Specification and Detection of Periodic Patterns in Clinical Data

Shubha Chakravarty and Yuval Shahar
Stanford Medical Informatics, Stanford University, Stanford, CA
{schakrav, shahar}@smi.stanford.edu

Abstract

We specify and detect periodic patterns in clinical data using a constraint-based pattern specification language. The language has been implemented within both a graphical knowledge acquisition tool and a pattern interpreter, both of which are implemented within a larger temporal abstraction architecture. We provide several examples of how to use the syntax of the pattern language to specify complex periodic patterns. Working in the domains of diabetes and oncology, we have begun an evaluation of the pattern language, the knowledge acquisition process, and the pattern interpreter. Using clinical data from the University of Chicago bone marrow transplantation center, we have tested the interpreter's ability to detect patterns. Preliminary results show that domain experts can use our language to specify most patterns of interest in the two domains.

1. Introduction

The detection of periodic patterns in time-oriented medical data is often an important component of the analysis of large data sets. In many medical domains, from diabetes to oncology, care providers and clinical researchers need to specify and detect recurring trends not only at the level of time-stamped data, but also at higher levels of data abstraction. For example, an oncologist is often interested in monitoring trends in a patient's anemia level (from moderate to severe), derived from a patient data set that contains only individual blood test values indicating the level of hemoglobin at a certain time point. To support the detection of periodicity, an intelligent medical data analysis system must first recognize a higher level, clinically meaningful concept such as anemia from raw level hemoglobin values; second, the system must combine individual data points into larger intervals; and third, the system must detect trends in the repeating intervals of anemia. Figure 1 shows these levels of data analysis.

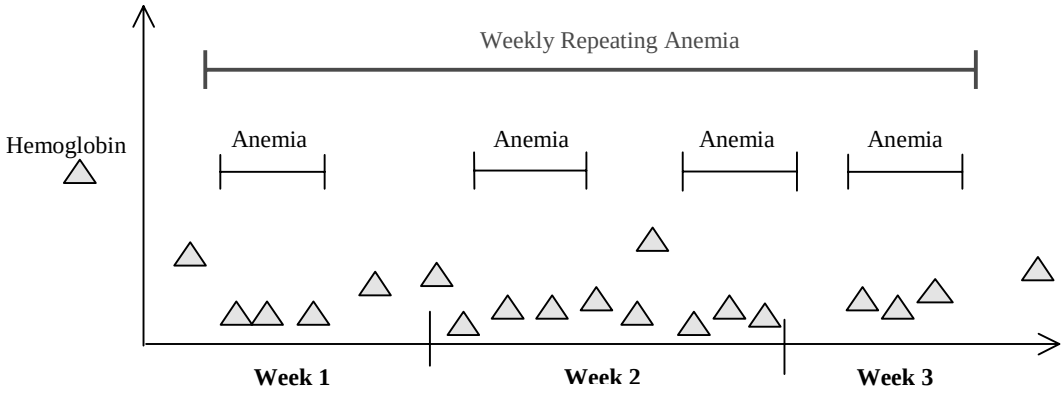

Figure 1. The levels of temporal data abstraction. The lowest level of this graph shows raw data indicating the results of hemoglobin tests. These data points are combined into intervals of a higher level concept *Anemia*. The intervals of *Anemia* can then be combined into an overall interval of *Repeating Anemia*. Thus a physician examining this patient record could see immediately that the patient has frequent periods of anemia.

In order to accomplish the task of periodic pattern abstraction, our data analysis system needs the following components: a language for expressing the sort of periodic patterns that we want to detect, a knowledge base to define the abstract concepts relevant to the domain, a knowledge acquisition tool that domain experts can use to modify and maintain the knowledge base, a patient database that can be mapped to the format expected by the interpreter , and an interpreter to derive the abstractions from the data set. Together, these components comprise the **Résumé** system [Shahar and Musen, 1996]. The Résumé system uses the **knowledge-based temporal-abstraction (KBTA)** method [Shahar, 1997] to derive high level concepts, or *abstractions*, from raw level data. Each derived abstraction must be associated with a *context*, since the definitions of these abstractions can change depending on the context in which the data were acquired [Shahar, 1998]. Using a knowledge base of domain-specific contexts, parameters, events, and patterns, and a database of time-stamped raw-level data, the Résumé interpreter derives a set of abstractions that can

then be visualized and summarized for a user. The knowledge base contains not only the definitions of the concepts, but also the properties of these concepts (such as persistence) that are necessary for deriving them from raw data .

To acquire and detect knowledge about patterns, Résumé uses the **Constraint-Based Pattern Specification Language (CAPSUL)** both to acquire knowledge about patterns from domain experts and to search for these patterns in a data set. CAPSUL, which has been previously described in detail [Chakravarty and Shahar, 1999], defines a set of expressive constraints upon temporal objects that already exist within the Résumé temporal ontology. Using a graphical knowledge acquisition (KA) tool whose frames mirror the syntax of CAPSUL, expert physicians can specify periodic pattern queries using a variety of temporal and value constraints. These patterns are then combined with the rest of the Résumé temporal ontology and can be derived from a data set by the Résumé interpreter, which includes the CAPSUL language interpreter. The output of the Résumé interpreter is a set of interval-based abstractions (including pattern abstractions) that are directly useful to applications such as a therapy planner (EON) [Musen, et. al 1996] or that can be displayed and explored using a visualization tool such as the Knowledge-Based Abstraction Visualization and Exploration (KNAVE) system [Shahar and Cheng, 1998]. Figure 2 shows a high-level view of all the components and typical applications using the Résumé system.

In this paper, we first describe briefly the CAPSUL language, providing examples of the types of patterns that we can specify and detect. Second, we discuss the knowledge acquisition process, including our success at having domain experts use our KA tool. Third, we provide results of a recent and ongoing evaluation of our pattern detection system using data sets in the domains of oncology and diabetes and an expert in each domain. Fourth, we briefly survey related work in periodic pattern representation. The preliminary results of our evaluation indicate that CAPSUL can capture most useful patterns in which experts are interested, that our KA tool is usable by domain experts with the help of a knowledge engineer, and that the Résumé interpreter can correctly derive patterns when they exist in the data set.

2. Language

CAPSUL aims to provide an expressive yet simple language in which to describe one-time only ("linear") and repeating ("periodic") patterns of intervals. We use the better-known term *periodic* although CAPSUL patterns are more accurately described as *episodic*. CAPSUL patterns do not necessarily repeat with any regularity. At times, users specify patterns in which the only constraint is that a certain event occurs more than once; this type of pattern has no specifiable frequency or period associated with it. Because most readers are more familiar with the term periodic, and because users usually have periodic patterns in mind when using CAPSUL, we continue to refer to CAPSUL patterns as *periodic*.

