Temporal Templates and Analysis of Time Related Data

Piotr Synak

Polish-Japanese Institute of Information Technology
Koszykowa 86, 02-008 Warsaw, Poland
email: synak@pjwstk.waw.pl

Abstract. In the paper we investigate the problem of analysis of time related information systems. We introduce notion of temporal templates, i.e. homogeneous patterns occurring in some periods. We show how to generate temporal templates and time dependencies among them. We also consider decision rules describing dependencies between some temporal features of objects. Finally, we present how temporal templates can be used to discover behaviour of such temporal features.
Keywords: temporal template, temporal feature extraction, rough sets

1 Introduction

The intelligent analysis of data sets describing real life problems becomes a very important topic of current research in computer science. Different kind of data sets, as well as different types of problems they describe, cause that there is no universal methodology nor algorithm to solve these problems. For example, analysis of a given data set (information system) may be completely different, if we define a time order on a set of objects described by this data set, because the problem may be redefined to include time dependencies. Also, the expectation of an analyst may be different for the same data set, according to the situation. Let us consider a decision problem described by an information system (decision table), where objects are ordered according to time. In one situation the analyst may want to extract typical decision rules, e.g. "*if* $a = v$ *and* $c = w$ *then decision* $= d$", but another time, information about how the change of given condition (attribute) influences change of decision, e.g. "*if* Δa*=high positive and* Δc*=high negative then* Δ*decision* $=$ *neutral*". Much more general problem is to find, for a given property of condition ($\Delta a = positive$ is an example of property *positive change* of condition a), temporal dependencies giving the idea about periods of occurrences of this property, as well as temporal dependencies between different properties. For example, we can discover, that, if properties "$\Delta a = positive$" and "Δc=*negative*" appear together in the same time and last for some long period, it means, that after a certain time another set of properties will appear together (e.g. properties "$\Delta b = neutral$" and "$\Delta c = positive$").

In the paper we introduce the notion of temporal template, i.e. homogeneous pattern occurring in time. We show how to extract several different temporal templates and how to discover dependencies among them. We also present a method of generation of more general decision rules, which contain information about, e.g changes of values in some period. We show how temporal templates can be used to track temporal behaviour of attribute properties.

W. Ziarko and Y. Yao (Eds.): RSCTC 2000, LNAI 2005, pp. 420−427, 2001.

2 Temporal templates

First, let us define the way we represent data sets. The basic notion is an *information system* [8] which is defined as a pair $\mathbf{A} = (U, A)$, where U is a non-empty, finite set called the *universe* and A is a non-empty, finite set of *attributes*. Each $a \in A$ corresponds to function $a : U \to V_a$, where V_a is called the *value set* of a. Elements of U are called *objects*, *situations* or *rows*, interpreted as, e.g., cases, states, patients, observations.

A special case of information system is a *decision table* $\mathbf{A} = (U, A \cup \{d\})$, where $d \notin A$ is a distinguished attribute called *decision*. The elements of A are called *conditional attributes (conditions)*.

Any expression of the form $(a \in V)$, where $a \in A \cup \{d\}, V \subseteq V_a$ we call a *descriptor*. A descriptor $(a \in V)$ is a *conditional descriptor* iff $a \in A$, or *decision descriptor* iff $a \in \{d\}$.

We say that an object $x \in U$ *matches* a descriptor $(a \in V)$ iff $a(x) \in V$.

