

## Chapter 7

# Classifiers Based on Data Sets and Domain Knowledge: A Rough Set Approach

Jan G. Bazan, Stanisława Bazan-Socha, Sylwia Buregwa-Czuma, Przemysław Wiktor Pardel, Andrzej Skowron, and Barbara Sokolowska

**Abstract** The problem considered is how to construct classifiers for approximation of complex concepts on the basis of experimental data sets and domain knowledge that are mainly represented by concept ontology. The approach presented in this chapter to solving this problem is based on the rough set theory methods. Rough set theory introduced by Zdzisław Pawlak during the early 1980s provides the foundation for the construction of classifiers. This approach is applied to approximate spatial complex concepts and spatio-temporal complex concepts defined for complex objects, to identify the behavioral patterns of complex objects, and to the automated behavior planning for such objects when the states of objects are represented

---

Jan G. Bazan

Institute of Computer Science, University of Rzeszów, Dekerta 2, 35 - 030 Rzeszów, Poland  
Institute of Mathematics, Warsaw University Banacha 2, 02-097 Warsaw, Poland  
e-mail: bazan@univ.rzeszow.pl

Stanisława Bazan-Socha

Second Department of Internal Medicine, Jagiellonian University Medical College, Skawinska 8, 31-066 Cracow, Poland  
e-mail: mmsocha@cyf-kr.edu.pl

Sylwia Buregwa-Czuma

Institute of Computer Science, University of Rzeszów, Dekerta 2, 35 - 030 Rzeszów, Poland  
e-mail: szczuma@univ.rzeszow.pl

Przemysław Wiktor Pardel

Institute of Computer Science, University of Rzeszów, Dekerta 2, 35 - 030 Rzeszów, Poland  
Institute of Mathematics, Warsaw University Banacha 2, 02-097 Warsaw, Poland  
e-mail: ppardel@univ.rzeszow.pl

Andrzej Skowron

Institute of Mathematics, Warsaw University Banacha 2, 02-097 Warsaw, Poland  
e-mail: skowron@mimuw.edu.pl

Barbara Sokolowska

Second Department of Internal Medicine, Jagiellonian University Medical College, Skawinska 8, 31-066 Cracow, Poland  
e-mail: basiasok1@gmail.com

by spatio-temporal concepts requiring approximation. The chapter includes results of experiments that have been performed on data from a vehicular traffic simulator and the recent results of experiments that have been performed on medical data sets obtained from Second Department of Internal Medicine, Jagiellonian University Medical College, Krakow, Poland. Moreover, we also describe the results of experiments that have been performed on medical data obtained from Neonatal Intensive Care Unit in the Department of Pediatrics, Jagiellonian University Medical College, Krakow, Poland.

**Key words:** Rough set, concept approximation, complex dynamical system, ontology of concepts, behavioral pattern identification, automated planning

## 7.1 Introduction

*Classifiers* also known in literature as *decision algorithms*, *classifying algorithms* or *learning algorithms* may be treated as constructive, approximate descriptions of concepts (decision classes). These algorithms constitute the kernel of *decision systems* that are widely applied in solving many problems occurring in such domains as *pattern recognition*, *machine learning*, *expert systems*, *data mining* and *knowledge discovery* (see, e.g., [17, 21, 24, 29–31, 40]).

In literature there can be found descriptions of numerous approaches to constructing classifiers, which are based on such paradigms of machine learning theory as *classical and modern statistical methods* (see, e.g., [30, 40]), *neural networks* (see, e.g., [30, 40]), *decision trees* (see, e.g., [30]), *decision rules* (see, e.g., [29, 30]), and *inductive logic programming* (see, e.g., [30]). Rough set theory introduced by Zdzisław Pawlak during the early 1980s also provides the foundation for the construction of classifiers (see, e.g., [1, 33, 34, 36]).

Recently, it has been noticed in the literature that with the development of modern civilization, not only the scale of the data gathered but also the complexity of concepts and phenomena which they concern are increasing rapidly. This crucial data change has brought new challenges to work out new data mining methods. Particularly, data more and more often concerns complex processes which do not give in to classical modeling methods. Of such a form may be medical and financial data, data coming from vehicles monitoring, or data about the users gathered on the Internet. Exploration methods of such data are in the center of attention in many powerful research centers in the world, and at the same time detection of models of complex processes and their properties (patterns) from data is becoming more and more attractive for applications (see, e.g., [2, 12, 26, 32, 45, 47]).

When modeling complex real-world phenomena and processes mentioned above and solving problems under conditions that require an access to various distributed data and knowledge sources, the so-called *complex dynamical systems* (CDS) are often applied (see, e.g., [4, 14, 28, 48]), or putting it in other way *autonomous multi-agent systems* (see, e.g., [20, 27, 28]) or *swarm systems* (see, e.g., [35]). These

are collections of complex interacting objects characterized by constant change of parameters of their components and their interactions over time, numerous relationships between the objects, the possibility of cooperation/competition among the objects and the ability of objects to perform more or less compound actions. Examples of such systems are *traffic, a patient observed during treatment, a team of robots performing tasks*, etc. It is also worthwhile mentioning that the description of a CDS dynamics is often not possible with purely analytical methods as it includes many complex vague concepts (see, *e.g.*, [23, 39]). Such concepts concern properties of chosen fragments of the CDS and may be treated as more or less complex objects occurring in the CDS. Hence, are needed appropriate methods of extracting such fragments (granules [6]) that are sufficient to conclude about the global state of the CDS in the context of the analyzed types of changes and behaviors. The identification of complex spatio-temporal concepts and using them to monitor a CDS requires approximation of these concepts.

Making a progress in this field is extremely crucial, among other things, for the development of intelligent systems making decision under uncertainty on the basis of results of analysis of the available data sets. Therefore, working out methods of detection of process models and their properties from data and proving their effectiveness in different applications are of particular importance for the further development of decision supporting systems in many domains such as medicine, finance, industry, transport, telecommunication, and others.

However, essential limitations have been discovered concerning the existing data mining methods for very large data sets regarding complex concepts, phenomena, or processes (see, *e.g.*, [13, 37, 49–51]). A crucial limitation of the existing methods is, among other things, the fact that they do not support an effective approximation of complex concepts, that is, concepts whose approximation requires discovery of extremely complex patterns. Intuitively, such concepts are too far in the semantical sense from the available concepts, *e.g.*, sensory ones. As a consequence, the size of searching spaces for relevant patterns crucial for approximation are so large that an effective search of these spaces very often becomes unfeasible using the existing methods and technology. Thus, as it turned out, the ambition to approximate complex concepts with high quality from available concepts (most often defined by sensor data) in a fully automatic way, realized by the existing systems and by most systems under construction, is a serious obstacle since the classifiers obtained are often of unsatisfactory quality.

Moreover, it has been noticed in the literature (see, *e.g.*, [16, 25, 37, 44]) that one of the challenges for data mining is discovery of methods linking detection of patterns and concepts with domain knowledge. The latter term denotes knowledge about concepts occurring in a given domain and various relations among them. This knowledge greatly exceeds the knowledge gathered in data sets; it is often represented in a natural language and usually acquired during a dialogue with an expert in a given domain. One of the ways to represent domain knowledge is to record it in the form of the so-called *concept ontology* where ontology is usually understood as a finite hierarchy of concepts and relations among them, linking concepts from different levels (see, *e.g.*, [18, 22]).

The main aim of the chapter is to present the developed methods for approximation of complex vague concepts involved in specification of real-life problems and approximate reasoning used in solving these problems. However, methods presented in the chapter are assuming that additional domain knowledge in the form of the concept ontology is given. Concepts from ontology are often vague and expressed in natural language. Therefore, an approximation of ontology is used to create hints in searching for approximation of complex concepts from sensory (low level) data.

We propose to link automatic methods of complex concept learning, and models of detection of processes and their properties with domain knowledge obtained in a dialogue with an expert. Interaction with a domain expert facilitates guiding the process of discovery of patterns and models of processes and makes the process computationally feasible.

As we mentioned before, our methods for approximating complex spatio-temporal concepts and relations among them assuming that the information about concepts and relations is given in the form of ontology. To meet these needs, by ontology we understand a finite set of concepts creating a hierarchy and relations among these concepts which link concepts from different levels of the hierarchy. At the same time, on top of this hierarchy there are always the most complex concepts whose approximations we are interested in aiming at practical applications. Moreover, we assume that the ontology specification contains incomplete information about concepts and relations occurring in ontology, particularly for each concept, sets of objects constituting examples and counterexamples for these concepts are given. Additionally, for concepts from the lowest hierarchical level (sensor level) it is assumed that there are also *sensor attributes* available which enable to approximate these concepts on the basis of positive and negative examples given (see example of ontology from Fig. 7.1 and [43]).

In this chapter, we present the following four types of methods for approximating spatial or spatio-temporal complex concepts.

1. Methods of approximation of spatial concepts - when a complex concept is a spatial concept not requiring an observation of changes over time (see Section 7.2).
2. Methods of approximation of spatio-temporal concepts - when a complex concept is a spatio-temporal concept; it requires observing changes of complex objects over time (see Section 7.3).
3. Methods of behavioral pattern identification - when a complex concept is represented as a certain directed graph which is called *a behavioral graph* (see Section 7.4).
4. Methods of automated behavior planning for complex object - when the states of objects are represented by spatio-temporal concepts requiring approximation (see Section 7.5).

The result of the works conducted is also a programming system the *Rough Set Interactive Classification Engine* (RoughICE), supporting the approximation of spatio-temporal complex concepts in the given ontology in the dialogue with the

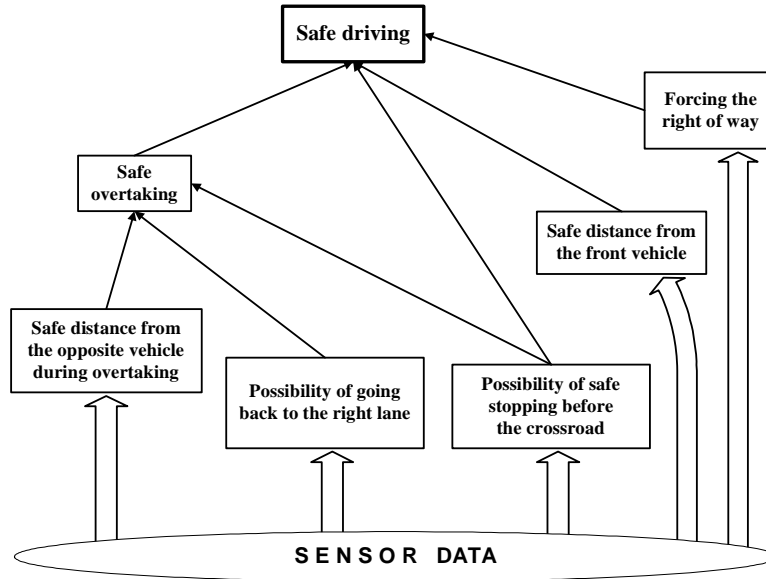


Fig. 7.1 An ontology for safe driving

user. The RoughICE includes an implementation of the algorithmic methods presented in this chapter and is available on the web side [41]. To simplify the use of RoughICE algorithms and make it more intuitive the RoughICE graphical user interface was constructed that consists of three following parts (see Figure 7.2):

1. *project designer* - directed towards visual representation of workflow,
2. *graph editor* - for representing domain knowledge in the form of ontology,
3. *script editor and compiler* - for representing domain knowledge in the form of scripts.

Sections 7.2, 7.4 and 7.5, apart from the method description, contain the results of computing experiments conducted on real-life data sets, supported by domain knowledge. It is worth mentioning that the requirements regarding data sets which can be used for computing experiments with modeling spatio-temporal phenomena are much greater than the requirements of the data which are used for testing process of classical classifiers. Not only have the data to be representative of the decision making problem under consideration but also they have to consist the relevant domain knowledge about approximated concepts (usually cooperation with experts in a particular domain is essential for acquisition of domain knowledge). It is important that such data should fully and appropriately represent complex spatio-temporal phenomena of the environment.

The authors of the chapter acquired such data sets from three sources.

The first source of data is the traffic simulator (see [43] and [5] for more details). The simulator is a computing tool for generating data sets connected to the traffic on

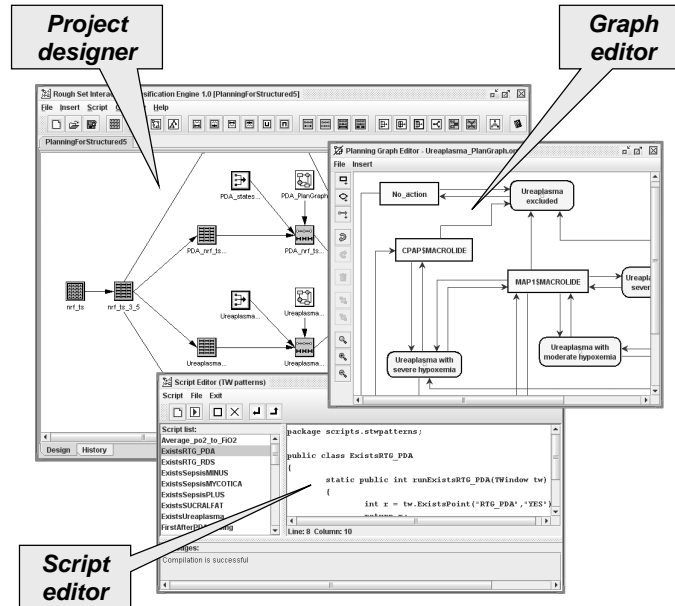


Fig. 7.2 The view of RoughICE graphical user interface

the street and at crossroads. During simulation each vehicle appearing on the simulation board behaves as an independently acting agent. On the basis of observation of the surroundings (other vehicles, its own location, weather conditions, etc.) this agent makes an independent decision what maneuvers it should make to achieve its goal which is to go safely across the simulation board and to leave the board using the outbound way given in advance. At any given moment of the simulation, all crucial vehicle parameters may be recorded, and thanks to this data sets for experiments can be obtained. The results of experiments with the data sets recorded in the road simulator we present in Section 7.2.

The second collection of data sets used in computer experiments was provided by Second Department of Internal Medicine, Collegium Medicum, Jagiellonian University, Krakow, Poland. This data includes characteristics of patients with stable coronary heart disease: clinical status, past history, the laboratory tests results, electrocardiographic (ECG) recordings, applied therapeutic procedures and coronary angiography outcomes. In the chapter we present recent results of experiments performed for this collection of data sets (see Section 7.4).