The CAPSUL language is designed for the twin tasks of knowledge acquisition and data interpretation. At a high level, CAPSUL is a template for a pattern; users fill in this template with a variety of temporal objects and constraints to form a pattern specification. This pattern template is not specific to Résumé; it can be used to specify and detect temporal patterns in any domain. To understand CAPSUL better, we examine three key distinctions that structure the syntax of the language.

2.1. Description of the language

First, CAPSUL distinguishes between *what repeats* (the *single repeating component* of the pattern) from *how it repeats* (the *global periodic constraints* of the pattern). The repeating pattern components are temporal objects that have been previously defined in the temporal ontology; They may be patterns, events, parameters, or contexts. Each periodic pattern is defined by at most one repeating component; the content of what repeats must be expressed as a single abstraction. For example, a component can be a parameter-component such as *Anemia*, where anemia has been defined in the Résumé knowledge base. The constraints define the temporal and value relations between the components. Both the components and constraints of the pattern are specified in disjunctive normal form; that is, the user can define more than one set of components or constraints, each of which comprises a valid definition of the pattern. The separation of components and constraints has important implications for the semantics of CAPSUL patterns: this distinction separates CAPSUL from other pattern languages that define periodic patterns as intersections of nonconvex intervals [Ladkin, 1986a]. We will return to this key idea later.

Second, CAPSUL distinguishes between linear and periodic patterns. Linear patterns consist of a set (of one or more) *linear pattern components* that occurs exactly once, such as the linear sequence of high glucose followed by an insulin injection (note that in this example the first linear component is a parameter, while the second is an event). Meanwhile, periodic patterns consist of exactly one *periodic (or repeating) component* that repeats over time, such as daily glucose measurements. Either type of pattern can be used as a component of its own or of the other type of pattern.

Third, CAPSUL distinguishes among three different levels of constraints: local, global, and periodic. All constraints are defined as ranges to enhance flexibility. Local constraints apply to a single interval; they define the duration, value, and relevant scope of the interval. For example, the anemia-component defined previously could have a local value constraint of *severe* (where a "severe" value of anemia has been defined in the Résumé knowledge base), and a local duration constraint of 3-5 days. Global constraints define the temporal and value relations between two

components of a linear pattern. For example, a gap between two intervals is a global constraint that defines the temporal distance between any two (of four) endpoints of the intervals (begin-begin, begin-end, end-begin, end-end). Gaps can be qualitative (using Allen's algebra [Allen, 1984]) or quantitative (using numerical values). Furthermore, gaps can be specified using calendaric or absolute units. An *absolute* day is twenty four hours, even if those twenty four hours occur from 4 p.m. to 4 p.m. By contrast, a *calendaric* day requires that the twenty four hours occur from midnight to the next midnight. Finally, CAPSUL allows a limited number of statistical constraints that can be applied to intervals of a linear pattern. A statistical constraint applies to the mean, variance, minimum, or maximum value of a parameter-component. An example of the use of a statistical constraint is 2 weeks of blood glucose values where the average value of the first week is less than the average value of the second week.

Periodic constraints define the temporal and value relations between successive intervals of a periodic pattern. Gap and value constraints of this sort, once defined, apply to every successive pair of intervals of the pattern; users cannot specify a different gap or value relation for each pair of intervals of the pattern. Recall, however, that these gap and value relations can be specified as a disjunction of constraints that applies to every pair of successive intervals. Periodic constraints can also apply to the entire set of intervals that comprise the pattern; for example, the cardinality of the set and the scope of relevance in which the pattern must occur are examples of periodic constraints that apply to a whole set of intervals rather than a pair of successive intervals. Furthermore, periodic constraints can define sub-patterns, or *clusters*, of intervals within the larger pattern over which a set of sub-constraints applies (e.g., an overall pattern of monthly episodes of *Anemia* must contain at least one month in which there are 2 episodes of *Anemia*, both severe). A more complete description of these constraints can be found in [Chakravarty and Shahar, 1999]. A complete BNF syntax is included in the appendix to this paper. We now present several examples to better understand the semantics of the CAPSUL constraints.

2.2. Examples

A patient who has undergone a bone-marrow transplant (BMT) receives frequent platelet transfusions until the patient's body recovers and starts to produce platelets on its own again. The context of this pattern is BMT, since it applies only to patients who have had a bone-marrow transplant. The half-life of a platelet transfusion is defined as the

Periodic Pattern: *Declining Platelet Half Life*
- **Context**: BMT
- **Repeating Component**: Platelet Half Life
- **Constraints**:
  - **Cardinality**: 5-7 repetitions
  - **Temporal Envelope**: 5 weeks
  - **Value Constraint between successive intervals**: Less-than-or-equal

Linear Pattern: *Platelet Half Life*
- **Context**: BMT
- **Linear Components**:
  - **Event Component**: Transfusion-Component
    - **Abstracted from**: Platelet Transfusion
  - **Parameter Component**: High-Platelet-State
    - **Abstracted from**: Platelet-State
    - **Local Constraints**: value HIGH
  - **Parameter Component**: Low-Platelet-State
    - **Abstracted from**: Platelet-State
    - **Local Constraints**: value LOW
- **Global Constraints**:
  - **Quantitative Gap Constraint**: High-Platelet-State <= 1 hour AFTER Transfusion-Component
  - **Qualitative Gap Constraint**: Low-Platelet-State AFTER High-Platelet-State
- **Output Value of Pattern**:
  - **Duration Function**: [(Start of Low-Platelet-State) – (Start of High-Platelet-State)]/2

Figure 2. A partial parsing of the pattern "Declining Platelet Half Life" in the CAPSUL syntax. The repeating component slot of the periodic pattern points to an instance of a linear pattern, namely, the "Platelet Half Life" pattern. Note that the periodic pattern and linear pattern exist as instantiated temporal objects of their respective classes in the knowledge base. This pattern has pointers to other temporal objects in the knowledge base, such as "Platelet State" and "Platelet Transfusion".

time it takes for the Platelet level to decrease to one half of the level that was measured immediately post-transfusion; each Platelet transfusion has an associated half life. A physician who is monitoring such a patient wants to know if these successive platelet half lives are increasing or decreasing over time (in order to see how well the patient is adjusting to the transfused platelets). In this periodic pattern the repeating component is the half life of a single transfusion. The cardinality is 5-7 intervals (to ensure that the trend is not a random one) and the periodic temporal constraint between intervals is decreasing inter-interval gaps. Each *Platelet-half-life* is itself a linear pattern that consists of a platelet transfusion, the immediate post-transfusion platelet level (the *High* level), and the later *Low* platelet level. These levels, in turn are abstractions of raw platelet counts, and the linear pattern imposes certain local constraints on their values (e.g., one is at least *High*, and the other is *Low* or *Very_Low*) and global temporal constraint on the relation between them (e.g., the high value interval is *before* the low value interval; both are *after* the transfusion event, etc.). This pattern can be parsed using CAPSUL as in Figure 2.

