A New Method for Determining of Extensions and Restrictions of Information Systems

Wojciech Rząsa[1] and Zbigniew Suraj[2]

[1] Institute of Mathematics,
University of Rzeszow,
Rejtana 16A, 35-310 Rzeszow, Poland
wrzasa@univ.rzeszow.pl
[2] Chair of Computer Science Foundations,
University of Information Technology and Management,
H. Sucharskiego 2, 35-225 Rzeszow, Poland
zsuraj@wenus.wsiz.rzeszow.pl

Abstract. The aim of the paper is presentation of some results that concern consistent extensions and consistent restrictions of information systems and decision systems. It is possible to determine them on the base of knowledge contained in rules generated from a given information (decision) system. The paper presents a way of consistent extensions and consistent restrictions determining, different from the one mentioned above. The paper contains also the definitions of notions of strict consistent extension (restriction) system and a necessary and sufficient condition for the existence of them for information (decision) system.

1 Introduction

Suraj in [2], [3] presents problems from domain of concurrency which may be formulated by rough set theory notions and may be solved finding consistent extensions of a given information or decision system. The problems are among others: discovering concurrent data models from experimental tables, re-engineering problem for cooperative information systems, the real time decision making problem and the control design problem for discrete event system. This paper presents further results that concern consistent extensions (restrictions). In particular in section 2 there are proofs of two propositions inserted in [3] and a corollary of the propositions. There are also algorithms of maximal consistent extension and minimal consistent restriction finding for information and decision systems, different from the algorithms proposed in [3]. In section 3 notions of strict consistent extension and restriction are considered. Although many notions of the rough set theory is used in the paper, most of them are fundamental or very common so their definitions are passed over. If the need arises the reader is referred to [1], [2], [3]. However, we define a special kind of reduct, namely the reduct related to fixed object and attribute, that is crucial in some places of the paper. This notion may be occur in two forms. For information system $S = (U, A)$ and fixed object $u \in U$ and attribute $a \in A$, a class of minimal sets of attributes

J.J. Alpigini et al. (Eds.): RSCTC 2002, LNAI 2475, pp. 197–204, 2002.

$X \subseteq A - \{a\}$ (with respect to inclusion) such that $\forall u' \in U - \{u\}[(u', u) \notin IND(\{a\}) \wedge (u', u) \notin IND(A - \{a\}) \Rightarrow (u', u) \notin IND(X)]$ is denoted in the paper as $u, \sim a, U - RR$ in contradistinction to a class of reducts $u, a, U - RR$, which is a class of minimal sets of attributes $Y \subseteq A - \{a\}$ such that $\forall u' \in U - \{u\}[(u', u) \in IND(\{a\}) \wedge (u', u) \notin IND(A - \{a\}) \Rightarrow (u', u) \notin IND(Y)]$. A set of all rules true in S [2] is denoted in the paper by $RUL(S)$ whilst set of minimal rules that are the rules true in S with minimal left hand sides [2] is denoted by $OPT(S)$. It may be determined as described in [2].

2 Consistent Extensions and Restrictions

2.1 Consistent Extensions of Information Systems

Let $S = (U, A)$ and $S' = (U', A')$ be information systems. The information system S' is called a consistent extension of system S if and only if the following conditions are satisfied:

1. $U \subseteq U'$;
2. $card(A) = card(A')$;
3. $\forall a \in A \exists a' \in A'[V_{a'} = V_a \wedge a'(u) = a(u)$ for all $u \in U]$;
4. $RUL(S) \subseteq RUL(S')$.

Below, sets A and A' are marked by the same letter A, so consequently $S' = (U', A)$ is written instead of $S' = (U', A')$. If $U' \neq U$ then S' is called a non-trivial consistent extension of system S. Set $EXT(S)$ of all consistent extensions of a given information system S may be ordered by relation '$\leqslant$', defined as follows: $\forall S'_1 = (U'_1, A), S'_2 = (U'_2, A) \in EXT(S)[S'_1 \leqslant S'_2$ if and only if $U'_1 \subseteq U'_2]$. Maximal elements in the set $EXT(S)$ ordered by $\leqslant$ are called maximal consistent extensions of system S. If S' is a consistent extension of S then S is called a consistent restriction of S'. A set of all consistent restrictions of a given information system S is denoted by $RES(S)$. If we order set $RES(S)$ in the same way as set $EXT(S)$ was ordered, then every minimal element of such ordered set $RES(S)$ is called a minimal consistent restriction of system S.