The third collection of data sets used in computer experiments was provided by Neonatal Intensive Care Unit, First Department of Pediatrics, Polish-American Institute of Pediatrics, Collegium Medicum, Jagiellonian University, Krakow, Poland. This data constitutes a detailed description of treatment of 300 infants, i.e., treatment results, diagnosis, operations, medication (see [5, 7–10]). The results for this data collection we present in Section 7.5.

## 7.2 Methods of Approximation of Spatial Concepts

The method of approximating concepts from ontology is proposed when a concept is a *spatial concept* (not requiring an observation of changes over time) and it is defined on a set of the same objects (examples) as the lower ontology level concepts; at the same time, the lower level concepts are also spatial concepts. An exemplary situation of this type is an approximation of the concept of *Safe overtaking* (concerning single vehicles on the road) using concepts such as *Safe distance from the opposite vehicle during overtaking*, *Possibility of going back to the right lane* and *Possibility of safe stopping before the crossroads* (see Fig. 7.1).

In the chapter, the method of approximating concepts from ontology is proposed when a higher ontology level concept is a spatial concept (not requiring an observation of changes over time) and it is defined on a set of the same objects (examples) as the lower ontology level concepts; at the same time, the lower level concepts are also spatial concepts. An exemplary situation of this type is an approximation of the concept of *Safe overtaking* (concerning single vehicles on the road) using concepts such as *Safe distance from the opposite vehicle during overtaking*, *Possibility of going back to the right lane* and *Possibility of safe stopping before the crossroads*.

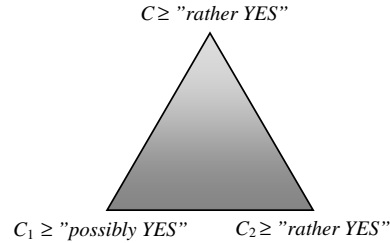
The concept approximation method described in this subsection is an example of the general methodology of approximating concepts from ontology described in [5]. That is why its specificity is the domain knowledge usage expressed in the form of a concept ontology and application of rough set methods, mainly in terms of application of classifier construction methods.

The basic terms used in the presented method is *pattern* and *production rule*. Patterns are descriptions of examples of concepts from an ontology and they are constructed by a *stratifying classifier*, defined as a classifying algorithm stratifying concepts, that is, classifying objects to different concept layers (see [5] for more details). Two approaches have been proposed to construction of these classifiers. One of them is *the expert approach* which is based on the defining, by an expert, an additional attribute in data which describes membership of the object to individual concept layers. Next, a classifier differentiating layers as decision classes is constructed. The second approach called *the automated approach* is based on the designing algorithms being the classifier extensions which enable to classify objects to concept layers on the basis of certain premises and experimental observations. In [5] a method of this type has been proposed which is based on shortening of decision rules relatively to various coefficients of consistency.

For example, we consider a concept  $C$ , where inclusion to this concept is described by six linearly ordered layers “*certainly NO*”, “*rather NO*”, “*possibly NO*”, “*possibly YES*”, “*rather YES*” and “*certainly YES*”. For the concept  $C$  we define a pattern ( $C \geq$  “*rather YES*”), that has the following interpretation: *the inclusion to the concept  $C$  is at least “rather YES”*. It is easy to see that a stratifying classifier can be used to judge whether a tested object belongs to this pattern or not.

A production rule is a decision rule which is constructed on two adjacent levels of ontology. In the *predecessor* of this rule there are patterns for the concepts from the lower level of the ontology whereas in the *successor*, there is a pattern for one

concept from the higher level of the ontology (connected with concepts from the rule predecessor) where both patterns from the predecessor and the successor of the rule are chosen from patterns constructed earlier for concepts from both adjacent levels of the ontology. In Fig. 7.3 we present an example of production rule for concepts  $C_1$ ,  $C_2$  and  $C$ . This production rule has the following interpretation: if inclusion degree to a concept  $C_1$  is at least “possibly YES” and to concept  $C_2$  at least “rather YES” then the inclusion degree to a concept  $C$  is at least “rather YES”.



**Fig. 7.3** The example of production rule

A rule constructed in such a way may serve as a simple classifier or an argument “for”/“against” the given concept, enabling classification of objects which match the patterns from the rule predecessor with the pattern from the rule successor. For example, the object  $u_1$  from Fig. 7.4 is classified by production rule from Fig. 7.3 because it matches both patterns from the left hand side of the production rule whereas, the object  $u_2$  from Fig. 7.4 is not classified by production rule because it does not match the second source pattern of production rule (the value of attribute  $C_2$  is less than “rather YES”).

In [5], there was proposed an algorithmic method of induction of production rules, consisting in an appropriate search for data tables with attributes describing the membership of training objects to particular layers of concepts. These tables (called a *layer table*) are constructed using the so-called constraints between concepts thanks to which the information put in the tables only concerns those objects/examples which might be found there according to the production rule under construction. In Fig. 7.5 we illustrate the process of extracting production rule for concept  $C$  and for the approximation layer “rather YES” of concept  $C$ . It is easy to see that if from the table from Fig. 7.5 we select all objects satisfying  $a_C = \text{“rather YES”}$ , then for selected objects minimal value of the attribute  $a_{C_1}$  is equal to “possibly YES” and minimal value of the attribute  $a_{C_2}$  is equal to “rather YES”. Hence, we obtain the production rule:

$$(C_1 \geq \text{“possibly YES”}) \wedge (C_2 \geq \text{“rather YES”}) \Rightarrow (C \geq \text{“rather YES”}).$$

Although a single production rule may be used as a classifier for the concept appearing in a rule successor, it is not a complete classifier yet, i.e., classifying all objects belonging to an approximated concept and not only those matching pat-



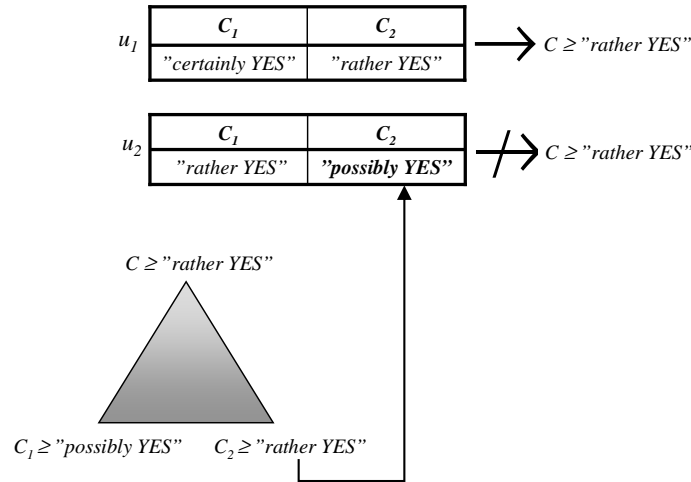


Fig. 7.4 Classifying tested objects by single production rule

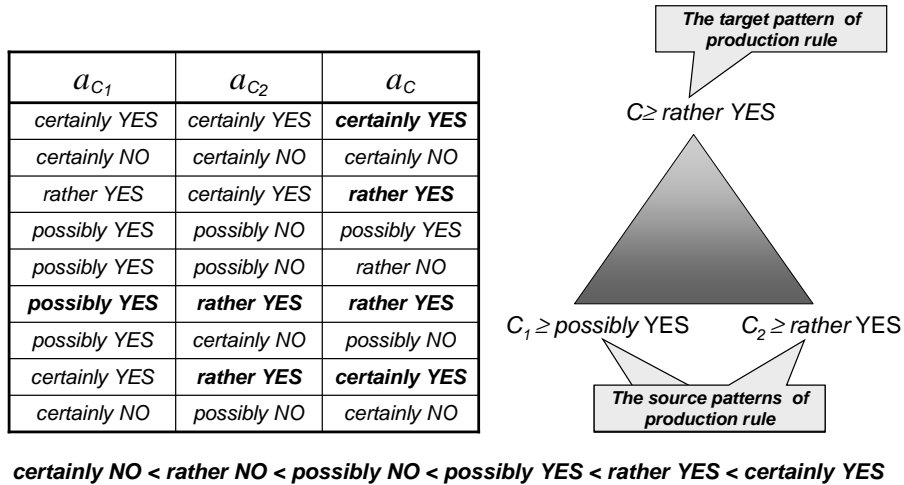
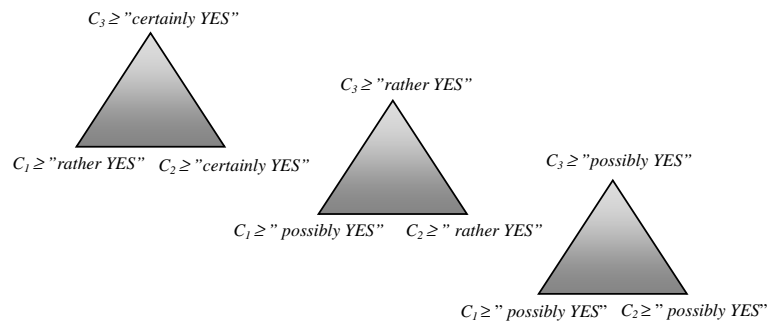


Fig. 7.5 The illustration of production rule extracting

terns of a rule predecessor. Therefore, in practice, production rules are grouped into the so-called *productions*, i.e., production rule collections, in a way that each production contains rules having patterns for the same concepts in a predecessor and the successor, but responding to their different layers. In Fig. 7.6 we present three production rules constructed for some concepts  $C_1$ ,  $C_2$  and  $C$  approximated by six linearly ordered layers “certainly NO”, “rather NO”, “possibly NO”, “possibly YES”, “rather YES” and “certainly YES”. This collection of production rules

is an exemplary production for concepts  $C_1$ ,  $C_2$  and  $C$ . Moreover, production rules from Fig. 7.6 have the following interpretation:

1. if inclusion degree to a concept  $C_1$  is at least “rather YES” and to concept  $C_2$  at least “certainly YES” then the inclusion degree to a concept  $C$  is at least “certainly YES”;
2. if the inclusion degree to a concept  $C_1$  is at least “possibly YES” and to a concept  $C_2$  at least “rather YES” then the inclusion degree to a concept  $C$  is at least “rather YES”;
3. if the inclusion degree to a concept  $C_1$  is at least “possibly YES” and to a concept  $C_2$  at least “possibly YES” then the inclusion degree to a concept  $C$  is at least “possibly YES”.



**Fig. 7.6** The example of production as a collection of three production rules

In the case of production from Fig. 7.6, concept  $C$  is the target concept and  $C_1$ ,  $C_2$  are the source concepts.

Such production makes it possible to classify much more objects than a single production rule where these objects are classified into different layers of the concept occurring in a rule successor. Both productions and production rules themselves are only constructed for the two adjacent levels of ontology. Therefore, in order to use the whole ontology fully, there are constructed the so-called AR-schemes, i.e., *approximate reasoning schemes* which are hierarchical compositions of production rules (see, e.g., [11, 15, 38]). The synthesis of an AR-scheme is carried out in a way that to a particular production from a lower hierarchical level of the AR-scheme under construction another production rule on a higher level may be attached, but only that one where one of the concepts for which the pattern occurring in the predecessor was constructed is the concept connected with the rule successor from the previous level. Additionally, it is required that the pattern occurring in a rule predecessor from the higher level is a subset of the pattern occurring in a rule successor from the lower level (in the sense of inclusion of object sets matching both patterns). To the two combined production rules other production rules can be attached (from above, from below or from the side) and in this way a multilevel structure is made

which is a composition of many production rules. The AR-scheme constructed in such a way can be used as a hierarchical classifier whose entrance are predecessors of production rules from the lowest part of the AR-scheme hierarchy and the exit is the successor of a rule from the highest part of the AR-scheme hierarchy. That way, each AR-scheme is a classifier for a concept occurring in the rule successor from the highest part in the hierarchy of the scheme and, to be precise, for a concept for which a pattern occurring in the rule successor from the highest part in the hierarchy of the AR-scheme is determined.

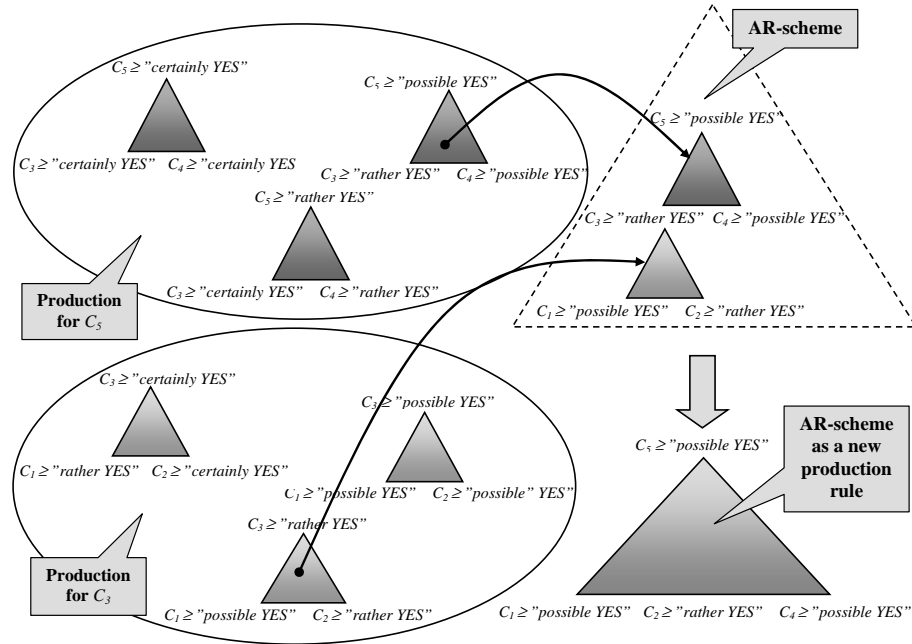


Fig. 7.7 Synthesis of approximate reasoning scheme

For example, in Fig. 7.7 we have two productions. The target concept of the first production is  $C_5$  and the target concept of the second production is the concept  $C_3$ . We select one production rule from the first production and one production rule from the second production. These production rules are composed and then a simple AR-scheme is obtained that can be treated as a new two-levels production rule. Notice, that the target pattern of lower production rule in this AR-scheme is the same as one of the source patterns from the higher production rule. In this case, the common pattern is described as follows: inclusion degree (of some pattern) to a concept  $C_3$  is at least "possibly YES".

In this way, we can compose AR-schemes into hierarchical and multilevel structures using productions constructed for various concepts. AR-scheme constructed in such a way can be used as a hierarchical classifier whose input is given by pre-

deceutors of production rules from the lowest part of AR-scheme hierarchy and the output is a successor of a rule from the highest part of the AR-scheme hierarchy.