A second example of a periodic pattern, from the diabetes domain, monitors how well a patient is controlling her glucose level. The relevant context in this case is diabetes. The *Recurring Weekend Glucose Monitoring* periodic pattern consists of low or very low morning glucose and high evening glucose (with at least one insulin injection in between), occurring during the weekend, for 5-7 weekends during the summer. Most diabetic patients record insulin administrations and blood glucose values on a daily basis. In this case the repeating component is a linear pattern called *Weekend Glucose Monitoring* that consists of three linear components: *Low-Morning-Glucose*, *High-Evening-Glucose*, and *Insulin-Injection*. To define the periodic constraints associated with this pattern, we must first define a calendar-

Periodic Pattern: *Recurring Weekend Glucose Monitoring*
- **Context**: Diabetes
- **Repeating Component**: Weekend Glucose Monitoring
- **Constraints**:
  - **Cardinality**: 5 –7 repetitions
  - **Relevant Scope**: Summer
  - **Cluster**: Highly-Variable Weekend
    - **Cluster Cardinality**: 2-5
    - **Cluster sub-constraints**: value VERY_LOW
      gap 7 days

Linear Pattern: *Weekend Glucose Monitoring*
- **Context**: Diabetes
- **Linear Components**:
  - **Parameter Component**: Low-Morning-Glucose
    - **Abstracted from**: Glucose State
    - **Local Constraints**: during MORNING
      value LOW or VERY_LOW
  - **Event Component**: Insulin-Injection
    - **Abstracted from**: Insulin
  - **Parameter Component**: High-Evening-Glucose
    - **Abstracted from**: Glucose State
    - **Local Constraints**: during EVENING
      Value HIGH
- **Global Constraints**:
  - **Quantitative Gap Constraint**: Gap b/w Insulin-Injection and High-Evening-Glucose < 2 hours
  - **Qualitative Gap Constraint**: Low-Morning-Glucose BEFORE Insulin-Injection
  - **Period**: Weekend
- **Output Value**: Value of Low-Morning-Glucose

Figure 3. A partial parsing of the pattern *Recurring Weekend Glucose Monitoring* in the CAPSUL syntax. The periodic pattern contains a cluster, or sub-pattern, called *Highly Variable Weekend*. The cluster cardinality refers to the number of such sub-patterns found in the overall pattern; in this case there are 2-5 consecutive weekends in the summer in which the value of the periodic component is *Very_Low*. Note that the slot for Period in the linear pattern points to a repeating temporal interval called *Weekend*, which has been defined elsewhere in the knowledge base. Similarly, the slot for Period in the periodic pattern points to *Summer*. Thus the linear pattern occurs during a weekend, and the periodic pattern occurs during the summer. The output value of the linear pattern *Weekend Glucose Monitoring* is mapped to the value of the Low-Morning-Glucose parameter, which is *Very_Low* or *Low*.

specific term, *Summer*, which itself is a repeating temporal interval. The periodic pattern contains a cardinality constraint (5-7 repetitions) and a period-constraint that specifies that the relevant scope of the pattern is Summer. Further, the *Recurring Weekend Glucose Monitoring* periodic pattern contains a cluster constraint that requires at least two consecutive so-called *Highly Variable Weekends* in which the value of the single repeating component is Very_Low. For this cluster, we must also define a repeating calendar interval of *Weekend*, just as we defined Summer. The entire pattern can be expressed using CAPSUL as in Figure 3.

The design of the CAPSUL syntax includes several limitations that prevent CAPSUL from expressing every possible periodic pattern. CAPSUL cannot handle periodic patterns for which the content of what repeats cannot be expressed as a single periodic component. CAPSUL also cannot handle patterns that require "non-linear" constraints; by non-linear we mean that the evaluation of the constraint over a given set of intervals requires an algorithm that is non-linear with respect to the number of intervals in the set. In other words, all constraints even at the periodic level must be able to be evaluated pairwise over a set of intervals. Consider the following example.

Suppose we want to compare how the average weekend glucose, as monitored by a diabetic patient, compares to the level of glycated hemoglobin, which is measured every other month and provides a more reliable measure of blood glucose. This pattern consists of: (1) high average glucose on weekends, 3 times a month, for at least half the months of the year, and (2) an episode of glycated hemoglobin during every other month in which the weekend glucose is high. On their own, either of these patterns can be expressed using CAPSUL; however, the combination of weekend glucose measurements and bimonthly glycated hemoglobin measurements cannot be expressed as a single CAPSUL pattern, since it consists of two independently repeating components that repeat at different frequencies. A repeating component cannot consist of both a high average weekend glucose and a glycated hemoglobin interval, since for most episodes of *High-Weekend-Glucose* there is no corresponding *Glycated Hemoglobin* measurement. We call this example the *Weekend-Glucose-and-Glycated Hemoglobin* pattern.

In this regard, CAPSUL patterns differ from intersections of nonconvex intervals. If we considered any intersection of nonconvex intervals to be a pattern, then this *Weekend-Glucose-and-Glycated Hemoglobin* pattern example could be expressed, since we could check if the two nonconvex intervals were aligned so that the glycated hemoglobin values occurred at the same time as every other month of high weekend glucose. CAPSUL patterns represent a subset of such patterns; CAPSUL patterns are those in which there exists a one-to-one correspondence between convex sub-intervals of the two (or more) nonconvex intervals. In such cases, the convex subintervals can be paired up exactly, and each pair represents one component of a periodic pattern.

For the *Weekend-Glucose-and-Glycated Hemoglobin* pattern, the best we can do using CAPSUL is to create two separate patterns, one in which the average weekend glucose constraints are satisfied, and a second pattern consisting of repeating glycated hemoglobin measurements for 1 year. We combine these two repeating patterns into a single linear pattern in which the two separate components overlap, but we cannot constrain the individual episodes of high glycated hemoglobin to occur during every other month in which the average weekend glucose is high.

2.3. Expressivity

Examples such as the previous one, which cannot be expressed in CAPSUL, lead us to ask: what is the set of patterns that can be expressed using CAPSUL? The following matrix representation of a periodic pattern provides a more precise demonstration of the expressive power of CAPSUL. It is important to note that this matrix representation is merely a simulation of the CAPSUL language; in reality, CAPSUL works completely within the single component framework and cannot express patterns consisting of interleaved nonconvex intervals. Consider a simple repeating pattern of daily occurrences of high blood glucose followed by an insulin injection. Using CAPSUL, this pattern consists of a linear pattern of high blood glucose and insulin injection; this linear pattern would then be used as a single repeating component of a periodic pattern. Using an intersection of nonconvex intervals, this pattern would consist of a series of high blood glucose occurrences, intersected with a series of insulin injections. We can simulate the CAPSUL language using the matrix representation of this pattern as follows.

