} \{d\}$ or $\{e\} \underset{S}{\rightarrow} \{d\}$. Eventually, we get the following minimal dependencies (i.e. dependencies with a minimal number of attributes on the left hand side) with respect to R_2 in the information system $S : \{a\} \underset{S}{\rightarrow} \{d\}$, $\{e\} \underset{S}{\rightarrow} \{c\}$, $\{e\} \underset{S}{\rightarrow} \{d\}$. This completes Step 2 of the above procedure.

Step 3. For dependencies $\{e\} \underset{S}{\rightarrow} \{c\}$ and $\{e\} \underset{S}{\rightarrow} \{d\}$ we construct a new dependency $\{e\} \underset{S}{\rightarrow} \{c, d\}$. Now we have $\{a\} \underset{S}{\rightarrow} \{d\}$ and $\{e\} \underset{S}{\rightarrow} \{c, d\}$. They define two normal components $S_1 = (U_1, A_1)$ and $S_2 = (U_2, A_2)$ of the system S, where: $A_1 = B_1 \cup C_1$, $B_1 = \{a\}$, $C_1 = \{d\}$, $A_2 = B_2 \cup C_2$, $B_2 = \{e\}$, $C_2 = \{c, d\}$. Since $X_{R_2} = \{a, e\}$, we have $R_2 - X_{R_2} = \{b\}$. This means that $\{b\} \underset{S}{\rightarrow} \emptyset$ is true in S. Hence S has the degenerated component $S_3 = (U_3, A_3)$ of the form: $A_3 = B_3 \cup C_3$, $B_3 = \{b\}$, $C_3 = \emptyset$.

Eventually, S is decomposed into three components (with respect to R_2). They are shown in Tables 2, 3 and 4, respectively. □

In a similar way we can compute the components of S with respect to the reduct R_1.

To compute a covering of an information system by its components (with respect to a reduct) it is sufficient to perform the following procedure.

PROCEDURE for computing $COVER_R(S)$:

Input: An information system $S = (U, A)$,
a reduct $R \in RED(S)$.
Output: The covering family of S,
i.e. $COVER_R(S)$.

Step 1. Compute all normal and degenerated components of S (with respect to R).

Step 2. Compute the set C of all external and internal linkings of S.

Step 3. Choose those combinations of components which together with C yield a C-covering by components of S (with respect to R). This step is to be performed as long as new solutions are obtained.

EXAMPLE 8. The information system S of Example 1 has one covering $(\{S_1, S_2, S_3\}, C)$ (with respect to the reduct R_2), where S_1, S_2, S_3 denote components of S (with respect to R_2) computed in Example 7 and C is the set of all internal and external linkings of S.

There are no internal linkings in the components of the system (with respect to R_2), since each component of S contains only one attribute from the reduct R_2. However, the components are connected by external linkings of the form: for S_1 and S_2 : $a_1 \vee a_2 \underset{S}{\Rightarrow} c_2$, $a_1 \vee a_2 \underset{S}{\Rightarrow} e_0$, $e_1 \vee e_2 \underset{S}{\Rightarrow} a_0$; for S_1 and S_3 : $a_0 \vee a_2 \underset{S}{\Rightarrow} b_0$, $b_1 \underset{S}{\Rightarrow} a_1$, $b_1 \underset{S}{\Rightarrow} d_1$; for S_2 and S_3 : $e_1 \vee e_2 \underset{S}{\Rightarrow} b_0$, $b_1 \underset{S}{\Rightarrow} e_0$, $b_1 \underset{S}{\Rightarrow} c_2$, $b_1 \underset{S}{\Rightarrow} d_1$.

How to Compute Concurrent Data Models from Information Systems?

We present a method for constructing a marked Petri net (N_S, M_S) for an arbitrary information system S such that the reachability set $R(N_S, M_S)$ represents the set of all global states consistent with a given information system S.

That method consists of two steps. First, decision rules corresponding to two kinds of dependencies are generated. The first kind consists of the partial dependencies between attributes within reducts, the second - of the dependencies between attributes not in reducts and those within reducts. Next, the decision rules so obtained (and represented in an optimal form with respect to the number of descriptors on the left hand side) (Skowron 1993) are implemented by means of a Petri net (Murata 1989).

First initial transformations of decision rules are performed. There are two rules (see Figure 1).