However, similarly to the case of a single production rule, an AR-scheme is not a full classifier yet. That is why, in practice, for a particular concept there are many AR-schemes constructed which approximate different layers or concept regions.

In the paper [5], there are proposed two approaches for constructing AR-schemes. The first approach is based on memory with AR-schemes and consists in building many AR-schemes after determining production, which later on are stored and used for the classification of tested objects.

The second approach is based on a dynamic construction of AR-schemes. It is realized in a way that during classification of a given tested object, an appropriate AR-schemes for classifying this particular object is built on the basis of a given collection of productions ("lazy" classification).

### **7.2.1 Experiments with Data**

To verify effectiveness of classifiers based on AR schemes, we have implemented our algorithms in the RoughICE (see Section 7.1 and [5]).

The experiments have been performed on the data set obtained from the road simulator (see [43]). Data set consists of 18101 objects generated by the road simulator. We have applied the train and test method. The data set was randomly divided into two parts: training and test ones (50% + 50%). In order to determine the standard deviation of the obtained results each experiment was repeated for 10 random divisions of the whole data set.

In our experiments, we compared the quality of two classifiers: RS and ARS. For inducing RS we use RSES system (see [42]) generating the set of decision rules by algorithm LEM2 (see [5] for more details) that are next used for classifying situations from testing data. ARS is based on AR schemes (the implementation from [41]).

During ARS classifier construction, in order to approximate concepts occurring in ontology we also used the LEM2 algorithm.

For production rule construction we used the expert method of stratifying classifier construction. However, to classify objects using the ARS classifier we used the method of dynamic construction of the AR-schemes for specific tested objects (see [5]).

We compared RS and ARS classifiers using the accuracy, the coverage, the accuracy for positive examples (also called as the sensitivity or the true positive rate), the accuracy for negative examples (also called as the specificity or the true negative rate), the coverage for positive examples and the coverage for negative examples, the real accuracy (where Real accuracy = Accuracy \* Coverage), the learning time and the rule set size.

Decision class	Method	Accuracy	Coverage	Real accuracy
YES	RS	$0.977 \pm 0.001$	$0.948 \pm 0.003$	$0.926 \pm 0.003$
	ARS	$0.967 \pm 0.001$	$0.948 \pm 0.003$	$0.918 \pm 0.003$
NO	RS	$0.618 \pm 0.031$	$0.707 \pm 0.010$	$0.436 \pm 0.021$
	ARS	$0.954 \pm 0.016$	$0.733 \pm 0.018$	$0.699 \pm 0.020$
All classes (YES + NO)	RS	$0.963 \pm 0.001$	$0.935 \pm 0.003$	$0.901 \pm 0.003$
	ARS	$0.967 \pm 0.001$	$0.937 \pm 0.004$	$0.906 \pm 0.004$

**Table 7.1** Results of experiments for the concept: *Is the vehicle driving safely?*

Method	Learning time	Rule set size
RS	$488 \pm 21$ seconds	$975 \pm 28$
ARS	$33 \pm 1$ second	$174 \pm 3$

**Table 7.2** Learning time and the rule set size for concept: *Is the vehicle driving safely?*

Table 7.1 shows the results of the considered classification algorithms for the concept *Is the vehicle driving safely?* (see Fig. 7.1). Together with the results we present a standard deviation of the obtained results.

One can see that accuracy of algorithm ARS for the decision class *NO* is higher than the accuracy of the algorithm RS for analyzed data set. The decision class *NO* is smaller than the class *YES*. It represents atypical cases in whose recognition we are most interested in (dangerous driving a vehicle on a highway).

Table 7.2 shows the learning time and the number of decision rules induced for the considered classifiers. In the case of the algorithm ARS we present the average number of decision rules over all concepts from the relationship diagram (see Fig. 7.1).

One can see that the learning time for ARS is much shorter than for RS and the average number of decision rules (over all concepts from the relationship diagram) for ARS algorithm is much lower than the number of decision rules induced for RS.

The experiments showed that classification quality obtained through classifiers based on AR-schemes is higher than classification quality obtained through traditional classifiers based on decision rules (especially in the case of the class *NO*). Apart from that the time spent on classifier construction based on AR-schemes is shorter than when constructing classical rule classifiers. Also, the structure of a single rule classifier (inside the ARS classifier) is less complicated than the structure of RS classifier (a considerably smaller average number of decision rules). It is worth noticing that the performance of the ARS classifier is much more stable than the RS classifier because of the differences in data in samples supplied for learning (e.g., to change the simulation scenario).

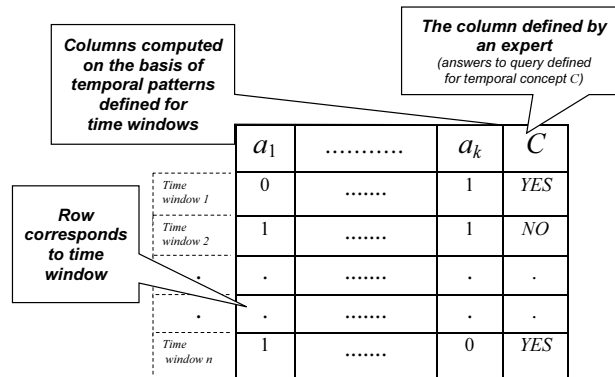
### 7.3 Methods of Approximation of Spatio-temporal Concepts

In this section, we propose a method of approximating concepts from hierarchical ontology when a higher ontology level concept is a spatio-temporal concept (it requires observing changes of complex objects over time) defined on a set of the same objects as the lower ontology level concepts; at the same time, the lower ontology level concepts are spatial concepts only. This case concerns a situation when during an observation of a single object in order to capture its behavior described by a higher ontology level concept, we have to observe it longer than it requires to capture behaviors described by lower ontology level concepts. For example, in case of data sets generated from the traffic simulator, lower ontology level concepts may concern simple vehicle behaviors such as *small increase in speed*, *small decrease in speed* or *small move towards the left lane*. However, the higher ontology level concept may be a more complex concept as, e.g., *acceleration in the right lane*. Let us notice that determining whether a vehicle accelerates in the right lane requires its observation for some time called a *time window*. On the other hand, determining whether a vehicle speed increases in the right lane requires only a registration of the speed of a vehicle in two neighboring instants (time points) only. That is why spatio-temporal concepts are more difficult to approximate than spatial concepts whose approximation does not require observing changes of objects over time.

Similarly to spatial concept approximation (see Section 7.2), the method of concept approximation described in this subsection is an example of the general methodology of approximating concepts from ontology described in [5]. Its specificity is, therefore, the domain knowledge usage expressed in the form of a concept ontology and rough set method application, mainly in terms of application of classifier construction methods. However, in this case more complex ontologies are used, and they contain both spatial and spatio-temporal concepts.

The starting point for the method proposed is a remark that spatio-temporal concept identification requires an observation of a complex object over a longer period of time called a *time window* (see [5]). To describe complex object changes in the time window, the so-called *temporal patterns* (see [5]) are used, which are defined as functions determined on a given time window. These patterns, being in fact formulas from a certain language, also characterize certain spatial properties of the complex object examined, observed in a given time window. They are constructed using lower ontology level concepts and that is why identification whether the object belongs to these patterns requires the application of classifiers constructed for concepts of the lower ontology level. Moreover, temporal patterns are often defined using queries with binary answers such as *Yes* or *No*. For example, in the case of road traffic we have exemplary temporal patterns such as *Did vehicle speed increase in the time window?*, *Was the speed stable in the time window?*, *Did the speed increase before a move to the left lane occurred?* or *Did the speed increase before a speed decrease occurred?*. We assume that any temporal pattern ought to be defined by a human expert using domain knowledge accumulated for the given complex dynamical system.

On a slightly higher abstraction level, the spatio-temporal concepts (also called *temporal concepts*) are directly used to describe complex object behaviors (see [5]). Those concepts are defined by an expert in a natural language and they are usually formulated using questions about the current status of spatio-temporal objects, e.g., *Does the vehicle examined accelerate in the right lane?*, *Does the vehicle maintain a constant speed during lane changing?* The method proposed here is based on approximating temporal concepts using temporal patterns with the help of classifiers. In order to do this a special decision table is constructed called a *temporal concept table* (see [5]). In case of method presented in this chapter, the rows of this table represent the parameter vectors of lower level ontology concepts observed in a time window. Columns of this table (apart from the last one) are determined using temporal patterns. However, the last column represents membership of an object, described by parameters (features, attributes) from a given row, to the approximated temporal concept (see Fig. 7.8).



**Fig. 7.8** The scheme of a temporal concept table

It is worth noticing that the presented above approach to temporal concept approximation can be extended to the case when higher ontology level concepts are defined on a set of objects which are structured objects in relation to objects (examples) of the lower ontology level concepts, that is, the lower ontology level objects are parts of objects from the higher ontology level. This case concerns a situation when during a structured object observation, which serves the purpose of capturing its behavior described by a higher ontology level concept, we must observe this object longer than it is required to capture the behavior of a single part of the structured object described by lower ontology level concepts (see [5] for more details).

## 7.4 Methods of Behavioral Pattern Identification

Temporal concepts may be treated as nodes of a certain directed graph which is called a *behavioral graph*. Links (directed edges) in this graph are the temporal relations between temporal concepts meaning a temporal sequence of satisfying two temporal concepts one after another. These graphs may be used to represent and identify the so-called behavioral patterns which are complex concepts concerning dynamic properties of complex objects expressed in a natural language depending on time and space. Examples of behavioral patterns may be: *overtaking on the road*, *driving in a traffic jam*, *behavior of a patient connected with a high life threat*. These types of concepts are much more difficult to approximate even than many temporal concepts. Fig. 7.9 presents an example of behavioral graph for a single object-vehicle exhibiting a behavioral pattern of vehicle while driving on a road. In this behavioral graph, for example, connections between node *Acceleration on the right lane* and node *Acceleration and changing lanes from right to left* indicates that after an acceleration in the right lane, a vehicle can change to the left lane (maintaining its acceleration during both time windows).

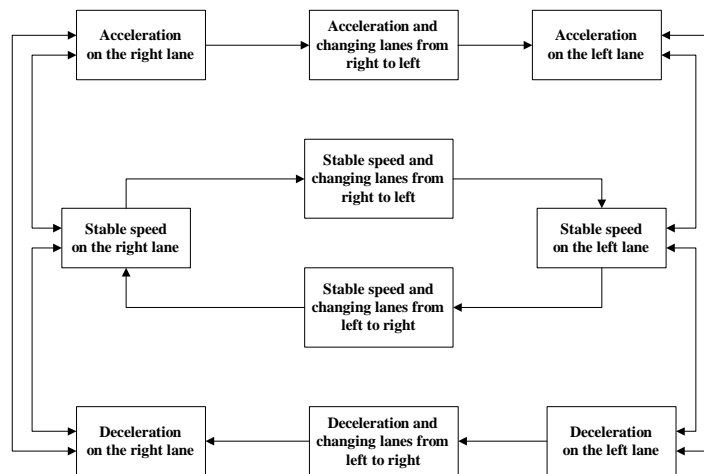


Fig. 7.9 A behavioral graph for a single object-vehicle

In [5] a new method of behavioral pattern identification is presented which is based on interpreting the behavioral graph of a complex object as a complex classifier enabling identification of a behavioral pattern described by this graph. This is possible based on the observation of the complex object behavior for a longer time and checking whether the behavior matches the chosen behavioral graph path. If this is so, then it is determined if the behavior matches the behavioral pattern represented by this graph, which enables a detection of specific behaviors of complex objects.



In order to test the quality and effectiveness of classifier construction methods based on behavioral patterns, there have been performed experiments on data generated from the road simulator and medical data connected to detection of higher-death risk in infants suffering from the respiratory failure (see [5, 7, 9, 10]). The experiments showed that the algorithmic methods presented in this chapter provide very good results in detecting behavioral patterns and may be useful with complex dynamical systems monitoring.

Additionally, in this chapter we present recent results of experiments investigating membership of patients with stable coronary heart disease to behavioral pattern related to risk of *sudden cardiac death* (SCD). Using Holter ECG recordings and well known predictors of SCD, the concepts of SCD risk intensity were defined (see Section 7.4.1).

#### **7.4.1 Risk Pattern Identification in Medical Data**

An identification of some behavioral patterns can be very important for identification or prediction of behavior of complex dynamical system, especially when behavioral patterns describe some dangerous situations. In this case, we call such behavioral patterns *risk patterns* and we need some tools for their identification (see, e.g., [19]). If in the current situation some risk patterns are identified, then the control object (a driver of the vehicle, a medical doctor, a pilot of the aircraft, etc.) can use this information to adjust selected parameters to obtain the desirable behavior of the complex dynamical system. This can make it possible to overcome inconvenient or unsafe situations. For example, a very important element of the treatment of the patients with coronary heart disease is the appropriate assessment of the risk of SCD. The appropriate assessment of this risk leads to the decision of particular method and level of treatment. Therefore, if some complex behavior of a patient that causes a danger of death SCD is identified, we can try to change her/his behavior by using some other methods of treatment (may be more radical) in order to avoid the patients's death. We describe how the presented approach can be applied to identification of the patients SCD risk caused by coronary heart disease (see next subsections). In this approach, a given patient is treated as an investigated complex dynamical system, whilst coronary heart disease is treated as a complex object changing and interacting over time with its environment.

#### **7.4.2 Medical Temporal Patterns**

Data sets used for complex object information storage occurring in a given complex dynamical system may be represented using information systems (see, e.g., [5, 33]). This representation is based on representing individual complex objects by object

(rows) of information system and information system attributes represent the properties of these objects at the current time point.

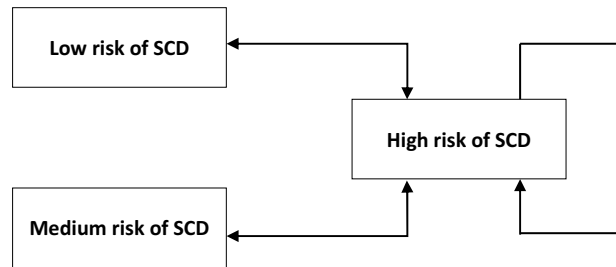
The concepts concerning properties of complex objects at the current time point (spatial concepts) can be defined on the basis of domain knowledge by human experts and can be approximated by properties (attributes) of these objects at the current time point (for instance, using the standard rough set approach to classifier construction [5, 33]).