Each CAPSUL periodic pattern can be visualized in an *n*-dimensional matrix, as in Figure 4, where *n* refers to the number of linear pattern components that comprise the single repeating component of a particular periodic pattern. This matrix representation is an extension of one inspired by Lina Khatib [Khatib, 1994]. In this matrix, each element represents an *n*-tuple of linear components; if the *n*-tuple satisfies the local and global constraints of the linear pattern, then the point is instantiated as a interval of the linear pattern. (Note that these linear patterns are recognized and instantiated before considering the constraints of the periodic pattern). Any set of *m* distinct, instantiated points in this space represents an *m*-tuple of single repeating components, where *m* is the cardinality of the periodic pattern in question. If this *m*-tuple of repeating components satisfies all periodic constraints associated with the periodic pattern, then the set is instantiated as a periodic pattern.

There are at least three interesting properties to note about the set of elements that define a pattern in this matrix representation. First, the CAPSUL syntax does not require the set of intervals that comprise a periodic pattern

		Occurrences of High Blood Glucose						
		1	**2**	**3**	**4**	**5**	**6**	**…**
Occurrences of Insulin injection	**1**	**False**	**False**	**False**	**False**	**False**	**False**	
	2	**True**	**False**	**False**	**False**	**False**	**False**	
	3	**False**	**True**	**False**	**False**	**False**	**False**	
	4	**False**	**False**	**True**	**False**	**False**	**False**	
	5	**False**	**False**	**False**	**True**	**False**	**False**	
	6	**False**	**False**	**False**	**False**	**False**	**False**	
	…							

Figure 4. A matrix representation of the periodic pattern "High Blood Glucose before Insulin Injection once a day". Note that the pattern begins at the second occurrence of insulin injection, which corresponds to the first occurrence of high blood glucose. The highlighted diagonal represents the pairs of occurrences that satisfy the temporal and value constraints of this pattern. This matrix representation is an extension of the one developed by Khatib [1994].

to consist of consecutive matrix elements. The geometric location of the elements within the matrix is entirely determined by the user-specified constraints; therefore, unless the pattern specification requires that there be no gaps between successive intervals, the matrix elements that comprise the pattern need not be consecutive.

Second, if the single repeating components of periodic patterns are not required to be non-overlapping, the periodic pattern can consist of elements that lie in the same matrix row or column. Usually, we think of a periodic pattern as consisting of a set of points that lie on a sub-diagonal of the matrix (which corresponds to a pattern in which no two periodic components contain the same linear component), but the imposition of such a constraint is left up to the user. Note that allowing the repeating components of a pattern to contain the same linear components is not the same as allowing different types of repeating components in the same periodic pattern. The definition of the repeating component remains constant for all elements of the matrix. Because the linear patterns can be identified and instantiated before considering periodic components, CAPSUL cannot express patterns such as the *Weekend-Glucose-and-Glycated Hemoglobin* pattern previously discussed: that pattern required two different types of linear components within a periodic pattern specification.

Finally, constraints must be evaluated by comparing pairs of single repeating components rather than considering an entire *m*-tuple at once. That is, all of the constraints in the CAPSUL syntax can be evaluated linearly for a set of elements; the CAPSUL interpreter can step down the set of (temporally ordered) intervals and evaluate the constraints in one pass. For example, the gap constraint defines one relation (which may be a disjunction of relations) to be evaluated for each consecutive pair of intervals. Once the constraint has been evaluated for a pair of intervals, the interpreter never revisits the intervals to evaluate this constraint. Thus the running time of the constraint-satisfaction, given a set of sorted elements is O(*m)*. This is called the linearity characteristic of CAPSUL constraints, and the CAPSUL constraints were designed specifically to maintain this characteristic.

3. Knowledge Acquisition

To complete the temporal abstraction task, Résumé requires a domain-specific knowledge base of concepts related to the domain. The four types of concepts in a Résumé knowledge base are parameters (e.g., *glucose*), contexts (e.g., *diabetes*), events (e.g., *platelet transfusion*), and patterns (e.g., *Platelet-Half-Life*). The interface to the Résumé knowledge base is a graphical tool through which domain experts can create, modify, visualize, and maintain the Résumé knowledge base; this graphical tool was created using the domain-independent Protégé set of tools [Musen et. al, 1996]. While the Protégé system is domain independent, as is the Résumé KA tool, in practice the current Résumé KA tool that we are using has been designed for the medical domain. The Résumé knowledge base, as visualized in the KA tool, is organized in an IS-A class hierarchy. The KA tool, developed using the Protégé system, provides an interface to view and modify the Résumé knowledge base. The main window contains frames for the four main classes of knowledge: Parameters, Contexts, Events, and Patterns; in each frame is listed the names of temporal instances of the corresponding class. When the user selects a temporal instance and clicks "edit", a new window appears with the definition of that object. The slots of the windows used to define patterns closely parallel the syntax of CAPSUL. A domain expert then uses this KA tool to create and modify an ontology of terms and concepts that applies to his/her specific domain, such as oncology. In order to maintain knowledge that is specific to different medical domains, we have created a separate knowledge base for each of our domain areas.

3.1. Examples

Figure 5 shows how the domain expert uses the KA tool to define how to abstract a concept such as *Low-Platelet-State* from individual platelet counts. The user can map ranges of platelet values to symbolic qualitative terms.

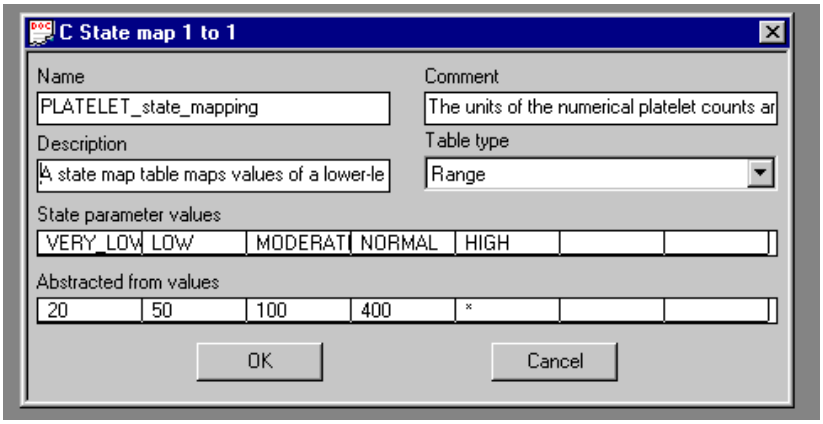

Figure 5. A Platelet-State mapping function. This table maps values of the numerical platelet-count parameters onto the symbolic values of the Platelet-state parameter. Such mappings are at the core of the temporal abstraction process. For the platelet parameter, a platelet count below 20 corresponds to a symbolic value of VERY_LOW, while a platelet count between 20 and 50 is considered LOW. Any value above 400 would be considered HIGH.