The notion of templates was intensively studied in literature (see e.g. [1], [5], [7]). For a given information system $\mathbf{A} = (U, A)$ by *generalized template* we mean a set of descriptors

$$T = \{(a \in V) : V \subseteq V_a\} \tag{1}$$

such, that, if $(a \in V) \in T$ and $(b \in W) \in T$ then we have $a \neq b$. An object $x \in U$ *matches* a generalized template T, if it matches all descriptors of T. A special case of generalized templates are templates with one-value descriptors, i.e. of form $(a = v)$. Templates can be understood as patterns which determine homogeneous subsets of information system

$$T(\mathbf{A}) = \{x \in U : \forall_{(a \in V) \in T} \ \ a(x) \in V\} \tag{2}$$

In many practical problems a very important role plays time domain. In this case, the set of objects is ordered according to time $\mathbf{A} = (\{x_1, x_2, ..., x_n\}, A)$, and we say, that t is the time of occurrence of object $x_t, 1 \leq t \leq n$. From now on by information system we mean one in a sense presented above. In such systems one can consider templates that occur in some period. This kind of templates we call *temporal templates* and define as

$$\mathbf{T} = (T, t_s, t_e), \ \ 1 \leq t_s \leq t_e \leq n \tag{3}$$

We say, that two temporal templates $\mathbf{T_1} = (T_1, t_s, t_e)$ and $\mathbf{T_2} = (T_2, t_s^{'}, t_e^{'})$ are *equal* if $T_1 = T_2$.

Now, let us define two important properties of temporal templates. By *width* of temporal template $\mathbf{T} = (T, t_s, t_e)$ we mean $width(\mathbf{T}) = t_e - t_s + 1$, i.e. the length of the period which T occurs in. Please notice, that not all objects $x_{t_s}, x_{t_s+1}, ..., x_{t_e}$ have to match T and intuitively the more objects match T the better. Thus, by *support* of $\mathbf{T}$ we understand $supp(\mathbf{T}) = card(T(\mathbf{A_T}))$, where $\mathbf{A_T} = (\{x_{t_s}, ..., x_{t_e}\}, A)$ is the information system determined by $\mathbf{T}$ and $T(\mathbf{A_T})$ is set of objects of this system that match T. The *quality* of temporal template is a function that maximizes width, support as well as number and precision of descriptors.

In Figure 1 we show two examples of temporal templates $\mathbf{T_1} = (\{(a \in \{u\}), (c \in \{v\})\}, 2, 8)$ and $\mathbf{T_2} = (\{(b \in \{x\}), (d \in \{y\})\}, 10, 13)$.

	A	a	b	c	d	e
	x_1	·	·	·	·	·
$\mathbf{T}_1$	x_2	u	·	v	·	·
	x_3	u	·	v	·	·
	x_4	·	·	·	·	·
	x_5	u	·	v	·	·
	x_6	u	·	v	·	·
	x_7	·	·	·	·	·
	x_8	u	·	v	·	·
	x_9	·	·	·	·	·
$\mathbf{T}_2$	x_{10}	·	x	·	y	·
	x_{11}	·	x	·	y	·
	x_{12}	·	·	·	y	·
	x_{13}	·	x	·	y	·
	x_{14}	·	x	·	·	·
	x_{15}	·	·	·	·	·

Fig. 1. An example of temporal templates.

3 Temporal templates generation

In this section we present an algorithm that generates several temporal templates which are disjoint according to time. The main idea is based on scanning the information system with some time window and generation of best generalized template within this window. There is a chance, that after a light shift of a time window, the previously found template is also the best one in a new window. By shifting the window we may discover the beginning (x_{t_s}) and the end (x_{t_e}) of the area where a given template is optimal or close to the optimal one. Shifting the time window through the whole information system generates a set of temporal templates.

Let $\mathbf{A} = (\{x_1, x_2, ..., x_n\}, A)$ be an information system. By *time window* on $\mathbf{A}$ of size s in point t we understand an information system $win^s_{\mathbf{A}}(t) = (\{x_t, x_{t+1}, ..., x_{t+s-1}\}, A)$, where $1 \leq t$ and $t + s - 1 \leq n$. In the process of temporal rules generation both size s of a window and number of used windows are parameters that have to be tuned according to the type of data. Below we present details of the algorithm.