The concepts concerning properties of complex objects at the current time point in relation to the previous time point are a way of representing very simple behaviors of the objects. However, the perception of more complex types of behavior requires the examination of behavior of complex objects over a longer period of time. This period is usually called the time window, which is to be understood as a sequence of objects of a given temporal information system (a kind of information system with special attribute represents time) registered for the established complex object starting from the established time point over the established period of time or as long as the expected number of time points are obtained. Therefore, learning to recognize complex types of behavior of complex objects with use of gathered data as well as the further use of learned classifiers to identify the types of behavior of complex objects, requires working out of the mechanisms of extraction of time windows from the data and their properties. Hence, if we want to predict such more complex behaviors or discover a behavioral pattern, we have to investigate values of attributes registered in the current time window. Such investigation can be expressed using temporal patterns (see Section 7.3). For example, in the case of the medical example one can consider patterns expressed by following questions: "Did HRV increase in the time window?", "Was the heart rate stable in the time window?", "Did ST interval level increase?" or "Was the QT segment time higher then the right time at any point in time window?". Notice that all such patterns ought to be defined by a human, medical expert using domain knowledge accumulated for the coronary heart disease.

### ***7.4.3 Medical Risk Pattern***

The temporal patterns can be treated as new features that can be used to approximate temporal concepts. In the case of the treatment of patient with cardiovascular failure one can define temporal concepts such as "Is the patient's SCD risk on low level?", "Is the patient's SCD risk on medium level?" or "Was high SCD risk detected?".

Temporal concepts defined for objects from a complex dynamical system and approximated by classifiers, can be treated as nodes of a graph called a behavioral graph, where connections between nodes represent temporal dependencies. Fig. 7.10 presents a behavioral graph for a single patient exhibiting a behavioral pattern of patient by analysis of the circulatory system failure caused by coronary heart disease. This graph has been created on the basis of observation of medical data sets and known factors for SCD risk stratification. In this behavioral graph, for example, connection between node "Medium SCD risk" and node "High SCD risk"



**Fig. 7.10** A behavioral graph of SCD risk by analyzing cardiovascular failure

indicates that after some period of progress in cardiovascular failure on medium level, a patient can change his behavior to the period, when progress in cardiovascular failure is high.

This behavioral graph is an example of risk pattern. If the patient matches the “Low SCD risk” concept in the first time window, “Medium SCD risk” in the following window, after which his state returned to the previous one, then the patient’s behavior doesn’t match this behavioral graph.

#### 7.4.4 Experiments with Medical Data

The next experiments were performed on data obtained from Second Department of Internal Medicine, Collegium Medicum, Jagiellonian University, Krakow, Poland. The data collection contains informations about 95 patients with stable coronary heart disease, collected between 2006 and 2009. It includes a detail description of clinical status (age, sex, diagnosis), coexistent diseases, pharmacological management, the laboratory tests outcomes (level of cholesterol, troponin I, LDL - low density lipoproteins), Holter ECG recordings (long term, 24-hour signals) and various Holter-based indices such as: ST-segment deviations, HRV, arrhythmias or QT dispersion. Sinus (normal) rhythm was observed in 73 patients, while 22 patients had permanent FA (atrial fibrillation). Two 24-hour Holter ECG recordings were performed using Aspel’s HolCARD 24W system. There was coronary angiography after first Holter ECG.

All data was imported to *Infobright Community Edition (ICE)* environment (see [46]). ICE is an open source software solution designed to deliver a scalable data warehouse optimized for analytic queries (data volumes up to 50 TB, market-leading data compression (from 10:1 to over 40:1)). Database schema was designed to store all information about patients, including supplementing the target database in the future. For further processing data have been imported into the RoughICE environment.

For the experiment, one table with 744 objects was formed. Each objects (row) contains information about the parameters of one patient with one hour of observation, being the average hourly values of observed parameters.

The experiments were performed in order to predict the behavioral pattern related to a high risk of SCD. This pattern was defined by medical experts on the base of well-known predictors of SCD. The evaluation of SCD risk includes: advanced age, male sex, coexisting diseases like DM (diabetes mellitus), HA (arterial hypertension), history of stroke, previous myocardial infarction, CRP (C-phase reaction protein) level, depressed LVEF (left ventricular ejection fraction), presence of arrhythmias and ischaemias, high heart rate, decreased HRV and HRT (heart rate turbulence). Taking into account simplicity of example model and temporal aspect of patterns, in this approach only few factors were chosen, such as HRV index: SDNN (standard deviation of NN intervals - normal to normal beat segments), average heart rate, ST interval decrease and QT segment changes. HRV parameter was calculated upon one hour period, though usually it is analyzed within 24 hour interval. Because of the lack of the appropriate data, such standard analyzes were not performed in this experiment.

We have applied the *train-and-test* method. However, because of the specificity of the analyzed data the method of data division differed slightly from the standard method. Namely, in each experiment the whole patient set was randomly divided into two groups (training group: 60% of patients and testing group: 40% of patients).

As a result of the above mentioned division of patients into training and testing ones, each of these parts made it possible to create time windows having duration of 2 time points (2 hours of patients observation) and sequences of such time windows (training part: approximately 400 time windows, testing part: approximately 270 sequences of time windows). Time windows created on the basis of training patients created a training table for a given experiment, while time windows sequences created on the basis of tested patients created a test table for the experiment.

In order to determine the standard deviation of the obtained results each experiment was repeated for 10 random divisions of the whole data set.

A single experiment was as follows (see also Figure7.11). First, for the training data the family of all time windows having duration of 2 time points were generated. Then, on the basis of temporal patterns proposed by experts, the behavioral graph from Figure 7.10 and the additional domain knowledge (represented by experts scripts in RoughICE - see [41] for more details) the temporal pattern tables were constructed for all concepts from the behavioral graph from the Figure 7.10. Then for all these tables a family of stratifying classifiers were generated that are able to classify objects (patients) to different concepts from the sequence of ordered layers. The first layer in this sequence represents objects which, without any doubt do not belong to the concept. The next layers in the sequence represent objects belonging to the concept more and more certainly. The last layer in this sequence represents objects certainly belonging to the concept (see [5] for more details). Next, a *complex classifier* was constructed on the basis of stratifying classifiers family that allow us to predict membership of a particular time window to various temporal concepts from the behavioral graph (see Figure7.11). The main idea of working

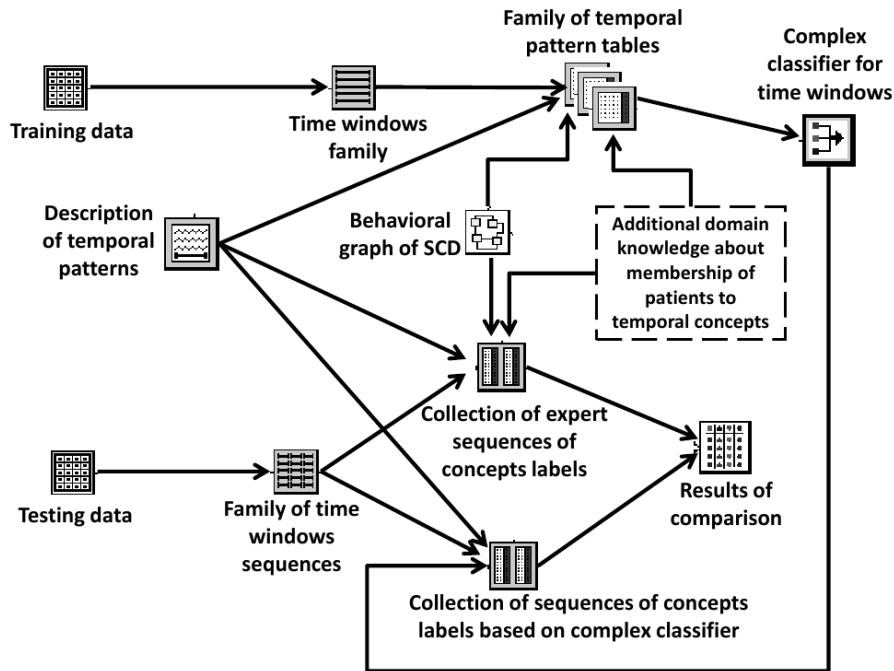


Fig. 7.11 A general scheme of experiments for the risk pattern of SCD

of such classifier is as follows. A given time window is classified to such temporal concept that a stratifying classifier corresponding to this concept classifies the time window to the top layer of all layers proposed by stratifying classifiers. Next, for the testing data the family of all sequences of time windows having duration of 2 time windows were generated (the length of every time window was 2 just as in the case of time windows for training data set). Then, for every sequence from this family, a sequence of labels of temporal concepts was generated in two different methods (see below). Such sequence of labels can be interpreted as a path of nodes from the behavioral graph. In this interpretation, the sequence of labels represents the answer if a given sequence matches the behavioral graph (behavioral pattern) from the figure 7.10.

The first method of the sequence of concepts labels generation is based on temporal patterns proposed by experts, the behavioral graph from Figure 7.10 and the additional domain (expert) knowledge about membership of patients to temporal concepts from behavioral graph. Therefore, this method we call as an *expert method* and the sequence of concepts labels generated with usage of this method we call as an *expert sequence of concepts labels*.

The second method of the sequence of concepts labels generation is based on the complex classifier generated for the training data. For a given sequence of time windows, the complex classifier has been used to generation of the concepts label for every time window separately. In this way, we obtain the sequence of concepts

labels, that can be also treated as a potential path of nodes from the behavioral graph. This method we call as a *classifier method* and the sequence of concepts labels generated with usage of this method we call as *a sequence of concepts labels based on classifier*.

Our method of presented approach evaluation is based on comparison of the expert and the classifier methods results. For a given sequence of time windows  $stw$ , the accuracy of identification of the sequence  $stw$  is computed in the following way:

- if the expert sequence of concepts labels computed for  $stw$  matches a path from the behavioral graph and a sequence of concepts labels based on the classifier also matches a path from the behavioral graph, the accuracy of identification of the sequence  $stw$  is equal 1,
- if the expert sequence of concepts labels computed for  $stw$  matches a path from the behavioral graph and a sequence of concepts labels based on classifier does not match a path from the behavioral graph, the accuracy of identification of the sequence  $stw$  is equal 0,
- if the expert sequence of concepts labels computed for  $stw$  does not match a path from the behavioral graph and a sequence of concepts labels based on classifier matches a path from the behavioral graph, the accuracy of identification of the sequence  $stw$  is equal 0,
- if the expert sequence of concepts labels computed for  $stw$  does not match a path from the behavioral graph and a sequence of concepts labels based on classifier does not match a path from the behavioral graph, the accuracy of identification of the sequence  $stw$  is equal 1.

The accuracy of identification of the whole family of time windows sequences is computed as an average value of accuracies computed for every sequence separately.

Table 7.3 shows the results of applying this algorithm for the concept related to the risk pattern of SCD. We present the accuracy, the coverage, the accuracy for positive examples (the expert sequence of concepts labels matches a path from the behavioral graph) and negative examples (the expert sequence of concepts labels computed does not match a path from the behavioral graph), the coverage for positive and negative examples and the real accuracy (Real accuracy = Accuracy \* Coverage). Together with the results we present a standard deviation of the obtained results.

Decision class	Accuracy	Coverage	Real accuracy
Yes (the high risk of SCD)	$0.953 \pm 0.048$	$1.0 \pm 0.000$	$0.953 \pm 0.048$
No (the low risk of SCD)	$0.971 \pm 0.010$	$1.0 \pm 0.000$	$0.971 \pm 0.010$
All classes (Yes + No)	$0.967 \pm 0.013$	$1.0 \pm 0.000$	$0.967 \pm 0.013$

**Table 7.3** Results of experiments for the risk pattern of SCD

Notice, that the accuracy of decision class Yes in medical statistics [3] is called a *sensitivity* (the proportion of those cases having a true positive test result of all pos-

itive cases tested), whereas the accuracy of decision class No is called a *specificity* (the proportion of true negatives of all the negative samples tested). We see both main parameters of our classifier (i.e., sensitivity and specificity) are sufficiently high.

Experimental results showed that the suggested method of behavioral patterns identification gives good results, also in the opinion of medical experts (compatible enough with the medical experience) and may be applied in medical practice as a supporting tool for medical diagnosis and treatment evaluation.

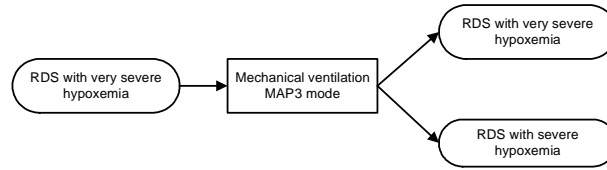
Finally, let us notice that the specific feature of the methods considered here is not only high accuracy (with low standard deviation) but also very high coverage (equal 1.0).

## 7.5 Methods of Automated Planning

In this section we present a method of automated planning behavior planning for complex object. This method also works on the basis of data sets and a domain knowledge represented by a concept ontology. A crucial novelty in the method proposed here, in comparison with the already existing ones, is the fact that performing actions according to plan depends on satisfying complex vague spatio-temporal conditions expressed in a natural language, which leads to the necessity of approximation of these conditions as complex concepts. Moreover, these conditions describe complex concept changes which should be reflected in the concept ontology.

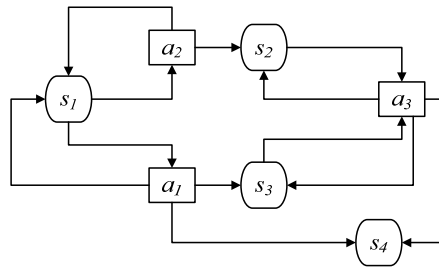
Behavior of unstructured complex objects (meaning those which may be treated as indivisible wholes) is modeled using the so-called *planning rules* being formulas of the type: *the state before performing an action*  $\rightarrow$  *action*  $\rightarrow$  *state 1 after performing an action* | ... | *state k after performing an action*, which are defined on the basis of data sets and a domain knowledge (see [5]). Each rule includes the description of the complex object state before applying the rule (that is, before performing an action), expressed in a language of features proposed by an expert, the name of the action (one of the actions specified by the expert which may be performed at a particular state), and the description of sequences of states which a complex object may turn into after applying the action mentioned above. It means that the application of such a rule gives indeterministic effects, i.e., after performing the same action the system may turn into different states.