(e.g., A platelet count below 10 corresponds to an abstraction of *Very_Low* platelet-state) Using this definition, the user can then define more complicated abstractions that depend on the level of Platelet-state. An important feature of any Protégé KA tool is that any element of the ontology, once defined, is reusable in any other part of the ontology.

For example, the domain expert can use the higher level concepts of low and high platelet values in the definition of the *Platelet-Half-Life* pattern from Figure 2. Recall that there are three components of this linear pattern: *Platelet-transfusion* (the occurrence of a transfusion event), *High-platelet-state* (a high platelet count measured immediately after the transfusion), and *Low-platelet-state* (a low platelet count, indicating that the level of platelets has fallen). By selecting any of the three components listed in a "linear components" frame and clicking "edit", a new window containing slots for local-level constraints appears, allowing the user to apply local constraints to any of these components. A separate frame is used to define the other constraints of this pattern, such as global constraints that specify that the interval of *High-platelet-state* must occur immediately (defined as: within 1-3 hours) after the transfusion, and that the *Low-platelet-state* must occur some time after the *High-platelet-state*. Given these constraints, which define the linear order of the components, the linear sequence of components that defines the half life can be defined. A slot labeled "output value" is where the actual computation of half life occurs; the output value of a pattern is the value that is assigned to the pattern abstraction. In this case it is half of the duration of the pattern, where the duration is the time elapsed between the *High-platelet-state* and *Low-platelet-state* intervals.

As a third example of using the KA tool, Figures 6 and 7 show the knowledge acquisition screens of the definitions of the constraints associated with the *Recurring Weekend Glucose Monitoring* pattern described earlier. First, the repeating interval *Weekend* must be defined. This temporal object is then plugged into a global period-constraint of the linear pattern *Weekend-Glucose-Monitoring* (this linear pattern serves as the repeating component of the larger periodic pattern). Recall that the periodic constraints of this periodic pattern are a cardinality constraint, a period constraint, and a cluster constraint. The cluster constraint specifies that during the pattern, there must exist 2-5 consecutive weekends for which the value of the repeating component is *Very_Low*. The graphical specification of this cluster constraint is shown in figure 6. One of the constraints on this sub-pattern is a gap constraint of 7 days, shown in figure 7. This gap constraint requires that the weekends of the sub-pattern must be consecutive.

To facilitate the knowledge acquisition process, the knowledge base ontology differs slightly from the knowledge base used by the Résumé interpreter. The filtering process that converts the knowledge base as defined by the user into the knowledge base to be interpreted by Résumé handles small but important tasks such as adding back pointers between temporal objects (so that each temporal object points to both what it is abstracted from as well as what it is abstracted into). The filter also restructures the class hierarchy to eliminate intermediate classes that are useful for knowledge acquisition but not necessary (and in fact, inefficient) for the detection process.

3.2. KA Evaluation

To test the knowledge acquisition process as well as the usability of the CAPSUL language, we have worked with two domain experts with moderate computer experience to create knowledge bases for two medical domains, diabetes and oncology. The domain experts have already shown their capability for independent use of the KA tool without the periodic pattern frames [Shahar, Chen, et. al., 1998b]. Our domain experts tried to use the enhanced KA tool with the help of a knowledge engineer. The process has proved very useful both for domain experts as well as

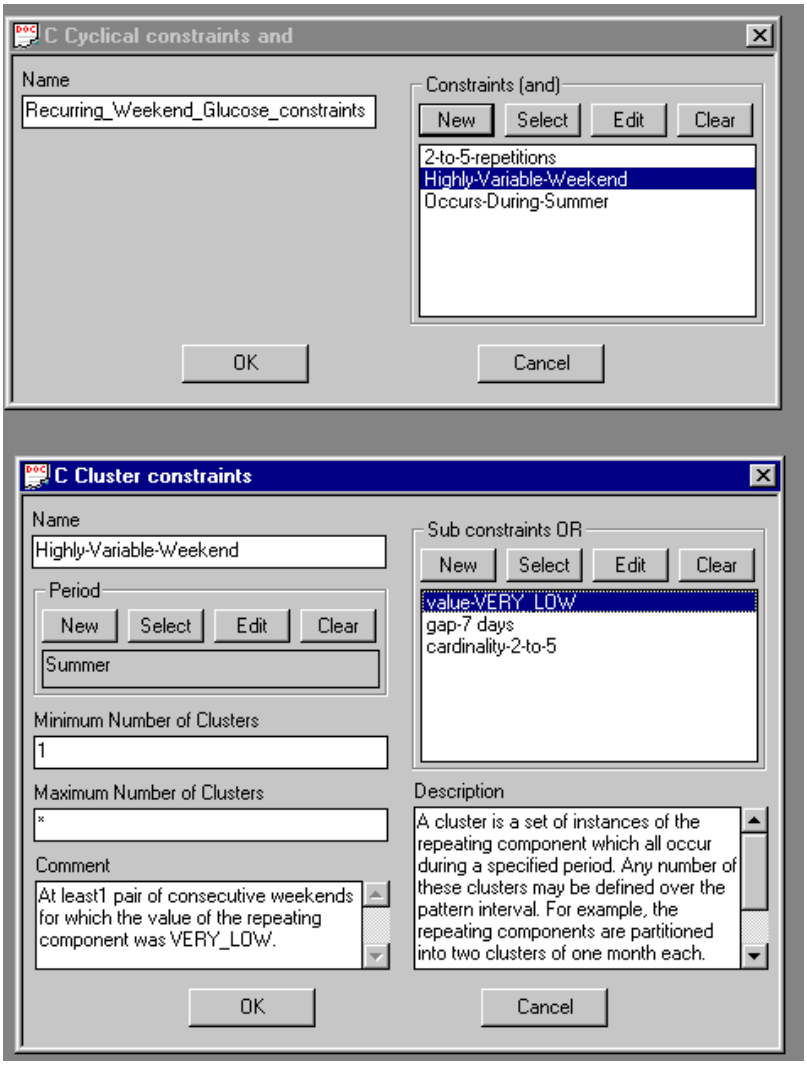

Figure 6. Definition of Periodic Constraints. The top window shows the definition of the set of periodic constraints associated with the pattern Recurring-Weekend-Glucose-Control. The set contains a cardinality constraint, a period constraint, and a cluster constraint. The period, or relevant scope, of the pattern is Summer. The bottom window shows the specification of a cluster constraint. The sub-pattern consists of at least one pair of consecutive weekends during which the value of the repeating component was *Very_Low*. The 7 day gap constraint ensures that the intervals are consecutive, and the cardinality constraint ensures that the cluster contains at least 2 such weekends. The definition of the 7-day gap constraint is shown in figure 7.