Algorithm: Temporal templates generation

Input: Information system $\mathbf{A}$, size of time window *size*, length of the shift of time window *step*, quality threshold τ

Output: Set of temporal templates

1. $i := 1, T = NULL, t_s = 1$
2. **while** $i < n - size$ **begin**
3. $\quad best = FindBestTemplate(win^{size}_{\mathbf{A}}(i))$
4. $\quad$ **if** $best \neq T$ **then begin**
5. $\quad\quad t_e = i$
6. $\quad\quad$ **if** $Quality((T, t_s, t_e)) \geq \tau$ **then**
7. $\quad\quad\quad$ **output** (T, t_s, t_e)
8. $\quad\quad t_s = i$
9. $\quad\quad T = best$
10. $\quad$ **end if**
11. $\quad i := i + step$
12. **end while**

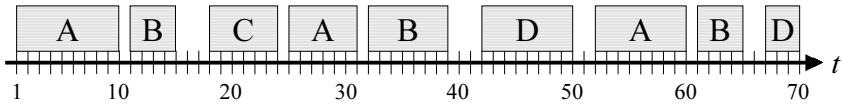

Fig. 2. A series of temporal templates.

At each run of the loop the algorithm is shifting the time window. The task of subroutine $FindBestTemplate(\mathbf{A})$ is to return the optimal (or semi-optimal) template for **A**. Because the problem of optimal template generation is NP-hard (see e.g. [5]) some approximation algorithm to be used in this procedure should be considered. In [5], [7] there can be found several very fast heuristics for templates generation.

The input parameter τ is used to filter out temporal templates of low quality. The definition of template quality may be different and it depends on the formulation of a problem.

4 Discovery of dependencies between temporal templates

Generation of series of temporal templates defines a very interesting problem related to discovery of dependencies among them. Several templates may occur many times and it can happen, that one template is always (or almost always) followed by another one. Generally, one set of templates may imply occurrence of other templates and extraction of such information may be very useful. Let us consider the situation, that we are observing occurrence of some temporal template in current data. On the basis of template dependencies we predict occurrence of another template in the near future. In this section we present a simple method that allows to discover such template dependencies.

Let us observe, that the process of temporal templates generation results with a series of templates. On time axis each template has its time of occurrence and width. In Figure 2 we have an example series of four temporal templates **A**, **B**, **C**, **D**. From such a series we want to extract three kinds of dependencies. First, the dependencies between templates, e.g. decision rules of form "*if template in time t-3 is* **B** *and template in time t-1 is* **A** *then template in time t is* **D**". Second, if we have such a rule then on the basis of widths of templates **B** and **A** we want to predict the width of **D**. Third kind of information, we need, is the time of occurrence (t_s) of template **D**.

Now, let us focus on the problem of decision rules generation, that map dependencies between templates. Templates can be treated as events and the problem may be reformulated in terms of frequent episodes detection in time series of events. This problem is investigated in e.g. [4], however, the difference is, that here events are not points on time axis (as in [4]), but intervals of different length. Thus, we propose another method which takes advantage of rough set theory.

The idea is based on construction of a decision table from time series of templates and further computation of decision rules. The number of condition attributes n is a parameter of this method and it reflects how many steps in past we want to consider. The objects of this table we construct from series of templates - one object is a consecutive sequence of $n + 1$ template labels. For

example, if our template series is one presented in Figure 2, i.e. **A**, **B**, **C**, **A**, **B**, **D**, **A**, **B**, **D** and $n = 2$ we obtain the decision table as in Table 1.