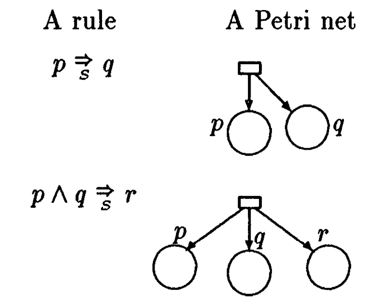
A rule
A Petri net
$p \underset{S}{\Rightarrow} q$
p
q
$p \wedge q \underset{S}{\Rightarrow} r$
p
q
r

Figure 1. Initial transformation rules, where p, q, r are descriptors in S

Next a method for transforming decision rules representing a given information system into a Petri net is executed. This method consists of three levels:

1. A net representing all processes (attributes) in a reduct of an information system is constructed.
2. The net obtained in the first step is extended by adding the elements (arcs and transitions) of the net induced by the decision rules determined by: (i) all nontrivial dependencies between processes of the information system; (ii) partial dependencies between processes within a reduct of the information system.
3. We add to the net obtained so far the subnets corresponding to situations when between some internal states of processes (but not all states) there are no dependencies represented by the information system.

This method is repeated for all reducts of the given information system. Finally, the obtained nets are merged.

Such an approach makes the appropriate construction of a net much more readable. In the example which follows we only illustrate step 2(i) of the above transformation method. This example uses the reduct R_2 of the information system S of Example 1.

EXAMPLE 9. Consider the decision rules from Example 3 for the system of Example 1: $a_0 \underset{S}{\Rightarrow} d_2$, $a_1 \vee a_2 \underset{S}{\Rightarrow} d_1$, $e_0 \underset{S}{\Rightarrow} d_1$, $e_1 \vee e_2 \underset{S}{\Rightarrow} d_2$, $e_0 \vee e_2 \underset{S}{\Rightarrow} c_2$, $e_1 \underset{S}{\Rightarrow} c_1$.

A net representation of the above decision rules obtained by an application of our construction (after some simplifications consisting in the deletion of superfluous arcs) is illustrated in Figure 2. The initial marking of the nets presented in the figure corresponds to the second row of Table 1. □

Remark. In Figure 2 places d_1 and d_2 are drawn separately for readibility reasons.

The construction method shortly described above has the following properties (Skowron & Suraj 1994):

THEOREM 2. *Let S be an information system and let (N_S, M_S) be a marked Petri net representing a system S. Then $INF(N_S, M_S) = CON(D(S))$, where $INF(N_S, M_S)$ denotes the set $\{v(M) : M \in R(N_S, M_S)\}$ and $R(N_S, M_S)$ is the reachability set of N_S from M_S.*

THEOREM 3. *Let S be an information system, S' its U'-extension constructed as above. Then S' is the largest consistent extension of S.*

COROLLARY 1. *Let S' be a maximal consistent extension of an information system S constructed by the method presented in the paper. Then $COVER_R(S) = COVER_R(S')$, for any reduct R of S.*

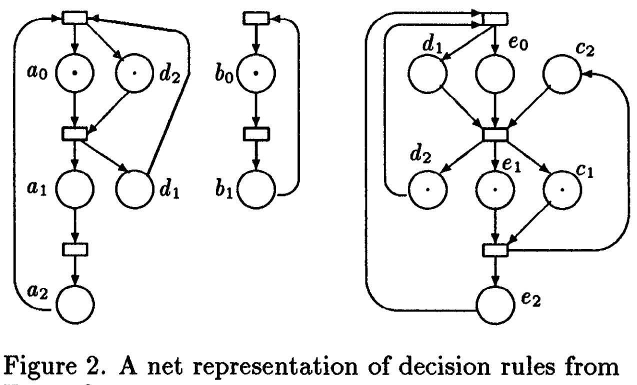

Figure 2. A net representation of decision rules from Example 9

In Figure 3 we show nets representing the external linkings between components S_1, S_3. These nets also include arcs which guarantee the correctness of the construction. □

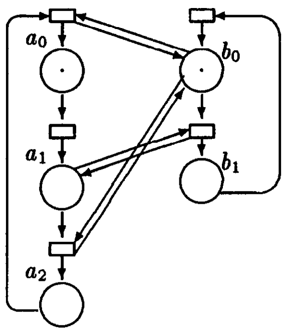

Figure 3. External linkings between components S_1 and S_3

In the Petri net (N_S, M_S) constructed for a given information system one can distinguish components and connections between them. The Petri net (N_S, M_S) can be decomposed with respect to any C-covering in $COVER_R(S)$, where R is any reduct of S and C is the set of all internal and external linkings of S.