Let us consider the planning rule from Fig. 7.12. This is the planning rule for treating RDS (respiratory distress syndrome) obtained from domain knowledge (see [5, 8]). The rule may be applied when RDS with very severe hypoxemia is present. The application of the rule consists in performing a medical action utilizing the respirator in the MAP3 mode (see [5, 8] for more medical details). As an effect of the application of this action at the following time point of observation (*e.g.*, the following morning) the patient's condition may remain unchanged or improve so as to reach the condition of RDS with severe hypoxemia.



**Fig. 7.12** The medical planning rule

All planning rules may be represented in a form of the so-called *planning graphs* whose nodes are state descriptions (occurring in predecessors and successors of planning rules) and action names occurring in planning rules. Let us consider planning graph from the Fig. 7.13, where the states are represented using ovals, and actions are represented using rectangles. Each link between the nodes of this graph represents a time dependencies. For example, the link between state  $s_1$  and action  $a_1$  tells us that in state  $s_1$  of the complex object action  $a_1$  may be performed, whereas the link between action  $a_1$  and state  $s_3$  means that after performing action  $a_1$  the state of the complex object may change to  $s_3$ . An example of a path in this graph is sequence  $(a_2, s_2, a_3, s_4)$  whereas path  $(s_1, a_2, s_2, a_3, s_3)$  is an exemplary plan in this graph.



**Fig. 7.13** An exemplary planning graph

In the graphical interpretation, solving the problem of automated planning is based on finding a path in the planning graph from the initial state to an expected final state. It is worth noticing that the conditions for performing an action (object states) are described by vague spatio-temporal complex concepts which are expressed in the natural language and require an approximation. For example, Fig. 7.14 presents a solution to the problem of finding a plan bringing state  $s_1$  to state  $s_4$  in the planning graph from Fig. 7.13.

For specific applications connected with the situation when it is expected that the proposed plan of a complex object behavior is to be strictly compatible with the determined experts' instructions (*e.g.*, the way of treatment in a specialist clinic is to be compatible with the treatment schemes used there), there has also been proposed an additional mechanism enabling to resolve the nondeterminism occurring in the



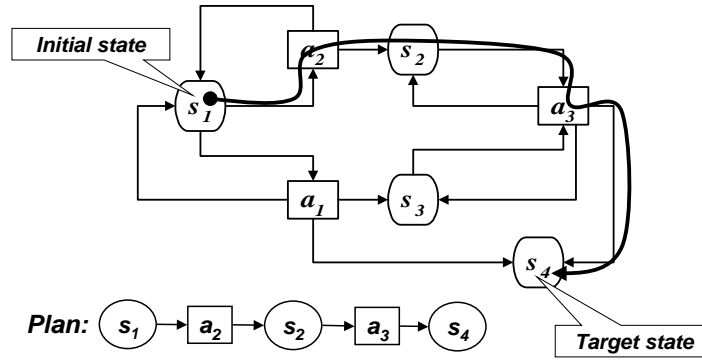


Fig. 7.14 The output for the planning problem

application of planning rules. This mechanism is an additional classifier based on data sets and domain knowledge. Such classifier (called a *resolving classifier*) suggests the action to be performed in a given state and show the state which is the result of the indicated action. A resolving classifier is a kind of stratifying classifier and is constructed on the basis of *resolving table* (see Fig. 7.15 and [5] for more details).

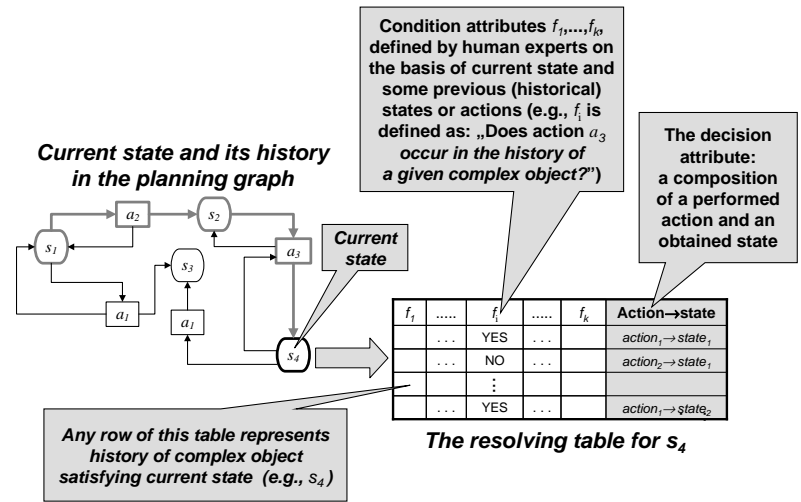


Fig. 7.15 The scheme of construction of the resolving table for a given state

### 7.5.1 Automated Planning for Structured Complex Objects

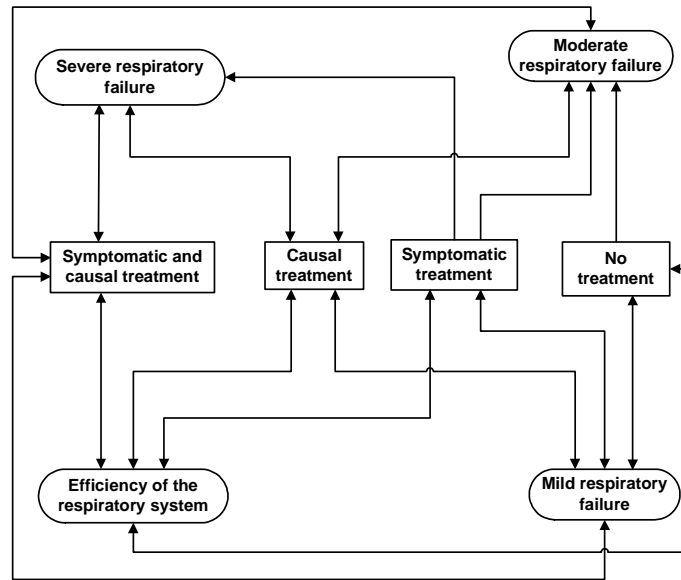
In planning the behavior of structured objects, an effective planning of the behaviors of all objects which are parts of these objects at the same time is not possible. Therefore, in such cases the behavior of all objects which are parts of a structured object is planned separately. However, this approach to planning of the behavior for a structured object requires a certain synchronization of the plans constructed for individual parts in such a way that these plans would not contradict each other and even complement each other in order to plan the best behavior for a structured object. For example, the treatment of illness  $A$  which is the resultant of two illnesses  $B$  and  $C$  requires such illnesses  $B$  and  $C$  treatment that the treatments of both illnesses would not be contradictory to each other, but even support and complement each other. For example, it may happen that in treating illness  $B$  a certain medicine  $M_1$  may be used which is usually an appropriate medicine but it may be applied only when illness  $C$  does not occur. Hence, the synchronization of both illnesses' treatment should exclude the application of medicine  $M_1$ . In a different situation it may happen that as a result of application of medicine  $M_2$  for illness  $C$  the treatment of illness  $B$  is safer, for instead of giving a certain strong medicine  $M_3$ , which has negative side effects, it is enough to give a safer medicine  $M_4$  which leads to the same improvement in the patient's condition as in the case of giving medicine  $M_3$ .

The automated planning method for unstructured objects has been generalized also in the case of planning of the behavior of structured objects (consisting of parts connected with one another by dependencies) (see [5]). The generalization is based on the fact that on the level of a structured object there is an additional planning graph defined where there are double-type nodes and directed edges between the nodes. The nodes of the first type describe vague features of states (meta-states) of the whole structured object, whereas the nodes of the second type concern complex actions (meta-actions) performed by the whole structured object (all its constituent parts) over a longer period of time (a time window). The edges between the nodes represent temporal dependencies between meta-states and meta-actions as well as meta-actions and meta-states.

In Fig. 7.16, we present an exemplary planning graph for a structured object, that is a group of four diseases: sepsis, Ureaplasma, RDS and PDA, related to the planning of the treatment of the infant during the respiratory failure. This graph was created on the basis of observation of medical data sets and with support of human experts (see [5, 8] for more medical details).

As we see, there are two kinds of nodes in the planning graph for structured object, namely, *meta states nodes* (denoted by ovals) that represent the current state of a structured object specified as complex concepts by a human expert in natural language, and *meta action nodes* (denoted by rectangles) that represent actions defined for structured objects.

The major difference between the planning graph for the unstructured complex object and the planning graph for the structured object is that in the last one instead of actions performed at a single time point meta-actions occur which are performed over a longer period of time, that is, a time window.



**Fig. 7.16** A planning graph for the treatment of infants during the respiratory failure

At the beginning of planning for a structured object, we have to identify the current meta state of this object. Any meta state node from a planning graph for structured objects can be treated as a complex spatio-temporal concept that is specified by a human expert in natural language. Such concepts can be approximated by classifiers using data sets and domain knowledge accumulated for a given complex dynamical system. Similarly to states from the planning graph for unstructured complex objects, any state from the planning graph for structured objects can be approximated as a temporal concept for structured object using method from Section 7.3. As a result, it is possible to recognize the initial state at the beginning of planning for a particular structured object.

Similarly to the previous case of unstructured objects, planning of a structured object behavior is based on finding a path in a planning graph from the initial meta-state to the expected final meta-state; and, at the same time, each meta-action occurring in such a path must be planned separately on the level of each constituent part of the structured object. In other words, it should be planned what actions each part of a structured object must perform in order for the whole structured object to be able to perform the meta-action which has been planned. For example, in the case of the treatment of infants with respiratory failure, if the infant is suffering from severe respiratory failure, we try to change the patient status using some methods of treatment to change its status to moderate or mild respiratory failure. However, any meta action from such constructed path should be checked on the lower level, i.e., on the level of any part of the structured object separately, if such action can be realized in practice in case of particular part of this structured object. In other words, it means

that for any part of the structured object the sequence of action should be planned in order to obtain meta-action on the level of the structured object.

The plan of execution of a single meta-action, which consists of short plans which execute this meta-action on the levels of individual parts of the structured object, is called a *g-plan* (see [5]). The *g-plan* is, thus, a family of plans assigned to be executed for all parts of the established structured object.

Let us notice that determining the plan for a structured object requires not only determining sets of plans for all parts of the structured object but also synchronizing them in time. In practise, all constructed plans for objects (parts) belonging to a given structured object should be compatible. Therefore, during planning a meta action for a structured object, we use a special tool for verifying the compatibility of plans generated for all members of a structured object. This verification can be performed by using some special decision rules that we call *elimination rules*. Such rules make it possible to eliminate combination of plans that are not compatible relative to domain knowledge. This is possible because elimination rules describe all important dependencies between plans that are joined together. If any combination of plans is not consistent with any elimination rule, then it is eliminated. A set of elimination rules can be specified by human experts or can be computed from data sets. In both of these cases, we need a set of attributes (features) defined for a single plan that are used for explaining elimination rules. Such attributes are specified by human experts on the basis of domain knowledge and they describe some important features of the plan (generated for some part of structured object) with respect to proper joining a plan with plans generated for other parts of structured object. These features are used as a set of attributes in the special table that we call an *elimination table*. Any row of an elimination table represents information about features of plans assigned for structured objects from the training data. For example, the respiratory failure may be treated as a result of four following diseases: RDS, PDA, sepsis and Ureaplasma. Therefore, treating respiratory failure requires simultaneous treatment of all of these diseases. This means that the treatment plan of respiratory failure comes into existence by joining the treatment plans for diseases RDS, PDA, sepsis and Ureaplasma, and at the same time the synchronization of the plans is very important. In this chapter, one of the synchronizing tools for this type of plans is the elimination table. In constructing the elimination table for treatment of respiratory failure, patterns describing the properties of the joint plans are needed. Moreover, planning graphs for all four diseases are necessary. In Fig. 7.17 the planning graph for RDS treatment is shown. In a very similar way the features of treatment plans for PDA, sepsis and Ureaplasma diseases may be defined.

On the basis of the elimination table a set of elimination rules can be computed that can be used to eliminate inappropriate plan arrangements for individual parts of the structured object. So, the set of elimination rules can be used as a filter of inconsistent combinations of plans generated for members of groups. Any combination of plans is eliminated when there exists an elimination rule that is not supported by features of a combination while the combination matches a predecessor of this rule. In other words, a combination of plans is eliminated when the combination matches

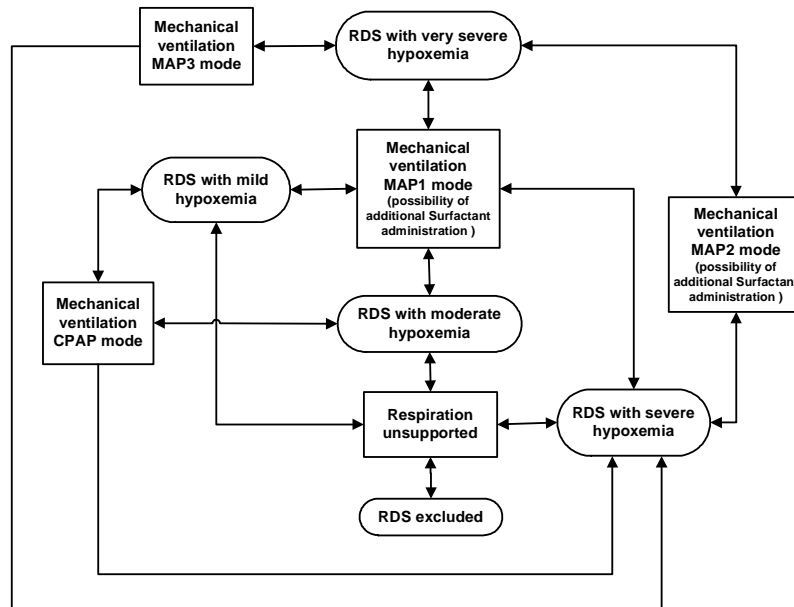


Fig. 7.17 A planning graph for the treatment of infants during the RDS

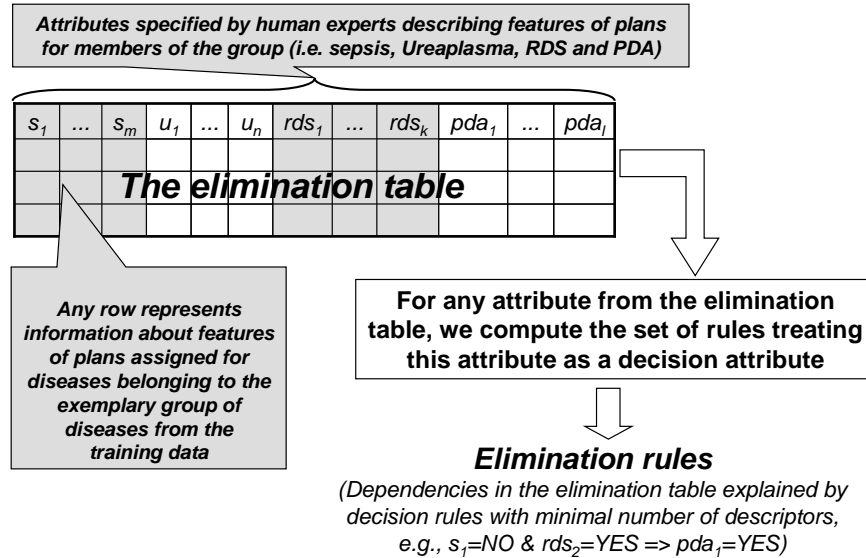
to the predecessor of some elimination rule and does not match the successor of a rule.