knowledge engineers; we have learned from them the features and bugs of our KA process. For the bone marrow transplant (BMT) knowledge base, our domain expert worked with us to create a knowledge base consisting of over 70 parameters that are typically of interest for a patient who has received a bone-marrow transplant, such as White Blood Cell count and Platelet Count. For each parameter, we not only had to define the term, but we also mapped the numerical values of the parameters as encountered in a database into symbolic values such as *Platelet-state*, as in figure 5. We also have entered several patterns in this domain, such as *Platelet-half-life*.

The users who tested the KA tool liked several features of the graphical tool. The Windows/NT design of the windows allows for intuitive browsing and filling of the slots in the various windows and frames. The users could quickly find the windows for the objects they wanted to modify. Within windows, the users were almost always able to quickly understand the meanings of different slots. For simple parameter definitions, the users were able to use the tool without assistance to modify existing definitions of ranges or symbolic values. Although the users found pattern specification a bit more complex, the disjunctive normal form of the pattern specification was surprisingly intuitive for our users with little explanation. The structuring of a pattern definition in an AND-OR tree made sense to them. With regard to the expressive power of our pattern specification language, we found that CAPSUL could specify almost all of the patterns in which our users expressed interested. In the BMT ontology, we were able to define patterns that describe the simultaneous toxicity levels of several organ systems. For example, the pattern

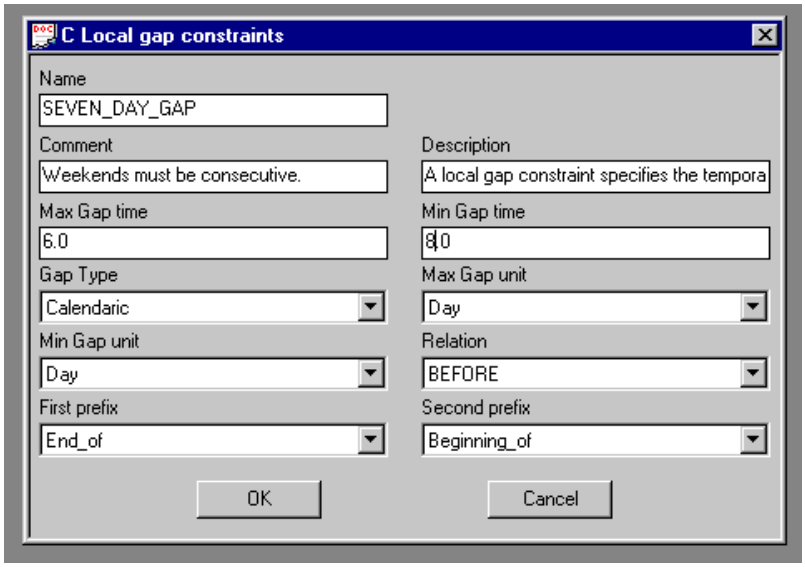

Figure 7. Definition of a gap constraint. This constraint, which is part of the Highly-Variable-Weekend sub-pattern, specifies that all weekends that are part of the sub-pattern must be 6-8 days apart. We use a gap measured in days rather than weeks to enhance flexibility. Since the periodic components are already required to occur during a weekend, this gap constraint is enough to ensure that the sub-pattern will consist of consecutive weekends. The gap type is calendaric: the gap must be measured using calendar days.

Multiple-Organ-Toxicity consists of overlapping intervals of Renal, Liver, and Myelotoxicity levels. Toxicity levels for each organ are characterized by different parameters: Renal toxicity is measured by Creatinine, Liver toxicity is governed by Total Bilirubin and Alkaline Phosphatase, and Myelotoxicity is defined by White Blood Cell count and Platelet-state. Thus the pattern of *Multiple-Organ-Toxicity* summarizes the values of 5 different raw level data elements, which would be very difficult for a person to try to find by hand in a patient data file.

Users found two primary types of constraints that CAPSUL could not express. The first area is that of statistical constraints. The domain experts were often interested in asking whether the abstractions of certain parameters followed a particular distribution. For efficiency reasons, CAPSUL has only very simple statistical capabilities: CAPSUL can find abstractions of the mean and variance of parameters, but CAPSUL cannot fit distributions to sets of data. This goes along with the linearity characteristic of CAPSUL constraints: CAPSUL is limited to those constraints that can be computed by just one pass over a data set. The second area of difficulty arose when trying to incorporate the idea of ordinality into patterns constraints. Since the derivation of abstractions occurs in random order, it is currently impossible for CAPSUL to specify that a certain component is the *n*th of its kind in the data. Therefore, constraints such as "the first occurrence of *Normal-Total-Bilirubin*" cannot be currently incorporated into a pattern. Even if the interpreter were to find and sort all instances of *Normal-Total-Bilirubin* at every step of the abstraction process, it could not know whether an earlier interval of *Normal-Total-Bilirubin* would be found later in the data set. We intend to enhance the computational framework somewhat to enable constraints such as "First occurrence". Most of the other difficulties encountered by the users can be solved with a more extensive help file or tutorial that explains the meanings of the slots more intuitively for users with moderate computer experience. Users also had trouble dealing with the number of windows required and the number of slots per window. Because of the large number of slots associated with patterns, users often required the assistance of a knowledge engineer to input patterns in the KA tool.

Overall, however, the domain experts were satisfied with the ability of CAPSUL to express complicated patterns that are useful in the analysis of patient records. Furthermore, all patterns that were found to be theoretically express-able in CAPSUL were successfully entered into the knowledge base via the graphical KA tool, demonstrating the ability of the current Résumé/CAPSUL KA tool to support entry of CAPSUL expressions.