Remark. A net representation of components S_1, S_3, and S_2 of the system S of Example 1 is shown in Figure 2.

Conclusions

The decomposition method has been implemented in $C++$ and preliminary tests are promising.

Our method can be applied for automatic feature extraction. The properties of the constructed concurrent systems (e.g. their invariants) can be interpreted as higher level laws of experimental data. New features can be also obtained by performing for a given decision table $S = (U, A \cup \{d\})$ the following steps:

Step 1. Extract from S a subtable S_i corresponding to the decision i, for any $i \in V_d$, i.e. $S_i = (U_i, A_i)$, where $U_i = \{u \in U : d(u) = i\}$, $A_i = \{a^i : a \in A\}$, and $a^i(u) = a(u)$ for $u \in U_i$.

Step 2. Compute the components of S_i for any $i \in V_d$.

Step 3. For a new object u compute the values of components defined on information included in $Inf_A(u)$ and check in what cases the computed values of components are matching $Inf_A(u)$ (i.e. they are included in $Inf_A(u)$). For any $i \in V_d$ compute the ratio $n_i(u)$ of the number of components matching $Inf_A(u)$ to all components of S_i. The simplest strategy classifies u to the decision class i_0, where $n_{i_0}(u) = \max_i n_i(u)$.

We also study some applications of our method in control design from experimental data tables.

References

Bazan, J.; Skowron, A.; and Synak, P. 1994. Dynamic Reducts as a Tool for Extracting Laws from Decision Tables. *Lecture Notes in Artificial Intelligence* 869:346-355, Springer.

Brown E.M. 1990. *Boolean Reasoning.* Kluwer, Dordrecht.

Kodratoff, Y., and Michalski, R. eds. 1990. *Machine Learning* Vol. 3. Morgan Kaufmann, San Mateo, CA.

Murata, T. 1989. Petri Nets: Properties, Analysis and Applications. In Proceedings of the IEEE 77(4):541-580.

Pawlak, Z. 1991. *Rough Sets - Theoretical Aspects of Reasoning about Data.* Kluwer, Dordrecht.

Pawlak, Z. 1992. Concurrent Versus Sequential the Rough Sets Perspective. *Bull. EATCS* 48:178-190.

Piatetsky-Shapiro, G., and Frawley, W. eds. 1991. *Knowledge Discovery in Databases.* The AAAI Press, Menlo Park, CA.

Shrager, J., and Langley, P. eds. 1990. *Computational Models of Scientific Discovery and Theory Formation.* Morgan Kaufmann, San Mateo, CA.

Skowron, A. 1993. Boolean Reasoning for Decision Rules Generation. *Lecture Notes in Artificial Intelligence* 689:295-305, Springer.

Skowron, A.; and Rauszer, C. 1992. The Discernibility Matrices and Functions in Information Systems. In R. Słowiński ed., *Intelligent Decision Support: Handbook of Applications and Advances of the Rough Sets Theory,* Kluwer, Dordrecht, 331-362.

Skowron, A.; and Suraj, Z. 1994. Synthesis of Concurrent Systems Specified by Information Systems, Inst. of Computer Science Rep. 39/93, Warsaw Univ. of Technology.

Wegner, I. 1987. *The Complexity of Boolean Functions.* Wiley and B.G. Teubner, Stuttgart.

Ziarko, W.; and Shan, N. 1993. An Incremental Learning Algorithm for Constructing Decision Rules. In Proceedings of the International Workshop on Rough Sets, Fuzzy Sets and Knowledge Discovery, 335-346. Canada.

Żytkow, J. 1991. Interactive Mining of Regularities in Databases. In G. Piatetsky-Shapiro, and Frawley, W. eds., *Knowledge Discovery in Databases.* The AAAI Press, Menlo Park.

Żytkow, J.; and Zembowicz, R. 1993. Database Exploration in Search of Regularities. *Journal of Intelligent Information Systems* 2:39-81, Kluwer, Boston.