Proposition 1. *[3] There exists exactly one maximal consistent extension for every information system $S = (U, A)$.*

Proof. $EXT(S)$ is nonempty for every S because $S \in EXT(S)$ and obviously $EXT(S)$ is finite for every S. Thus, there are maximal elements in the set $EXT(S)$, ordered as above. Let us assume that there are at least two maximal elements $S'_1 = (U'_1, A)$, $S'_2 = (U'_2, A)$ and let $u'_1 \in U'_1$ but $u'_1 \notin U'_2$. $RUL(S) \subseteq RUL(S'_1)$ and moreover for any $R \in RUL(S)$ such that: $R := a_{i_1} = v_{i_1} \wedge \ldots \wedge a_{i_n} = v_{i_n} \Rightarrow a_p = v_p$ implication $a_{i_1} = a_{i_1}(u') \wedge \ldots \wedge a_{i_n} = a_{i_n}(u') \Rightarrow a_p = a_p(u')$ is identical with R or its predecessor differs from predecessor of R. Therefore, $RUL(S) \subseteq RUL(S''_2)$, where $S''_2 = (U'_2 \cup \{u'_1\}, A)$ and that is contradiction with maximal character of S'_2. □

Suraj in [3] describes algorithms of finding maximal consistent extensions for arbitrary information system S with use of the set $OPT(S)$. However, the problem of $OPT(S)$ generating is NP-hard (cf. [4]). Moreover, the set $OPT(S)$ is unique for every information system S. Therefore, it seems natural to try to miss the step of $OPT(S)$ generating in the process of maximal consistent extensions finding. It is also useful to check, not in a very complex way, whether there exists non-trivial consistent extension. Now, we will prove, proposed in [3], necessary condition of the fact.

Proposition 2. *[3] If $S = (U, A)$ has non-trivial consistent extension S', then for at least two attributes $a_1, a_2 \in A$ $card(V_{a_1}) > 2$ and $card(V_{a_2}) > 2$.*

First we will prove the following lemma.

Lemma 1. *If information system $S = (U, A)$ has non-trivial consistent extension S', then for at least one attribute $a \in A$ $card(V_a) > 2$.*

Proof (of Lemma). Let the information system $S = (U, A)$ $(card(A) = k)$ have a consistent extension $S' = (U', A)$. Then, for every $u' \in U'$, such that $v(u') = (a_1(u'), \ldots, a_k(u'))$ the following conditions are satisfied:

1. $\forall i \leqslant k \exists u \in U[a_i(u') = a_i(u)]$;
2. $RUL(S) \subseteq RUL(S'')$, where $S'' = (U \cup \{u'\})$.

Moreover, let

3. $\forall a_i \in A[card(V_{a_i}) \leqslant 2]$.

We will show that conditions 1. - 3. imply existence of only a trivial consistent extension for a given information system S. The rule of mathematical induction is used to show, that for satisfied conditions 1. - 3. every object $u' \in U'$ is indiscernible from some object $u \in U$ with respect to all k attributes. Condition 1 shows that for any $u' \in U'$ there is object $u \in U$ such that $a_i(u) = a_i(u')$ for arbitrary $i \leqslant k$. Now, let $u \in U$ be such object that $a_{i_1}(u) = a_{i_1}(u') \wedge \ldots \wedge a_{i_n}(u) = a_{i_n}(u')$ for some $a_{i_1}, \ldots, a_{i_n} \in A$. If $a_{i_{n+1}}(u) = a_{i_{n+1}}(u')$ for some $a_{i_{n+1}} \in A$ different from $a_{i_1}, \ldots, a_{i_n}$ then it is the end of the proof. So, let $a_{i_{n+1}}(u) \neq a_{i_{n+1}}(u')$ for every $a_{i_{n+1}}$ different from $a_{i_1}, \ldots, a_{i_n}$. If condition 2 is satisfied then it is impossible that the rule $a_{i_1} = a_{i_1}(u) \wedge \ldots \wedge a_{i_n} = a_{i_n}(u) \Rightarrow a_{i_{n+1}} = a_{i_{n+1}}(u)$ is true in system S. That means there is object $w \in U$, such that $a_{i_1}(u) = a_{i_1}(w) \wedge \ldots \wedge a_{i_n}(u) = a_{i_n}(w) \wedge a_{i_{n+1}}(u) \neq a_{i_{n+1}}(w)$. Taking into account condition 3 and the fact that $a_{i_{n+1}}(u) \neq a_{i_{n+1}}(u')$ it is easy to prove that $a_{i_{n+1}}(u') = a_{i_{n+1}}(w)$. The rule of mathematical induction ends the proof of lemma. □

Proof (of Proposition 2). Let $u' \in U'$ and $v(u') = (a_1(u'), a_2(u'), \ldots, a_k(u'))$. Conditions 1. - 2. from the lemma are satisfied and let the following condition 3' be true:

3'. Exactly one attribute $a \in A$ has more than 2 values.