	T_{t-2}	T_{t-1}	T_t
x_1	**A**	**B**	**C**
x_2	**B**	**C**	**A**
x_3	**C**	**A**	**B**
x_4	**A**	**B**	**D**
x_5	**B**	**D**	**A**
x_6	**D**	**A**	**B**
x_7	**A**	**B**	**D**

Table 1. Decision table constructed from a sequence of temporal templates

From this table we can generate decision rules using, e.g. rough set methods (see [2], [3], [8]). In our example we obtain, among other, the following rules:
if $T_{t-1} =$ **A then** $T_t =$ **B**
if $T_{t-2} =$ **B then** $T_t =$ **A**
if $T_{t-2} =$ **A and** $T_{t-1} =$ **B then** $T_t =$ **D**

Once we have a set of decision rules we can compute, for each rule, how widths of predecessor templates of the rule determine width of successor template. For a given rule, we construct a new decision table with condition attributes responding to widths of predecessor templates of the rule and decision attribute from width of successor template. As condition attributes we also consider length of gaps on time axis between consecutive templates. The set of objects we create on the basis of all objects of input decision table matching the rule.

Suppose, we consider decision rule *if* $T_{t-2} =$ **A** *and* $T_{t-1} =$ **B** *then* $T_t =$ **D**. There are two objects x_4, x_7 matching this rule. We check widths and gaps between templates represented by these objects and create a decision table as in Table 2a. It is obvious, that before further processing of this table, it should be scaled to contain more general values (Table 2b). From such a table we can compute decision rules expressing widths of templates.

	A	$Gap_{\mathbf{AB}}$	**B**	**D**
x_1	7	1	8	9
x_2	9	1	5	4

(a)

	A	$Gap_{\mathbf{AB}}$	**B**	**D**
x_1	*high*	*small*	*high*	*high*
x_2	*high*	*small*	*medium*	*medium*

(b)

	A	$Gap_{\mathbf{AB}}$	**B**	$Gap_{\mathbf{BD}}$
x_1	7	1	8	3
x_2	9	1	5	2

(c)

	A	$Gap_{\mathbf{AB}}$	**B**	$Gap_{\mathbf{BD}}$
x_1	*high*	*small*	*high*	*small*
x_2	*high*	*small*	*medium*	*small*

(d)

Table 2. Decision tables describing gaps between and widths of templates generated for sample decision rule.

If we already have the information about which template is going to appear next and what is its expected width, what we still need to know is the estimated time of its appearance. This information we can generate in an analogous way as in case of width. The difference is, that, when constructing decision table, we take as decision attribute not the template, which is in the successor of the rule, but the length of the gap between this template and last predecessor template (see Table 2cd). Finally, we compute rules expressing time of template occurrence.

5 Temporal features and decision rules

In this section we investigate the problem of decision rules generation that contain new features describing behaviour of objects in time. We assume, that the input data are numerical, so we can say about the degree of change of attribute value. An example of such a rule is "*if change of attribute a is positive and change of attribute c is negative then change of decision is negative*". Decision rules of this kind are very useful for analysis of time related information systems. We also show, how to compute templates built from descriptors of form $(\Delta a = positive)$, that can be further used to more general association rules generation.

In our method, first, we construct an information system which is a result of scanning of input data with a time window of some size. The size of a window is the parameter and can be tuned up according to type of data. When looking at the data through a time window we observe a history of changes of attribute values within some period (which is related to the size of the window). On the basis of this history we can construct new features describing behaviour of attributes in time, e.g. characteristics of plots of values, information about trends of changes. In our method let us focus on one example of such a temporal feature, which is the change of value within a time window.

Let $\mathbf{A} = (\{x_1, x_2, ..., x_n\}, \{a_1, ..., a_m\})$ be an information system and let us consider time windows of size s on $\mathbf{A}$. We construct a new information system $\mathbf{A}_s = (\{y_1, y_2, ..., y_{n-s+1}\}, \{\Delta a_1, ..., \Delta a_m\})$ in the following way:

$$\Delta a_i(y_j) = a_i(x_{j+s-1}) - a_i(x_j). \tag{4}$$

In Table 3a we have a sample information system, which after scanning with a time window of size $s = 3$ is as one in Table 3b.