Fig. 7.18 shows the scheme of elimination rules of not-acceptable g-plans constructed in the case of the treatment of respiratory failure, which is a result of the four following diseases: sepsis, Ureaplasma, RDS and PDA.

As we see, for any attribute from the elimination table, we compute the set of rules with minimal number of descriptors treating this attribute as a decision attribute. In this way, we obtain a set of dependencies in the elimination table explained by decision rules. In practice, it is necessary to filter elimination rules to remove the rules with low support because such rules can be too strongly matched to the training data.

On the basis of the set of elimination rules an *elimination classifier* may be constructed that enable elimination of inappropriate plan arrangements for individual parts of the structured object.

If the combination of plans for parts of the structured object is consistent (it was not eliminated by elimination rules), we should check if the execution of this combination allows us to realize the expected meta action from the level of structured objects. This can be done by a special classifier constructed for a table called a *meta action table*. The structure of a meta action table is similar to the structure of an elimination table, i.e., attributes are defined by human experts, where rows represent information about features of plans assigned for parts of exemplary structured objects from the training data. In addition, we add to this table a decision attribute. Values of such decision attributes represent names of meta actions which are real-



**Fig. 7.18** The scheme of construction of elimination rules for group of four diseases: sepsis, Ureaplasma, RDS and PDA

ized as an effect of the execution of plans described in the current row of a training table.

The classifier computed for an action table makes it possible to predict the name of a meta action for a given combination of plans from the level of parts of a structured object. The last step is the selection of combinations of plans that makes it possible to obtain a target meta action with respect to a structured object (see Fig. 7.19).

After planning the selected meta action from the path of actions from the planning graph (for a structured object), the system begins the planning of the next meta action from this path. The planning is stopped, when the planning of the last meta action from this path is finished.

### 7.5.2 Estimation of the Similarity Between Plans

In construction and application of classifiers approximating complex spatio-temporal concepts, there may appear a need to construct, with a great support of the domain knowledge, a similarity relation of two elements of similar type, such as complex objects, complex object states, or plans generated for complex objects. Hence, in the paper [5] a new method of similarity relation approximation has been proposed which is based on the use of data sets and a domain knowledge expressed mainly in the form of a concept ontology. We apply this method, among other things, to verify

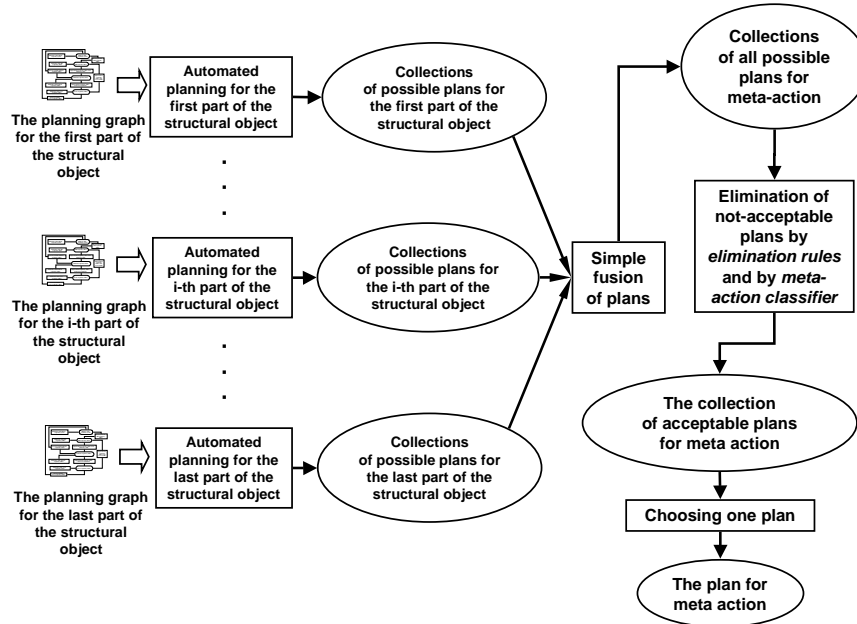


Fig. 7.19 The scheme of meta action planning

automated planning methods, that is, to compare the plan generated automatically with the plan suggested by experts from a given domain.

The problem of inducing classifiers for similarity relations is one of the challenging problems in data mining and knowledge discovery (see bibliography from [5]). The existing methods are based on building models for similarity functions using simple strategies for fusion of local similarities. The optimization of the assumed parameterized similarity formula is performed by tuning parameters relative to local similarities and their fusion. For instance, if we want to compare two medical plans of treatments, e.g., one plan generated automatically by our computer system and another one proposed by medical expert, we need a tool to estimate the similarity. This problem can be solved by introducing a function measuring the similarity between medical plans. For example, in the case of our medical data, a formula is used to compute a similarity between two plans as the arithmetic mean of similarity between all corresponding pairs of actions (nodes) from both plans, where the similarity for the single corresponding pair of actions is defined by a consistence measure of medicines and medical procedures comprised in these actions. For example, let  $M = \{m_1, \dots, m_k\}$  be a set consisting of  $k$  medicines. Let us assume that actions in medical plans are specified by subsets of  $M$ . Hence, any medical plan  $P$  determines a sequence of actions  $\mathcal{A}(P) = (A_1, \dots, A_n)$ , where  $A_i \subseteq M$  for  $i = 1, \dots, n$  and  $n$  is the number of actions in  $P$ . In our example, the similarity between plans is defined by a similarity function  $Sim$  established on pairs of medical plans  $(P_1, P_2)$  (of the same

length) with the sequences of actions  $\mathcal{A}(P_1) = (A_1, \dots, A_n)$  and  $\mathcal{A}(P_2) = (B_1, \dots, B_n)$ , respectively as follows

$$Sim(P_1, P_2) = \frac{1}{n} \sum_{i=1}^n \frac{|A_i \cap B_i| + |M \setminus (A_i \cup B_i)|}{|M|}.$$

However, such an approach seems to be very abstract and ad hoc, because it does not take into account any deeper knowledge about the similarity of plans, e.g., domain knowledge. Whereas, the similarity relations for real-life problems are usually more complex objects, i.e., their construction from local similarities cannot be obtained by simple fusion functions. Hence, such similarity relations cannot be approximated with the satisfactory quality by employing the existing simple strategies. For this reason we treat this similarity measure, *Sim*, only as an example and do not take into account in our further research (and in our proposed method). Whereas, to support the process of similarity relation approximation, we propose to use domain knowledge represented by concept ontology expressed in natural language. The ontology consists of concepts used by expert in his explanation of similarity and dissimilarity cases. Approximation of the ontology makes it possible to obtain some relevant concepts for approximation of the similarity relation.

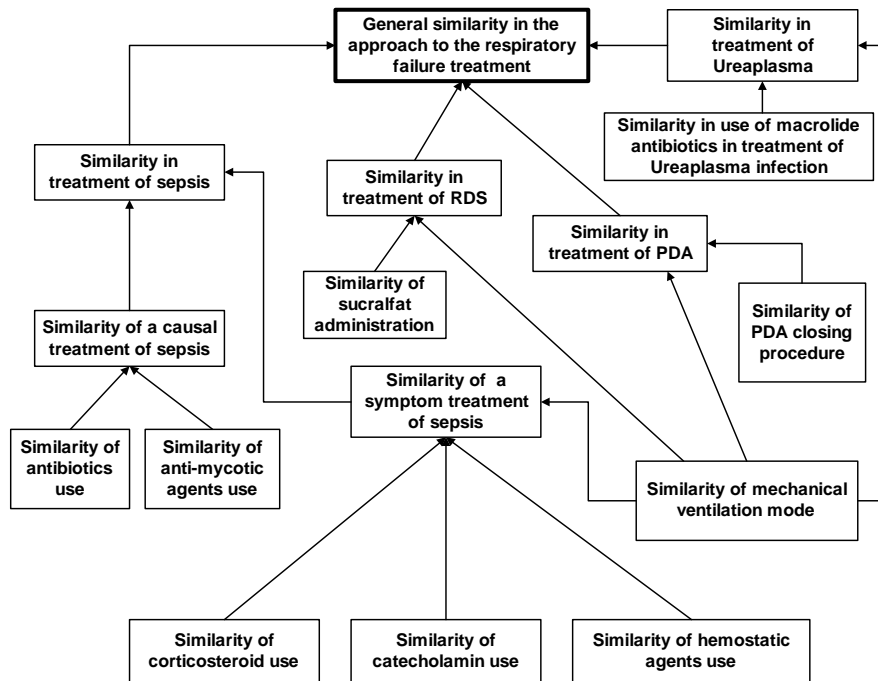
### 7.5.3 *Ontology of the Similarity Between Plans*

According to the domain knowledge, it is quite common, that there are many aspects of similarity between plans. For example, in case of comparison of medical plans used for the treatment of infants with respiratory failure, we should take into consideration, e.g., the similarity of the antibiotics use, the ventilation mode and the similarity of PDA closing (see [5] for more medical details). Moreover, every aspect of the similarity should be understood in a different way. For example, in estimation of the similarity in the antibiotic treatment, it should be evaluated the kind of antibiotic, as well as the time of administration. Therefore, it is necessary to investigate and take into account all incompatibilities of the antibiotic use between corresponding pairs of nodes from both plans. Excessive doses are rather acceptable (based on expert knowledge), whilst the lack of medicine (if it is necessary) should be taken as a very serious mistake. In such situation, the difference in our assessment is estimated as very significant. A bit different interpretation of similarity should be used in case of the ventilation. As in antibiotic use, we investigate all incompatibilities of the ventilation mode between corresponding pairs of nodes from both plans. However, sometimes, according to expert knowledge, we simplified our assessments, e.g., respiration unsupported and CPAP are estimated as similar for more medical details). More complicated situation is present if we want to judge the similarity in treatment of PDA. We have to assign the ventilation mode, as well as the similarity of PDA closing procedure. In summary, any aspect of the similarity between plans should be taken into account in the specific way and the domain knowledge



is necessary for joining all these similarities (obtained for all aspects). Therefore, the similarity between plans should be assigned on the basis of a special ontology specified in a dialog with human experts. Such ontology we call *similarity ontology*. Using such similarity ontology we developed methods for inducing classifiers predicting the similarity between two plans (generated automatically and proposed by human experts).

In the chapter, we assume that each similarity ontology between plans has a tree structure. The root of this tree is always one concept representing general similarity between plans. In each similarity ontology there may exist concepts of two-way type. In this chapter, the concepts of the first type will be called *internal concepts* of ontology. They are characterized by the fact that they depend on other ontology concepts. The concept of the second type will be called *input concepts* of ontology (in other words the concepts of the lowest ontology level). The input concepts are characterized by the fact that they do not depend on other ontology concepts. Fig. 7.20 shows an exemplary ontology of similarity between plans of the treatment of newborn infants with the respiratory failure. This ontology has been provided by human experts. However, it is also possible to present some other versions of such ontology, instead of that presented above, according to opinions of some other group of human experts.



**Fig. 7.20** An exemplary ontology of similarity between plans of the treatment of newborn infants with respiratory failure

### 7.5.4 Similarity Classifier

Using the similarity ontology (*e.g.*, the ontology presented in Fig. 7.20), we developed methods for inducing classifiers predicting the similarity between two plans (generated automatically and proposed by human experts).

The diagram shows a table with the following structure:

- Columns:** The first column is labeled "Pairs of plans". The subsequent columns are labeled  $C_1$ , ...,  $C_k$ , and  $C$ . Callouts indicate that "Condition columns represent concepts  $C_1, \dots, C_k$  from the similarity ontology" and "The decision column represents the concept  $C$ ".
- Rows:** Each row represents a pair of plans, labeled from (pa1, pe1) to (pa5, pe5). A callout states: "Row corresponds to a pair of plans: the first generated automatically and the second proposed by experts".
- Values:** The cells contain numerical values representing similarity scores. For example, the first row has values 0.2, ..., 0.3, 0.1.

Pairs of plans	$C_1$	.....	$C_k$	$C$
(pa1, pe1)	0.2	.....	0.3	0.1
(pa2, pe2)	0.4	.....	0.5	0.5
(pa3, pe3)	0.2	.....	0.8	0.8
(pa4, pe4)	0.8	.....	0.1	0.2
(pa5, pe5)	0.3	.....	0.2	0.6

Fig. 7.21 The scheme of the similarity table of plans

The method for construction of such classifier can be based on a *similarity table of plans*. The similarity table of plans is the decision table which may be constructed for any concept from the similarity ontology. The similarity table is created in order to approximate a concept for which the table has been constructed. The approximation of the concept takes place with the help of classifiers generated for the similarity table. However, because of the fact that in the similarity ontology there occur two types of concepts (internal and input), there are also two types of similarity tables. Similarity tables of the first type are constructed for internal concepts, whereas the tables of the second type are constructed for input concepts.

Similarity tables for internal concepts of similarity ontology are constructed for a certain fragment of similarity ontology which consists of a concept of this ontology and concepts on which this concept depends. In the case of ontology from Fig. 7.20 it may be for instance the concept *Similarity of a symptom treatment of sepsis* and concepts *Similarity of corticosteroid use*, *Similarity of catecholamin use* and *Similarity of hemostatic agents use*. To simplify further discussion let us assume that it is the concept  $C$  that depends in the similarity ontology on the concepts  $C_1, \dots, C_k$ . The aim of constructing a similarity table is approximation of concept  $C$  using concepts  $C_1, \dots, C_k$  (see Fig. 7.21). Condition columns of such similarity table represent concepts  $C_1, \dots, C_k$ . Any row corresponds to a pair of plans: generated automatically and proposed by experts. Values of all attributes have been provided by experts from the set  $\{0.0, 0.1, \dots, 0.9, 1.0\}$ . Finally, the decision column represents the concept  $C$ .