4. Data Analysis

With collaboration from the University of Chicago Bone Marrow Transplantation (BMT) Center, we are currently evaluating the pattern-matching Résumé interpreter by analyzing data sets from patients who have received bone marrow transplants. An extensive database was obtained from the University of Chicago that contained the full patient records of over 100 patients. Each patient was hospitalized for between 2 months and 2 years, and during the course of treatment, the patients had daily tests taken for about 70 parameters each time, out of a total of some 550 laboratory data types. These parameters fell into the following classes: hematological, drug levels, electrolytes, infection, metabolic, pancreatic, renal, and respiratory. The records also recorded the various blood transfusion that the

patients received during their hospitalization, the date(s) of transplant operation(s), and basic biographical information. Overall, each patient data set contains several thousand test values. The data was recorded during the period 1993-1996 using a Paradox (DOS-based) database. This database was first converted into a Microsoft Access database; from this format it was easily converted into a text-based format that could be interpreted by the Résumé interpreter.

With help from our domain expert, we selected the full patient records of three patients. We used the Résumé interpreter and the BMT knowledge base to derive a full set of abstractions for these patients. These abstractions included 3 types of higher-level parameters (states, gradients, and rates) as well as a few patterns. On average, the output of Résumé was about five times the size of the input; this means that each element of raw data was used in the derivation of about 4 higher-level abstraction intervals. The average running time for these files was on the order of five hours. With over 10000 output intervals, our domain experts could not hope to examine the text output of Résumé in order to analyze the output; instead, the output intervals were entered into an online visualization tool, a derivation of KNAVE [Shahar and Cheng, 1998] so that both the raw level data as well as the abstracted intervals could be explored. In this evaluation, we address two main questions: (1) how much time can we save by analyzing the patient data sets using Résumé; and (2) what insights can Résumé provide that are impossible to discover without the use of such a tool?

With regard to the first question, the sheer size of the data set made it difficult to analyze the patient data without the help of computer software. By reformatting and creating charts of small, selected parts of the data set, the expert could identify trends in the values of individual, raw-level parameters. Within 2-3 hours, the expert was able to reformat the data and plot it using a spreadsheet package. Thus, for the task of looking only at individual raw data values and identifying extrema, using a spreadsheet program such as Excel may actually be easier than using Résumé. Résumé is most useful when identifying trends not only in raw data, but in higher level abstractions, and when identifying trends in more than one parameter at once. Many interesting patterns that involve more than one parameter are tedious to find by hand. For example, consider the earlier example of *Liver-toxicity*, a high level concept that is derived from Alkaline Phosphatase and Total Bilirubin. When both of these parameters are high at the same time, Résumé derives the abstraction of *Liver-toxicity*. To find this pattern by hand in the patient data record, the expert must (mentally or using a spreadsheet) align the graphs for Total Bilirubin and Alkaline Phosphatase to see if and when they both are high at the same time. For tasks such as these, the Résumé interpreter greatly speeds up data analysis.

In other cases, Résumé not only speeds up data analysis, but also makes possible the detection of patterns that could not be found at all by looking only at raw level data. Patterns that involve the trends in one or more parameters over years, for example, or those that require complex intermediate levels of abstraction (that cannot be easily derived mentally) are made tractable by Résumé. For example, consider a pattern of quickly changing Glucose levels at least once a week for more than half of the weekends in a year. Such a pattern is useful to both patient and physician in order to identify lifestyle patterns that cause rapid changes in Glucose level. To find this pattern by hand, an analyst would have to plot the Glucose levels, derive the *Glucose-state* values, count up the weekends in which the value of *Glucose-state* changed sharply (either decreasing or increasing), and determine if this occurred more than 26 times within any 12 month period in the database. It is impractical for a physician to detect such a pattern without system such as Résumé.

5. Related Work

Most work in the area of temporal reasoning involving interval-based representations of time refers to [Allen,1984], who defined an interval algebra of 13 basic binary temporal relations for convex time intervals. These temporal relations are the basis for the qualitative temporal relations allowed in CAPSUL. [Ladkin, 1986a] showed that the number of relations between unions of nonconvex intervals is at least exponential in the number of convex subintervals, a result that proved useful to us. Rather than allowing any possible union of nonconvex intervals, CAPSUL allows only those unions in which the convex subintervals can be matched one-to-one. In another work, [Ladkin 1986b] dealt with the representation of calendar-based time. He developed a formal representation of repeating intervals such as *Mondays* that we referred to when incorporating repeating events. In a paper dealing with event representation calendar time, [Clifford and Rao, 1988] disallowed the use of weeks as a time unit, unlike us. Weeks can overlap with months and do not fit into the natural subset chain of <seconds, minutes, hours, days, months, years>.

Several people have proposed frameworks similar to ours in which to define repeating events. Cukierman and Delgrande [1996] studied calendar-based repeating temporal objects. In this language, a series of *n* repeats is represented as follows: $<r_1,g_1,r_2,g_2,...,r_n, g_n>$, where *r* refers to a repeat (an instance of the repeating event) and *g* refers to a gap. An important component of the Cukierman language is the pattern of gaps $< g_1, g_2,..., g_n >$, which allows each gap to be the same, to be different, or to be defined probabilistically. More recently, Cukierman and Delgrande [1997] introduced a revised framework for representing repetition with the notion of a *time loop* that is made up of a cycle (or a repeat) and the relations between cycles. This loop may be "flattened" to form a "closed-view representation" of the sequence of instances. In the Cukierman- Delgrande language, loops are represented in closed time: A pattern of Math class on Mondays and English class on Tuesdays is represented as the convex interval <Math before English> that repeats weekly. Representing this same pattern in Ladkin's language involves correlating the two nonconvex intervals

<Math on Mondays> and <English on Tuesdays>. Cukierman's representation was useful to us; CAPSUL operates entirely under this closed-view scheme.

The temporal-repetition language presented by Terenziani [1997] contains many important distinctions that we used in our language. For example, Terenziani distinguishes between local and global constraints (where *local* constraints are applied to instances of the repeating event, and *global* constraints apply to the overall pattern). Morris and Khatib [1998a] discussed a set of temporal constraints in which gaps can be specified either by a fixed length or by a probability distribution. More recently, Morris and Khatib [1998b] have developed the matrix of binary relations to represent the temporal relations between every pair of nonconvex intervals in a pattern. This matrix idea, as discussed in section 2.3, helped us to clarify the semantics of allowing users to specify only one relationship for all pairs of intervals.

6. Conclusion

In this paper, we have shown the ability of the Résumé system to detect repeating trends in time oriented medical data. The pattern matching module of Résumé, which consists of a temporal pattern specification language (CAPSUL), a KA tool, and an interpreter, provides an invaluable tool for efficiently summarizing large data sets. The CAPSUL language allows a domain expert to express complicated patterns involving multiple levels of abstraction of parameters. The graphical KA tool allows the user to enter such patterns into the Résumé knowledge base, and the Résumé interpreter then finds these patterns in data sets. We are currently conducting an evaluation of the usability of the KA tool as well as the performance of the Résumé interpreter. Initial results indicate that experts with moderate computer experience are able to use the KA tool with help from a knowledge engineer, and that the Résumé interpreter can correctly derive abstractions that would otherwise be unattainable to data analysts.