$\mathbf{A}$	a_1	a_2	a_3	d
x_1	1.1	2.5	2.1	0.8
x_2	2.0	3.0	1.8	1.0
x_3	2.3	2.6	0.6	0.5
x_4	1.0	1.8	0.7	2.5
x_5	1.8	1.7	1.2	1.6
x_6	1.2	1.9	2.0	1.7
x_7	0.5	1.5	0.2	0.4
x_8	0.7	1.7	1.5	0.9
x_9	1.0	2.5	1.4	1.9

(a)

$\mathbf{A}_3$	Δa_1	Δa_2	Δa_3	Δd
y_1	1.2	0.1	-1.5	-0.3
y_2	-1.0	-1.2	-1.1	1.5
y_3	-0.5	-0.9	0.6	1.1
y_4	0.2	0.1	1.3	-0.8
y_5	-1.3	-0.2	-1.0	-1.2
y_6	-0.5	-0.2	-0.5	-0.8
y_7	0.5	1.0	1.2	1.5

(b)

$\mathbf{A}'_3$	Δa_1	Δa_2	Δa_3	Δd
y'_1	positive	neutral	negative	neutral
y'_2	negative	negative	negative	positive
y'_3	neutral	negative	positive	positive
y'_4	neutral	neutral	positive	negative
y'_5	negative	neutral	negative	negative
y'_6	neutral	neutral	neutral	negative
y'_7	neutral	positive	positive	positive

(c)

$\mathbf{A}''_3$	Δa_1	a_1	Δa_2	a_2	Δa_3	a_3	Δd
y''_1	positive	low	neutral	med.	negative	high	neutral
y''_2	negative	high	negative	high	negative	high	positive
y''_3	neutral	high	negative	high	positive	low	positive
y''_4	neutral	low	neutral	low	positive	low	negative
y''_5	negative	high	neutral	low	negative	med.	negative
y''_6	neutral	med.	neutral	low	neutral	high	negative
y''_7	neutral	low	positive	low	positive	low	positive

(d)

Table 3. (a) Sample information system, (b) after scanning with a time window of size 3, (c) scaled, (d) scaled and containing new attributes (levels).

The obtained information system should be scaled next, so the results of analysis could be more general. In our example we use three-value scale, which describes the degree of change - *negative, neutral* or *positive* (see Table 3c). Obtained information system is a base for our further analysis.

First, let us consider templates computed for such system (see e.g. [7], [5]). They are built from descriptors of form $(\Delta a = v)$ (or $(\Delta a \in \{v_1, v_2, ...\})$ in more general case) and long templates with large support may contain very useful knowledge about higher-order dependencies between attributes. These templates can be used to generation of approximate association rules (see [6]) built from higher-order descriptors. An example of templates for Table 3c is

$T_1 = \{(\Delta a_1 = neutral), (\Delta a_3 = positive)\}$
$T_2 = \{(\Delta a_2 = neutral), (\Delta d = negative)\}$

Now, suppose the decision attribute is defined and we consider a decision table $\mathbf{A} = (\{x_1, x_2, ..., x_n\}, \{a_1, ..., a_m\} \cup \{d\})$ describing some decision problem. Using the method described above we can generate a decision table built from information about attribute changes (including decision), which we can compute decision rules for. Obtained rules contain knowledge about how value changes of conditional attributes determine change of decision values.

For example, if we consider the information system presented in Table 3a as decision table, with last attribute being a decision, after processing, we obtain Table 3c. From this table we can extract, e.g., the following rules:

if $\Delta a_2 = negative$ **then** $\Delta d = positive$
if $\Delta a_1 = neutral$ **and** $\Delta a_2 = neutral$ **then** $\Delta d = negative$

Another extension of this method is to include, in the final decision table, more conditional attributes describing, e.g. levels of attribute values. It can happen, that the degree of change of decision attribute depends not only on degrees of conditional attributes changes, but also on the levels of values. In Table 3d we have an example of such attributes which are scaled into three classes - *low, medium, high*. From this table we extract, among other, the following rules:

if $\Delta a_2 = neutral$ **and** $a_2 = low$ **then** $\Delta d = negative$
if $\Delta a_2 = negative$ **and** $a_2 = high$ **then** $\Delta d = positive$

The first rule, for example, can be interpreted as "*if value of* a_2 *isn't changing much (in some period) when it is low, then the value of decision is decreasing*".