In the same way as above it may be shown, that if some object $u \in U$ is indiscernible from u' with respect to n ($n < k$) attributes, including a ($card(V_a) > 2$), then there is also object $w \in U$ indiscernible from u' with the use of $n+1$ attributes, $n < k$. □

Corollary 1. *If for $S = (U, A)$ there exists non-trivial consistent extension $S' = (U', A')$ and $B = \{a \in A \mid$ for no more than one attribute $card(V_a) > 2\}$, then $\forall u' \in U' - U \exists u \in U[(u, u') \in IND(B)]$.*

Table 1.

$U \setminus A$	a	b	c
u_1	0	2	1
u_2	1	0	1
u_3	1	1	0
u_4	1	1	2

Information system S presented in Table 1 satisfies necessary condition for existence of non-trivial, consistent extension. None of objects u such that $v(u) = (0, 0, *)$ or $v(u) = (0, 1, *)$ or $v(u) = (1, 2, *)$ or $v(u) = (0, *, 0)$ or $v(u) = (0, *, 2)$ belongs to consistent extension of system S.

The following proposition gives necessary and sufficient condition for existence of non-trivial consistent extension of a given information system.

Proposition 3. *An information system $S' = (U', A)$ is a consistent extension of system $S = (U, A)$ if and only if $\forall u \in U, u' \in U', a \in A[(u', u) \notin IND(\{a\}) \Rightarrow u, \sim a, U - RR = u, \sim a, U \cup \{u'\} - RR]$.*

Proof. Validity of sufficient condition is implied by correctness of algorithm of set $OPT(S)$ generating. Now, let us check whether the necessary condition is true. So, let $u, \sim a, U - RR \neq u, \sim a, U \cup \{u'\} - RR$ for some $u \in U, u' \in U'$ and $a \in A$ such that $(u', u) \notin IND(\{a\})$. Obviously, every element of the set $u, \sim a, U - RR$ is a subset of some element from the set $u, \sim a, U \cup \{u'\} - RR$. Let $R \in u, \sim a, U - RR$ be a proper subset of $R' \in u, \sim a, U - \{u'\} - RR$. It means that the rule, for which predecessor is created from R and which is true in S, is not true in S'. This implies that $OPT(S)$ is not a subset of $OPT(S')$ and S' is not a consistent extension of S. □

Below, the algorithm of maximal consistent extension generation for a given information system is presented. This algorithm is based on propositions presented above and misses the stage of the set $OPT(S)$ generating.

`Algorithm 1`

`Input:` Information system $S = (U, A)$, $card(A) = k$.

`Output:` Maximal consistent extension of S.

Step 1. Verify, whether there are at least two attributes a_i, $a_j \in A$, such that $card(V_{a_i}) > 2$ and $card(V_{a_j}) > 2$; if not – go to *Step* 10.

Step 2. From among objects u', such that $v(u') \in V_{a_1} \times \ldots \times V_{a_k}$, throw away all belonging to U.
Step 3. Test, whether there exist such attributes $a \in A$ that $card(V_a) \leqslant 2$; if not – go to *Step* 5.
Step 4. Throw away all objects that do not satisfy corollary concerning necessary condition for existence of non-trivial consistent extension.
Step 5. Create a discernibility matrix for information system S.
Step 6. Complete matrix made in *Step* 5 with a column corresponding to object u' which has not been thrown away, yet. If there is no such object – go to *Step* 10.
Step 7. For the first cell in added column do the following operation: check whether for every attribute from the cell there exists a cell in the same row of the discernibility matrix such that includes considered attribute and which is a subset of the cell from added column. If for some attribute such cell is not found, break the comparison, otherwise repeat the same operation with next cells of the added column. If all cells have been checked, then break the comparison and go to the *Step* 8; otherwise go to *Step* 9.
Step 8. $U := U \cup \{u'\}$.
Step 9. Delete added column, throw away considered object and go back to *Step* 6.
Step 10. Write $S = (U, A)$.