In the future, we plan to continue and expand upon our evaluation in the diabetes domain, as well as to study the performance of the interpreter on larger and more data sets. We also plan to work more on the theoretical aspects of CAPSUL, including the formulation of the exact limitations and expressive powers of the language. Once the Résumé system has been tested and evaluated, it can be combined with visualization tools such as KNAVE to provide a new level of analysis of clinical data.

7. References

Allen, J. Towards a general theory of action and time. *Artificial Intelligence* 23(2): 123–154, 1984.

Chakravarty, S. and Y. Shahar. A constraint-based specification of periodic patterns in time-oriented data. In *Proc. of the Intl. Workshop on Temporal Representation and Reasoning* (*Time)-99*, pp. 29-40. IEEE Press. 1999.

Clifford, J. and A. Rao. A simple, general structure for temporal domains. *Temporal Aspects in Information Systems*, pgs 17-28. IFIP: 1988.

Cukierman, D. and J. Delgrande. Characterizing Temporal Repetition. In *Proc. of the Intl. Workshop on Temporal Reasoning and Representation (TIME) -96*, pages 80-87. IEEE Press. 1996.

Cukierman, D. and J. Delgrande. Towards a formal characterization of temporal repetition with closed time. In *Proc. of the Intl. Workshop on Temporal Reasoning and Representation (TIME)-98*. pgs 140-147. IEEE Press. 1998.

Khatib, L. Reasoning with Non-convex Time Intervals. PhD. Thesis. Florida Institute of Technology. September 1994.

Ladkin, P. Time representation: A taxonomy of interval relations. In *Proc. of the AAAI-86*, pages 360-366, 1986a.

Ladkin, P. Primitives and Units for Time Specification. In *Proc. of the AAAI-86*, pages 354-359, 1986b.

Morris, R. and L. Khatib. Quantitative Structural Temporal Constraints on Repeating Events. In *Proc. of the Intl. Workshop on Temporal Reasoning and Representation (TIME)-98*, pgs 74-79. IEEE Press. 1998.

Morris, R. and L. Khatib. Periodic and Repeating Events. Florida Inst. of Technology. July 17, 1996. pgs. 1-17.

Musen, M.A., S.W. Tu, A.K. Das, and Y. Shahar. EON: A component-based approach to automation of protocol-directed therapy. *Journal of the American Medical Association*. 3(6): 367–388, 1996.

Shahar, Y. and M. Musen. Knowledge-based temporal abstraction in clinical domains. *Artificial Intelligence in Medicine*; 1996. 8(3): pgs 267-298.

Shahar, Y. A framework for knowledge-based temporal abstraction. *Artificial Intelligence* 90(1–2): 79–133, 1997.

Shahar, Y. Dynamic Temporal Interpretation Contexts for temporal abstraction. *Annals of Mathematics and Artificial Intelligence*, 1998; 22(1-2): pgs 159-192.

Shahar, Y. and C. Cheng. Intelligent Visualization and Exploration of Time-Oriented Clinical Data. In press for *Topics in Health Information Management* (THIM).

Shahar, Y., H. Chen, D. P. Stites, L. Basso, H. Kaizer, D. M. Wilson, and M. A. Musen. Semiautomated Acquisition of Clinical Temporal-abstraction Knowledge. In press for *Journal of the American Medical Informatics Association* (JAMIA) 6(6) 99.

Terenziani, P. Qualitative and Quantitative Temporal Constraints About Numerically Quantified Periodic Events. In *Proc. of the Intl. Workshop on Temporal Reasoning and Representation (TIME)-97*, pgs 94-101. IEEE Press. 1997.

Appendix. BNF syntax for CAPSUL

pattern: <linear pattern> | <periodic pattern>
linear pattern: <name> <disjunctive-linear-component-set>$^{+}$ <value> <output context> <necessary context>
periodic pattern: <name> <disjunctive-periodic-component-set>$^{+}$ <value> <output context> <necessary context>

disjunctive-periodic-component-set: <pattern-component> <disjunctive-periodic-constraint-set>
disjunctive-periodic-constraint-set: <name> <conjunctive-periodic-constraint-set>$^{+}$
conjunctive-periodic-constraint-set: <name> <periodic-constraint>$^{+}$
periodic-constraint: <cardinality-constraint> | <value-constraint> | <global-gap-constraint> | <local-gap-constraint> | <cluster-constraint>
cardinality-constraint: <minimum-number-of-repetitions> <maximum-number-of-repetitions>
value-constraint: <minimum-difference-between-instances> <maximum-difference> | <qualitative-value-relation>
global-gap-constraint: <qualitative-temporal-relation>
local-gap-constraint: <minimum-time-difference> <maximum-time-difference> <gap-type> <qualitative-temporal-relation>
cluster-constraint: <period> <minimum-number-of-clusters> <maximum-number-of-clusters> <disjunctive-sub-constraint-set>$^{+}$
disjunctive-sub-constraint-set: <conjunctive-periodic-constraint-set>$^{+}$

disjunctive-linear-component-set: <pattern-component>$^{+}$ <disjunctive-linear-constraint-set>$^{+}$
disjunctive-linear-constraint-set: <conjunctive-linear-constraint-set>$^{+}$
conjunctive-linear-constraint-set: <linear-constraint>$^{+}$
linear-constraint: <gap-constraint> | <interval-constraint> | <period-constraint> | <value-constraint>
gap-constraint: <first-point> <second-point> <minimum-time-difference> <maximum-time-difference> <gap type>
interval-constraint: <first-interval> <second-interval> <qualitative-temporal-relation>
period-constraint: <period>
linear-value-constraint: <first-interval> <second-interval> <value-constraint>

qualitative-temporal-relation: <before> | <after> | <starts> | <ends> | <within> | <meets> | <overlaps> | <equal>
qualitative-value-relation: <greater-than> | <less-than> | <greater-than-or-equal> | <less-than-or-equal> | <equal>
period: <time-period> | <repeating-interval>
time-period: <name> <period-type> <minimum-amount-of-time> <maximum-amount-of-time>
repeating-interval: <name> <hour-start> <hour-end> <week-start> <week-end> <year-start> <year-end> <minute-start> <minute-end> <day-start> <day-end> <month-start> <month-end> <global-start-time> <global-end-time>

pattern-component: <proposition> <local-constraint>$^{+}$
proposition: <linear-pattern > | <periodic-pattern> | <parameter> | <context> | <event>
local-constraint: <duration> | <value> | <earliest-starting-time> | <latest-starting-time> | <earliest-ending-time> | <latest-ending-time>
value: <true> | <function> | <mapping-function>