The stratifying classifier computed for a similarity table (called a *similarity classifier*) can be used to determine the similarity between plans (generated by our meth-

ods of automated planning and plans proposed by human experts) relatively to a given internal concept  $C$ .

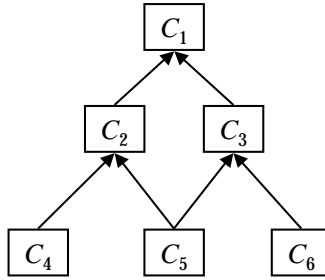
Such stratifying classifiers may be constructed for all concepts from the similarity ontology which depend, in this ontology, on other concepts. However, we also need stratifying classifiers for input concepts of ontology, that is, those lying on the lowest level of the ontology. Hence, they are the concepts which do not depend on other concepts in this ontology. To approximate them we do not use other ontology concepts but we apply the features of comparable plans which are expressed in the form of patterns defined in the special language. Obviously, such types of patterns are also concepts determined in the set of pairs of plans. However, they are usually not placed in the similarity ontology between plans. Therefore, approximation tables of input concepts of the similarity ontology should be treated as a specific type of similarity table.

Let us notice that the similarity table defined above is constructed in the way that concept  $C$  is approximated on the basis of the features of both plans corresponding to a given object from the set  $U$ .

It is worth noticing that for approximation of complex concepts from the similarity ontology one can use also features (attributes) describing relations between plans. Such features are formulated in a natural language using special questions about both plans. Examples of such questions are: *Were antibiotics used simultaneously in both plans?*, *Was the average difference between mechanical ventilation mode in both plans significant?*. However, it requires a simple extension of the language.

Classifiers constructed for similarity tables corresponding to all concepts from the similarity ontology may be used to construct a complex classifier which gives the general similarity between plans (represented by the concept lying in the root of the similarity ontology). We provide an example of how such a classifier works. Let us assume that there is a certain similarity ontology between pairs of plans in which there occur six following concepts:  $C_1$ ,  $C_2$ ,  $C_3$ ,  $C_4$ ,  $C_5$  and  $C_6$ . The concept  $C_1$  depends on concepts  $C_2$  and  $C_3$ , the concept  $C_2$  depends on concepts  $C_4$  and  $C_5$ , and the concept  $C_3$  depends on concepts  $C_5$  and  $C_6$ . In this ontology concept  $C_1$  is the concept of general similarity between plans, whereas concepts  $C_4$ ,  $C_5$  and  $C_6$  are input concepts of the similarity ontology (see Fig. 7.22).

Firstly, we construct similarity tables for concepts  $C_4$ ,  $C_5$ ,  $C_6$  and stratifying classifiers  $\mu_{C_4}$ ,  $\mu_{C_5}$ ,  $\mu_{C_6}$  corresponding to them. Let us also assume that there are given stratifying classifiers  $\mu_{C_1}$ ,  $\mu_{C_2}$ ,  $\mu_{C_3}$  which were constructed for similarity tables which correspond to concepts  $C_1$ ,  $C_2$  and  $C_3$ . Tested object  $u = (p_1, p_2)$  which is a pair of compared plans is classified to the layer of concept  $C$  corresponding to it in the following way. At the beginning, the object  $u$  is classified by classifiers  $\mu_{C_4}$ ,  $\mu_{C_5}$  and  $\mu_{C_6}$ . This way we obtain values  $\mu_{C_4}(u)$ ,  $\mu_{C_5}(u)$  and  $\mu_{C_6}(u)$ . Next, values  $\mu_{C_4}(u)$  and  $\mu_{C_5}(u)$  are used as the values of conditional attributes in the similarity table constructed for concept  $C_2$ . Thus, the object  $u$  may be classified by classifier  $\mu_{C_2}$ , which gives us value  $\mu_{C_2}(u)$ . At the same time, values  $\mu_{C_5}(u)$  and  $\mu_{C_6}(u)$  are used as the values of conditional attributes in the similarity table constructed for concept  $C_3$ . It gives the possibility to classify object  $u$  by classifier  $\mu_{C_3}$  and obtain



**Fig. 7.22** The scheme of a simple similarity ontology

value  $\mu_{C_3}(u)$ . Finally, values  $\mu_{C_2}(u)$  and  $\mu_{C_3}(u)$  are used as the values of conditional attributes of the similarity table constructed for concept  $C_1$ . Thus, the object  $u$  may be classified by classifier  $\mu_{C_1}$  to layer  $\mu_{C_1}(u)$ .

The complex classifier described above can be used to determine the general similarity between plans generated by our methods of automated planning and plans proposed by human experts, *e.g.*, during the real-life clinical treatment (see Section 7.5.5).

### 7.5.5 Experiments with Medical Data

To verify the effectiveness of presented in this chapter methods of automated planning, we have implemented the algorithms in the RoughICE system (see [41]).

It should be emphasized that, in general, automated planning of treatment is a very difficult and complicated task because it requires extensive medical knowledge combined with sensor information about the state of a patient. Even so, the proposed approach makes it possible to obtain quite satisfactory results in the short-term planning of treatment of infants with respiratory failure. The reason is that medical data sets have been accurately prepared for purposes of our experiments using the medical knowledge. For example, the collection of medical actions, that are usually used during the treatment of infants with respiratory failure, has been divided into a few groups of similar actions (for example: antibiotics, anti-mycotic agents, mechanical ventilation, catecholamines, corticosteroids, hemostatic agents). It is very helpful in the prediction of actions because the number of actions is significantly decreased.

The experiments have been performed on the medical data sets obtained from Neonatal Intensive Care Unit, First Department of Pediatrics, Polish-American Institute of Pediatrics, Collegium Medicum, Jagiellonian University, Krakow, Poland (see [5, 7, 9, 10]). We used one data table, that consists of 11099 objects. Each object of this table describes parameters of one patient in single time point. There were prepared 7022 situations on the basis of this data table, where the plan of treatment has been proposed by human experts during the real-life clinical treatment.

We have applied the *train-and-test* method. In each experiment the whole set of patients was randomly divided into two groups (training and tested one). Each of these groups allowed creating approximately 4000 time windows which have duration of 7 time points. Time windows created on the basis of patients from the training part created a training table for a given experiment (when plans of treatment have been assigned), whereas time windows created on the basis of patients from the tested part created a test table for the experiment (when plans have been generated by automated method and expert plans are known in order to compare both plans)

In the discussed experiments, the distance between time points recorded for a specific patient was constant (one day). In a single experiment concerning a patient's treatment, a 7-point sequence of time points was used. In terms of planning the treatment each such sequence may be written as  $s_1, a_1, s_2, a_2, s_3, a_3, s_4, a_4, s_5, a_5, s_6, a_6, s_7$ , where  $s_i$  (for  $i = 1, \dots, 7$ ) is a patient state and  $a_i$  (for  $i = 1, \dots, 6$ ) is a complex medical action performed in the state  $s_i$ . The first part of the above sequence of states and actions, that is, from state  $s_1$  to state  $s_3$ , was used by the method of automated planning as the input information (corresponding to the values of conditional attributes in the classic approach to constructing classifiers). The remaining actions and states were automatically generated to create plan  $(s_3, a'_3, s'_4, a'_4, s'_5, a'_5, s'_6, a'_6, s'_7)$ . This plan may be treated as a certain type of a complex decision value. Verification of the quality of the generated plan consisted in comparing plan  $(s_3, a'_3, s'_4, a'_4, s'_5, a'_5, s'_6, a'_6, s'_7)$  with plan  $(s_3, a_3, s_4, a_4, s_5, a_5, s_6, a_6, s_7)$ . It is worth adding that a single complex action concerned one time point, meta action concerned two time points and a single experiment consisted in planning two meta actions. Hence, in a single experiment four actions were planned (patient's treatment for four days). In other words, at the beginning of the automated planning procedure the information about the patient's state in the last three days of his hospitalization was used ( $s_1, s_2, s_3$ ) together with the information about complex medical actions undertaken one or two days before ( $a_1, a_2$ ). The generated plan included information about a suggested complex medical action on a given day of hospitalization ( $a'_3$ ), information about actions which should be undertaken in the three following days of hospitalization ( $a'_4, a'_5, a'_6$ ) and information about the patient's state anticipated as a result of the planned treatment in the four following days of hospitalization ( $s'_4, s'_5, s'_6, s'_7$ ).

As a measure of planning success (or failure) in our experiments, we use the special classifier that can predict the similarity between two plans as a number between 0.0 (very low similarity between two plans) and 1.0 (very high similarity between two plans) (see Section 7.5.4). We use this classifier to determine the similarity between plans generated by our methods of automated planning and plans proposed by human experts during the real-life clinical treatment. In order to determine the standard deviation of the obtained results each experiment was repeated for 10 random divisions of the whole data set.

The average similarity between plans for all tested situations was **0.802**. The corresponding standard deviations was **0.041**. The coverage of tested situation by generated plans was **0.846** with standard deviation **0.018**.

Due to the fact that the average similarity is not too high (less than 0.9) and the standard deviation is relatively high for our algorithm, we present also the distri-

Intervals	Average percent of plans	Average similarity of plans
[0.0, 0.2]	<b>12.1%</b> $\pm$ 4.5%	<b>0.139</b> $\pm$ 0.002
(0.2, 0.4]	<b>6.2%</b> $\pm$ 1.5%	<b>0.349</b> $\pm$ 0.003
(0.4, 0.6]	<b>7.1%</b> $\pm$ 1.7%	<b>0.563</b> $\pm$ 0.002
(0.6, 0.8]	5.8% $\pm$ 0.9%	0.773 $\pm$ 0.004
(0.8, 1.0]	68.9% $\pm$ 5.6%	0.987 $\pm$ 0.002

**Table 7.4** The average percent of plans belonging to the specified interval and the average similarity of plans in this interval

bution of the results. We describe results in such a way that we present how many generated plans belong to the specified interval of similarity. For this reason we divided interval [0.0, 1.0] into 5 equal intervals, i.e., [0.0, 0.2], [0.2, 0.4], [0.4, 0.6], [0.6, 0.8] and [0.8, 1.0]. Table 7.4 shows the average percent of the plans belonging to the specified interval and the average similarity of plans in this interval.

It is easy to see that some group of plans generated automatically is not enough similar to the plans proposed by the experts. If we assume that inadequate similarity is lower than 0.6, in this group we found about 25% of all plans (see Table 7.4). To explain this issue, we should observe more carefully plans, which are incompatible with the proposals prepared by experts. In practice, the main medical actions influencing the similarity of plans in accordance with ontology of the similarity from Fig. 7.20 are mechanical ventilation, antibiotics, anti-mycotic agents and macrolide antibiotics. Therefore, it may be interesting how the treatment similarity changed in the range of applying these actions in the individual intervals of similarity between the plans.

On Fig. 7.23 we can see that a significant incompatibility of treatment plans most often concerns mechanical ventilation and perhaps antibiotic therapy - the situation when a patient develops a sudden and severe infection (*e.g.*, sepsis). Such circumstances cause rapid exacerbation of respiratory failure are required higher level of mechanical ventilation and immediate antibiotic treatment. For example, although microbiological confirmation of current infection is achieved after 2-3 days, physician starts treatment after first symptoms of suspected disease and often intensify mechanical ventilation mode. It would seem that the algorithms of automated planning presented in this chapter may imitate the strategy of treatment described above. Unfortunately, in practice, these algorithms are not able to learn this strategy for a lot of information because they were not introduced to the base records or were introduced with delay. For instance, hemoglobin saturation which is measured for the whole time, as the dynamic marker of patients respiratory status, was not found in the data, whilst results of arterial blood gases were introduced irregularly, with many missing values. So, the technical limitation of the current data collection lead to the intensive work modifying and extending both, the equipment and software, served for gathering clinical data. It may be expected that in several years the auto-

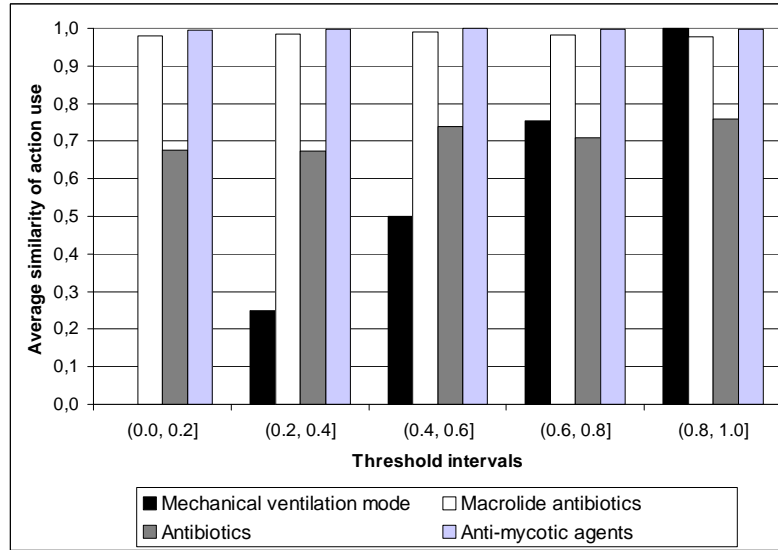


Fig. 7.23 The average similarity of plans in the specified interval for medical actions

mated planning algorithms, described in this chapter, will achieve much better and useful results.

A separate problem is a relatively low coverage of the algorithms described in this chapter which equals averagely 0.846. Such a low coverage results from the specificity of the automated planning method used which synchronizes the treatment of four diseases (RDS, PDA, sepsis and Ureaplasma). We may identify two reasons of a low coverage. Firstly, because of data shortage the algorithm in many situations may not synchronize the treatment of the above mentioned diseases. It happens this way because each proposed comparison of plans may be debatable in terms of the knowledge gathered in the system. Therefore, in these cases the system does not suggest any treatment plan and says *I do not know*. The second reason for low coverage is the fact that the automated planning method used requires application of a complex classifier which consists of many classifiers of lesser complexity. Putting these classifiers together often causes the effect of decreasing the complex classifier coverage. For instance, let us assume that making decision for tested object  $u$  requires application of complex classifier  $\mu$ , which consists of two classifiers  $\mu_1$  and  $\mu_2$ . We apply classifier  $\mu_1$  directly to  $u$ , whereas classifier  $\mu_2$  is applied to the results of classification of classifier  $\mu_1$ . In other words, to make classifier  $\mu_2$  work for a given tested object  $u$  we need value  $\mu_1(u)$ . Let us assume that the coverage for classifiers  $\mu_1$  and  $\mu_2$  equals respectively 0.94 and 0.95. Hence, the coverage of classifier  $\mu$  is equal  $0.94 \cdot 0.95 = 0.893$ , that is the coverage of classifier  $\mu$  is smaller than the coverage of classifier  $\mu_1$  as well as the coverage of classifier  $\mu_2$ .