The meaning of the *Step* 7 is as follows: it tests whether $u, \sim a, U - RR = u, \sim a, U \cup \{u'\} - RR$ without those reducts' generating; there is only used the fact that $X \cap Y = X$ if and only if $X \subseteq Y$. X is a symbol of $u, \sim a, U - RR$ and Y denotes $u, \sim a, U \cup \{u'\} - RR$.

Unfortunately, it may happen that not too many objects will be reduced in *Step* 4 or the step will not be made at all. Then, a sequence of *Steps* 6-9 will be repeated $\prod_{p=1}^{k} card(V_{a_p}) - card(U)$ times, in spite of possibility that there is no non-trivial consistent extension for a given information system. Thus, the algorithm is useful rather for systems with considerable number of attributes with no more than 2 values, or when the number $\prod_{p=1}^{k} card(V_{a_p}) - card(U)$ is not too large. If a given system is little, and $\prod_{p=1}^{k} card(V_{a_p}) - card(U)$ is large, then the algorithm with $OPT(S)$ generation may be more efficient.
Algorithm for finding of maximal consistent extension of decision systems is similar to `Algorithm 1`. The difference results two facts: necessary condition for existence of non-trivial extension formulated for information systems is not true for decision systems (see Table 2). Moreover, verifications or comparisons made in the case of information systems for all attributes focus on decision attribute only in decision systems. Thus algorithm for determining of consistent extension of decision system is a sequence of the following steps from `Algorithm 1`: *Step* 2, *Step* 4 - *Step* 10 with simplified *Step* 7.

2.2 Consistent Restrictions of Information Systems

A dual problem for finding maximal consistent extension of a given information system is a problem of minimal consistent restriction determining. This is not a

Table 2.

$U \setminus A$	a	b	c	d
u_1	0	1	1	0
u_2	0	1	0	1
u_3	1	0	1	0
u_4	1	0	0	1
u'	0	0	1	0
u''	0	0	0	1

rule that there exists exactly one minimal consistent restriction for any information system. For information system given by Table 3, two minimal consistent restrictions exist. Those are as follows: the system without object u_6 or without object u_7. Yet, the system without u_6 and u_7 is not a minimal consistent restriction of a given system.

Table 3.

$U \setminus A$	a	b
u_1	1	1
u_2	0	0
u_3	0	2
u_4	2	0
u_5	3	2
u_6	1	0
u_7	1	2

Necessary condition for existence of non-trivial consistent restriction for a given information system is analogous to condition of non-trivial extensions.

Proposition 4. *If $S = (U, A)$ has non-trivial consistent restriction then for at least two attributes $a_1, a_2 \in A$ $card(V_{a_1}) > 2$ and $card(V_{a_2}) > 2$.*

Proof. If Proposition 4 is not true, then S is not a non-trivial consistent extension of any information system. In consequence, there is not any non-trivial consistent restriction of S. □

Both, the corollary for Proposition 4 and the necessary and sufficient condition for existence of non-trivial consistent restriction are also analogous to the corresponding propositions of consistent extensions so proofs of correctness of them as like correctness of `Algorithm 2` that is based on those theorems are omitted.

Corollary 2. *If $S' = (U', A)$ is non-trivial consistent restriction of $S = (U, A)$ and $B = \{a \in A \mid$ for no more than one attribute $card(V_a) > 2\}$, then $\forall u' \in U' - U \exists u \in U[(u, u') \in IND(B)]$.*

Proposition 5. *An information system* $S = (U, A)$ *is a consistent restriction of system* $S' = (U', A)$ *if and only if* $\forall u \in U, u' \in U', a \in A[(u, u') \notin IND(\{a\}) \Rightarrow u, \sim a, U - RR = u, \sim a, U \cup \{u'\} - RR]$.

The following algorithm of minimal restriction determining is based on the above propositions.

`Algorithm 2`

`Input:` Information system $S = (U, A)$.

`Output:` Minimal consistent restriction of S.

Step 1. Check, whether there are at least two attributes $a_i, a_j \in A$, such that $card(V_{a_i}) > 2$ and $card(V_{a_j}) > 2$; if not – go to *Step* 7.

Step 2. Determine set U' of objects belonging to U and indiscernible with some objects from U with the use of every set B from corollary.

Step 3. Create a discernibility matrix for information system $S = (U, A)$.

Step 4. Choose arbitrary object u from set U'; if the set is empty – go to *Step* 7.

Step 5. For object chosen in *Step* 4 check, whether it belongs to consistent extension of system $S' = (U - \{u\}, A)$; if yes, then $U := U - \{u\}$ and modify discernibility matrix by reducing a row and a column corresponding to object u.