6 Behaviour of temporal features in time

In the previous section we showed how new features, that express behaviour of attributes in a time window, can be extracted from an information system. There can be several features considered - as an example we took a difference between value at the end and the beginning of time window. Now, let us investigate the problem of temporal behaviour discovery of a group of features.

Suppose, we consider a feature "a_1 grows fast" discovered from time related information system. There can be considered several different problems related to this feature. The basic one is to discover periods that this feature holds in. Besides, we would like to know what are other properties (e.g. "a_3 behaves stably") that hold at the same time. Another question is what are the symptoms that this feature is about to finish and what features are going to appear next.

We believe, that temporal templates generation is a tool which helps to answer above questions. First, we analyze the information system using time window method and construct new system built from temporal features. Then, we compute temporal templates and generate dependencies among them. One can notice, that temporal template, for so processed information system, contains information about a set of features that hold at the same period, as well as about beginning and end time of this period. Analysis of dependencies between temporal templates gives the idea about what new set of features may appear after a current one.

7 Summary

We claim that the notion of temporal templates can be very useful for analysis of time related information systems. Investigation of dependencies between temporal templates gives the idea how the knowledge hidden in data is changing in time. Very important topic is the extraction of new features from data, describing temporal properties of data. Finally, temporal templates can be used to check how these features behave in time.

Acknowledgments

This work was supported by the grants of Polish National Committee for Scientific Research (KBN) No. $8T11C02417$ and $8T11C02519$.

References

1. Agrawal, R., Mannila, H., Srikant, R., Toivonen, H., Verkamo, I.: Fast Discovery of Association Rules, *Proceedings of the Advances in Knowledge Discovery and Data Mining.* AAAI Press/The MIT Press, CA (1996) 307–328.
2. Bazan, J.: Dynamic reducts and statistical inference, *Proceedings of Information Processing and Management of Uncertainty on Knowledge Based Systems (IPMU-96)*, July 1-5, Granada, Spain, Universidad de Granada, vol. III, (1996) 1147–1152.
3. Bazan, J., Skowron, A. and Synak, P.: Discovery of Decision Rules from Experimental Data, *Proceedings of the Third International Workshop on Rough Sets and Soft Computing.* San Jose, California (1994) 526–533.
4. Mannila, H., Toivonen, H.,Verkamo, A.,I.: Discovery of frequent episodes in event sequences. *Report C-1997-15*, University of Helsinki, Department of Computer Science, Finland (1997).
5. Nguyen S.H.: Regularity Analysis And Its Applications In Data Mining. PhD Dissertation, Warsaw University, Poland (2000).
6. Nguyen, H.S., Nguyen, S.H.: Rough Sets and Association Rule Generation. Fundamenta Informaticae **40/4** (2000) pp. 383–405.
7. Nguyen, S.H., Skowron, A., Synak, P.: Discovery of data pattern with applications to decomposition and classification problems. In L. Polkowski, A. Skowron (eds.):*Rough Sets in Knowledge Discovery* **2**. Physica-Verlag, Heidelberg (1998) 55–97.
8. Pawlak, Z., Skowron, A.: A rough set approach for decision rules generation, *ICS Research Report 23/93*, Warsaw University of Technology, *Proceedings of the IJCAI'93 Workshop W12: The Management of Uncertainty in AI*, France (1993).
9. Ziarko, W., Shan, N.: An incremental learning algorithm for constructing decision rules, *Proceedings of the International Workshop on Rough Sets and Knowledge Discovery.* Banff. (1993) 335–346.