In summation, we conclude that experimental results showed that the proposed automated planning method gives good results, also in the opinion of medical ex-

perts (compatible enough with the plans suggested by the experts), and may be applied in medical practice as a supporting tool for planning the treatment of infants suffering from respiratory failure.

## 7.6 Conclusion

The aim of this chapter was to present new methods of approximating complex concepts on the basis of experimental data and domain knowledge which is mainly represented using concept ontology.

At the beginning of the chapter a method of spatial complex concepts approximation were presented (see Section 7.2). Next, in Sections 7.3 we presented the method of approximate spatio-temporal complex concepts. In the further part of the chapter, the method of behavioral pattern identification was overviewed (see Section 7.4). Finally, in Section 7.5, we described the method of automated planning of behavior of complex objects when the states of objects are represented by spatio-temporal concepts which require an approximation.

We have also described the results of computer experiments conducted on real-life data sets which were obtained from the road traffic simulator (see [43]) and on medical data sets which were made available by Neonatal Intensive Care Unit, First Department of Pediatrics, Polish-American Institute of Pediatrics, Collegium Medicum, Jagiellonian University, Krakow, Poland and by Second Department of Internal Medicine, Collegium Medicum, Jagiellonian University, Krakow, Poland.

In light of theoretical discourse and the results of computer experiments presented in the chapter the following conclusions may be drawn:

1. The method of approximation of complex spatial concepts, described in the chapter (see Section 7.2), with the help of approximate reasoning schemes (AR-schemes) leads to better results than the classical methods based on decision rules induced directly from sensor data because the quality of classifier classification based on AR-schemes is higher than the quality of classification obtained by classifiers based on decision rules, particularly for small decision classes representing atypical cases in the recognition of which we are most interested in, *e.g.*, a dangerous driving vehicle on a highway. Moreover, for larger data sets, the time of constructing classifiers based on AR-schemes is much shorter than the time of inducing classifiers based on decision rules, and the structure of classifiers based on AR-schemes is less complex than the structure of classifiers based on decision rules. It is also worth mentioning that the classifiers based on AR-schemes are more robust (stable or tolerant) when it comes to changes in training data sets serving the construction of classifiers, that is, a classifier based on AR-schemes, constructed for one data set, often proves itself good for another data set. For example, a classifier constructed for data generated from the traffic simulator with one simulation scenario proves itself useful in classification of objects generated by the simulator with the use of another simulation scenario.



2. The methodology of modeling complex object behavior with the use of behavioral graphs of these objects, proposed in the chapter (see Section 7.4), is a convenient and effective tool for identifying behavioral or risk patterns of complex objects. On the one hand this methodology, enables to represent concepts on a high abstraction level, and on the other hand, owing to the use of a domain knowledge, it enables to approximate these concepts on the basis of sensor data and using a domain knowledge.
3. The methods of automated planning of complex object behavior proposed in the chapter facilitate an effective planning of behavior of objects whose states are defined in a natural language using vague spatio-temporal conditions (see Section 7.5). The authenticity of conditions of this type is usually not possible to be verified on the basis of a simple analysis of available information about the object and that is why these conditions must be treated as spatio-temporal complex concepts and their approximation requires methods described in this chapter which are based on data sets and domain knowledge.

In summation, it may be concluded that in executing real-life projects related to the construction of the intelligent systems supporting decision-making, apart from data sets it is necessary to apply domain knowledge. Without its application successful execution of many such projects becomes extremely difficult or impossible. On the other hand, appropriate space must be found for the automated methods of classifier construction wherever it is feasible. It means, thus, finding a certain type of “the golden mean” to apply appropriate proportions in domain knowledge usage and automated methods of data analysis. Certainly, it will determine the success or failure of many projects.

## Acknowledgement

This work was supported by the grant N N516 077837 from the Ministry of Science and Higher Education of the Republic of Poland, the Polish National Science Centre (NCN) grant 2011/01/B/ST6/03867 and by the Polish National Centre for Research and Development (NCBiR) grant No. SP/I/1/77065/10 in frame of the the strategic scientific research and experimental development program: “Interdisciplinary System for Interactive Scientific and Scientific-Technical Information”.

## References

1. A. Skowron J. Stepaniuk, R.W.S.: Modeling rough granular computing based on approximation spaces. *Information Sciences* **184**, 20–43 (2012)
2. van der Aalst, W.M.P.: *Process Mining: Discovery, Conformance and Enhancement of Business Processes*. Springer-Verlag, Berlin (2011)
3. Altman, D.G.: *Practical Statistics for Medical Research*. Chapman and Hall/CRC, London (1997)

4. Bar-Yam, Y.: Dynamics of Complex Systems. Addison Wesley, New York, USA (1997)
5. Bazan, J.G.: Hierarchical classifiers for complex spatio-temporal concepts. Transactions on Rough Sets **5390**(IX), 474–750 (2008)
6. Bazan, J.G.: Rough sets and granular computing in behavioral pattern identification and planning. In: W. Pedrycz, A. Skowron, V. Kreinovich (eds.) Handbook of Granular Computing, pp. 777–799. John Wiley & Sons, The Atrium, Southern Gate, Chichester, England (2008)
7. Bazan, J.G., Kruczek, P., Bazan-Socha, S., Skowron, A., Pietrzyk, J.: Automatic planning based on rough set tools: Towards supporting treatment of infants with respiratory failure. In: Proceedings of the Workshop on Concurrency, Specification, and Programming (CS&P'06), September 27–29, Wandlitz, Germany, *Informatik-Bericht*, vol. 170, pp. 388–399. Humboldt University, Berlin, Germany (2006)
8. Bazan, J.G., Kruczek, P., Bazan-Socha, S., Skowron, A., Pietrzyk, J.: Automatic planning of treatment of infants with respiratory failure through rough set modeling. In: Proceedings of the Fifth International Conference on Rough Sets and Current Trends in Computing (RSCTC'06), November 6–8, Kobe, Japan, *Lecture Notes in Artificial Intelligence*, vol. 4259, pp. 418–427. Springer-Verlag, Berlin, Heidelberg, Germany (2006)
9. Bazan, J.G., Kruczek, P., Bazan-Socha, S., Skowron, A., Pietrzyk, J.: Risk pattern identification in the treatment of infants with respiratory failure through rough set modeling. In: Proceedings of the Eleventh Conference of Information Processing and Management of Uncertainty in Knowledge-Based Systems (IPMU'06), July 2–7, Paris, France, pp. 2650–2657 (2006)
10. Bazan, J.G., Kruczek, P., Bazan-Socha, S., Skowron, A., Pietrzyk, J.: Rough set approach to behavioral pattern identification. *Fundamenta Informaticae* **75**(1–4), 27–47 (2007)
11. Bazan, J.G., Skowron, A.: Classifiers based on approximate reasoning schemes. In: B. Dunin-Kępcicz, A. Jankowski, A. Skowron, M. Szczuka (eds.) Monitoring, Security, and Rescue Techniques in Multiagent Systems, *Advances in Soft Computing*, pp. 191–202. Springer Verlag, Heidelberg (2005)
12. Borrett, S.R., Bridewell, W., Langley, P., Arrigo, K.R.: A method for representing and developing process models. *Ecological Complexity* **4**, 1–12 (2007)
13. Breiman, L.: Statistical modeling: the two cultures. *Statistical Science* **16**(3), 199–231 (2001)
14. Desai, A.: Adaptive complex enterprises. *Communications ACM* **5**(48), 32–35 (2005)
15. Doherty, P., Łukaszewicz, W., Skowron, A., Szałas, A.: Knowledge Engineering: A Rough Set Approach. Springer, Heidelberg, Germany (2006)
16. Domingos, P.: Toward knowledge-rich data mining. *Data Mining and Knowledge Discovery* **1**(15), 21–28 (2007)
17. Friedman, J., Hastie, T., Tibshirani, R.: The Elements of Statistical Learning: Data Mining, Inference, and Prediction, vol. I–V. Springer-Verlag, Heidelberg (2001)
18. Guarino, N.: Formal ontology and information systems. In: Proceedings of the First International Conference on Formal Ontology in Information Systems (FOIS'98), June 6–8, Trento, Italy, pp. 3–15. IOS Press (1998)
19. Hillerbrand, R., Sandin, P., M.Peterson, (Eds.), S.R.: Handbook of Risk Theory: Epistemology, Decision Theory, Ethics, and Social Implications of Risk. Springer, Heidelberg, Germany (2012)
20. Hoen, P.J., Tuyls, K., Panait, L., Luke, S., Poutré, J.A.L.: Overview of cooperative and competitive multiagent learning. In: K. Tuyls, P.J. Hoen, K. Verbeeck, S. Luke (eds.) Learning and Adaption in Multi-Agent Systems: First International Workshop (LAMAS 2005), July 25, Utrecht, The Netherlands, pp. 1–46 (2005)
21. Ignizio, J.P.: An Introduction to Expert Systems. McGraw-Hill, New York (1991)
22. Jarrar, M.: Towards methodological principles for ontology engineering. Ph.D. thesis, Vrije Universiteit Brussel (2005)
23. Keefe, R.: Theories of Vagueness. Cambridge University Press, New York (2000)
24. Kloesgen, E., (eds.), J.Z.: Handbook of Knowledge Discovery and Data Mining. Oxford University Press, Oxford, UK (2002)
25. Kriegel, H.P., Borgwardt, K.M., Kröger, P., Pryakhin, A., Schubert, M., Zimek, A.: Future trends in data mining. *Data Mining and Knowledge Discovery* **1**(15), 87–97 (2007)

26. Langley, P.: Cognitive architectures and general intelligent systems. *AI Magazine* **27**, 33–44 (2006)
27. Liu, J., Jin, X., Tsui, K.: *Autonomy Oriented Computing: From Problem Solving to Complex Systems Modeling*. Kluwer/Springer, Heidelberg (2005)
28. Luck, M., McBurney, P., Shehory, O., Willmott, S.: *Agent technology: Computing as interaction. a roadmap for agent-based computing*. Agentlink iii, the european coordination action for agent-based computing, University of Southampton, UK (2005)
29. Michalski, R., et al. (eds.): *Machine Learning*, vol. I-IV. Morgan Kaufmann, Los Altos (1983, 1986, 1990, 1994)
30. Michie, D., Spiegelhalter, D.J., Taylor, C.C.: *Machine learning, neural and statistical classification*. Ellis Horwood Limited, England (1994)
31. Mitchell, T.M.: *Machine Learning*. McGraw-Hill, Boston, MA (1997)
32. Pancercz, K., Suraj, Z.: Rough sets for discovering concurrent system models from data tables. In: A.E. Hassanien, Z. Suraj, D. Ślęzak, P. Lingras (eds.) *Rough Computing: Theories, Technologies and Applications*. Idea Group, Inc. (2007)
33. Pawlak, Z.: *Rough Sets: Theoretical Aspects of Reasoning about Data, D: System Theory, Knowledge Engineering and Problem Solving*, vol. 9. Kluwer Academic Publishers, Dordrecht, The Netherlands (1991)
34. Pawlak, Z., Skowron, A.: Rudiments of rough sets. *Information Sciences* **177**, 3–27 (2007)
35. Peters, J.F.: Rough ethology: Towards a biologically-inspired study of collective behavior in intelligent systems with approximation spaces. *Transactions on Rough Sets* **3400(III)**, 153–174 (2005)
36. Peters, J.F., Skowron, A.: Zdzisław Pawlak life and work (1926–2006). *Information Sciences* **177**, 1–2 (2007)
37. Poggio, T., Smale, S.: The mathematics of learning: Dealing with data. *Notices of the American Mathematical Society (AMS)* **5(50)**, 537–544 (2003)
38. Polkowski, L., Skowron, A.: Rough mereology. In: Z.W. Raś, M. Zemankova (eds.) *Proceedings of the Eighth International Symposium on Methodologies for Intelligent Systems (IS-MIS'94)*, October 16-19, Charlotte, NC, USA, *Lecture Notes in Artificial Intelligence*, vol. 869, pp. 85–94. Springer Verlag (1994)
39. Read, S.: *Thinking about Logic: An Introduction to the Philosophy of Logic*. Oxford University Press, New York (1994)
40. Ripley, B.D.: *Pattern Recognition and Neural Networks*. Cambridge University Press, Cambridge (1996)
41. Rough ICE: Project web site: <http://logic.mimuw.edu.pl/~bazan/roughice/>
42. RSES: Project web site: <http://logic.mimuw.edu.pl/~rses>
43. Simulator, R.: Project web site: <http://logic.mimuw.edu.pl/~bazan/simulator>
44. Stone, P., Sridharan, M., Stronger, D., Kuhlmann, G., Kohl, N., Fidelman, P., Jong, N.K.: From pixels to multi-robot decision-making: A study in uncertainty. *Robotics and Autonomous Systems* **54(11)**, 933–943 (2006)
45. Suraj, Z.: Discovering concurrent data models and decision algorithms from data: A rough set approach. *International Journal on Artificial Intelligence and Machine Learning* **IRSI**, 51–56 (2004)
46. The Infobright Community Edition (ICE): Homepage at: <http://www.infobright.org/>
47. Unnikrishnan, K.P., Ramakrishnan, N., Sastry, P.S., Uthrusamy, R.: Service-oriented science: Scaling escience impact. In: *Proceedings of the Fourth KDD Workshop on Temporal Data Mining: Network Reconstruction from Dynamic Data*, The Twelfth ACM SIGKDD International Conference on Knowledge Discovery and Data (KDD'06), August 20-23, Philadelphia, USA (2006)
48. Urmson, C., et al.: High speed navigation of unrehearsed terrain: Red team technology for grand challenge. Report CMU-RI-TR-04-37, The Robotics Institute, Carnegie Mellon University (2004)

49. Vapnik, V. (ed.): *Statistical Learning Theory*. Wiley, New York (1998)
50. Zadeh, L.A.: From computing with numbers to computing with words – from manipulation of measurements to manipulation of perceptions. *IEEE Transactions on Circuits and Systems – I: Fundamental Theory and Applications* **1**(45), 105–119 (1999)
51. Zadeh, L.A.: Toward a generalized theory of uncertainty (GTU) - an outline. *Information Sciences* **171**, 1–40 (2005)