Step 6. Throw away object u from set U' and come back to *Step* 4.

Step 7. Write system $S = (U, A)$.

When *Steps* 1 -2 are missed and set U' in *Step* 4 is replaced by U, then algorithm for minimal restriction of decision system determinig is received.

3 Strict Consistent Extensions and Restrictions

Now, let us consider a special kind of consistent extensions or restriction of information (decision) system. If a system S' is a consistent extension of system S and $RUL(S') = RUL(S)$, then system S' is called a strict consistent extension of system S and respectively S is called a strict consistent restriction of S'. The following proposition expresses necessary and sufficient condition for existence of a strict consistent extension S' of system S:

Proposition 6. *System* $S' = (U', A)$ *is a strict consistent extension of system* $S = (U, A)$ *if and only if* S' *is consistent extension of* S *and*

$$\forall u' \in U' - U, a \in A[u', a, U \cup \{u'\} - RR \cap u', \sim a, U' - RR = \varnothing].$$

Proof. It is enough to prove that the second condition of proposition 6 is equivalent with $RUL(S') \subseteq RUL(S)$. Let us put attention on arbitrary object $u' \in U' - U$ and attribute $a \in A$ and let $X = \{a_{i_1}, \ldots, a_{i_n}\}$ $(a \notin X)$ be an element of $u', a, U \cup \{u'\} - RR \cap u', \sim a, U' - RR$. $X \in u', \sim a, U' - RR$ imlplies that object u' is discernible from all objects $u \in U' - \{u'\}$ such that $(u, u') \notin IND(\{a\})$ with the use of reduct X and rule $a_{i_1} = a_{i_1}(u') \wedge \ldots \wedge a_{i_n} = a_{i_n}(u') \Rightarrow a = a(u')$ is true in system S'. $X \in u', a, U \cup \{u'\} - RR$ means that rule $a_{i_1} = a_{i_1}(u') \wedge \ldots \wedge a_{i_n} = a_{i_n}(u') \Rightarrow a = a(u')$ is not true in system S. That is equivalent to $RUL(S') \not\subseteq RUL(S)$. □

Proposition 6 is true for decision systems if expression '$\forall a \in A$' is replaced with 'decision attribute'. Moreover, set of reducts $u', \sim a, U' - RR$ that appears in the proposition means the impossibility of checking one after another whether objects $u' \notin U$ belong to strict consistent extension. Table 2 presents decision system with its strict consistent extension, but system S with universum $U = (u_1, u_2, u_3, u_4)$ extended with only one object from among u', u'' is not strict consistent extension of S because objects u' and u'' added one by one 'bring in' rule $a = 0 \wedge b = 0 \Rightarrow d = 0$ or rule $a = 0 \wedge b = 0 \Rightarrow d = 1$, respectively.

4 Conclusions

In the paper the new results concerning the extensions and restrictions of information systems have been presented. The results involve necessary and sufficient conditions together with the proofs for existence of extensions (restrictions) of information systems. Moreover, new algorithms for determining of extensions (restrictions) without the set of minimal rules generating have been proposed. It is still a challenge to pass more efficient algorithm of consistent extensions (restrictions) generating for large information systems, for instance by finding stronger necessary condition of existing non-trivial extensions (restrictions) of an information system. Actually algorithms proposed in the paper are implemented to execute some experiments using real life data. It will make possible to compare efficiency of the algorithms described in the paper and those ones described in [3].

Acknowledgment. This work was partially supported by the grant #8 T11C 025 19 from the State Committee for Scientific Research (KBN) in Poland.

References

1. Pawlak, Z.: Rough sets: Theoretical Aspects of Reasoning About Data, Boston, MA, Kluwer Academic Publishers, 1991
2. Suraj, Z.: Rough Set Methods for the Synthesis and Analysis of Concurrent Processes, in: Studies in Fuzziness and Soft Computing, Rough Set Methods and Applications. New Developments in Knowledge Discovery in Information Systems, L. Polkowski, S. Tsumoto, T.Y. Lin (Eds.), Physica-Verlag,Heidelberg, 2000, 379-488
3. Suraj, Z.: Some Remarks on Extensions and Restrictions of Information Systems. Proceedings of the Second International Conference on Rough Sets and Current Trends in Computing (RSCTC'2000), Banff, Canada, October 16-19, 2000, 172–179
4. Wegener, I.: The complexity of Boolean Functions, Wiley and B.G. Teubner Stuttgart 1987