Representing Temporal Knowledge for Case-Based Prediction

Martha Dørum Jære[1], Agnar Aamodt[2], Pål Skalle[3]

[1]Borak S.L., Ronda de Poniente N4, Tres Cantos, CP 28760, Madrid, Spain
[2]Artificial Intelligence Research Institute, IIIA, Spanish Council for Scientific Research, CSIC, 08193 Bellaterra, Barcelona, Spain.
[3]Department of Petroleum Technology, Norwegian University of Science and Technology, NO-7491, Trondheim, Norway.

Abstract. Cases are descriptions of situations limited in time and space. The research reported here introduces a method for representation and reasoning with time-dependent situations, or temporal cases, within a knowledge-intensive CBR framework. Most current CBR methods deal with snapshot cases, descriptions of a world state at a single time stamp. In many time-dependent situations, value sets at particular time points are less important than the value changes over some interval of time. Our focus is on prediction problems for avoiding faulty situations. Based on a well-established theory of temporal intervals, we have developed a method for representing temporal cases inside the knowledge-intensive CBR system Creek. The paper presents the theoretical foundation of the method, the representation formalism and basic reasoning algorithms, and an example applied to the prediction of unwanted events in oil well drilling.

1 Introduction

Most current CBR systems represent episodes as distinct snap-shots in time. Wherever temporal relationships may exist between parameters in a case, they are either ignored or implicitly handled within the reasoning algorithms. Two current trends are strengthening the need for explicit representation of temporality. Firstly, CBR is continuously addressing increasingly challenging problems, and in particular problem solving in a real world context. Examples are process supervision and control [18, 21], event prediction from trends [18, 19], and temporal planning [5]. Even "classical" problems, such as diagnosis and treatment, involve extensive temporal reasoning when moving beyond the simple classification approach [14, 22]. Secondly, CBR systems, as well as decision support systems in general, are becoming more interactive and user-transparent. The trend is away from self-contained problem solvers and towards user-interactive assistants, where the sequentiality of interactions often becomes an important piece of information. Both these trends call for an explicit representation of temporal relationships.

Creek [2, 24] is a CBR system that integrates cases with general domain knowledge within a single semantic network. Each feature and feature value of a case is a concept in the semantic network. They are interlinked with other concepts (which may be case features or not) through semantic relations specified by the general domain model. Typical relations include subclass and instance relations, part-subpart, process-subprocess, causal and functional relations, as well as particular domain relations (e.g. "has color" or "owes money to"). The general domain knowledge is used as a model-based reasoning support to the case based reasoning processes Retrieve, Reuse, and Retain [1]. Case retrieval, for example, becomes partly an explanation process, in which the initial matching based on index links are justified or criticized. Reuse becomes a process of explaining the adaptation of a past case within the context of the current problem, and Retain the process of explaining/justifying what to retain from a problem just solved. The research described here extends Creek's representation to include temporal relationships, which in turn are utilized by the explanatory mechanisms that underlie the system's reasoning method [12]. Creek addresses problem solving and learning in weak theory, open, and changing domains, such as medicine and most engineering domains. Hence, relations are generally uncertain, and most concepts are described by typical features rather than universally quantified assertions.

In addition to the theoretical and methodological interesting aspects of temporal cases, another motivation for developing a temporal representation within this system comes from a type of application we have been addressing over the last years: Decision support for oil well drilling tasks. While our past work on diagnosis and treatment of unwanted events have utilized cases in order to find out what did happen in the past (such as the cause of a drill string getting stuck) [3, 24], we wanted to utilize the growing case base as a resource for avoiding unwanted events in the future. As the basis for our method, we have adopted James Allen's theory of temporal intervals [4]. This is a well worked-out theory for temporal representations, accompanied by an efficient inference method.

The paper first briefly reviews some of the related research in representing and reasoning about temporality in CBR. This is followed by a summary of James Allen's theory on temporal intervals. The subsequent section introduces the problem of predicting unwanted events in an industrial process, exemplified by the oil drilling domain. Section five presents the temporal representation in our system, and section six describes how the representation is utilized for matching of temporal intervals. Then an example of case-based prediction of a drill-sticking situation is given. The conclusion sums up the results and discusses strengths and weaknesses of the method.

2 Temporal Reasoning and CBR

Early AI research on temporal reasoning makes a distinction between *point-based* and *interval-based* approaches (also referred to as instants-based and period-based approaches [26]). Many well-known theories and systems, such as the Situation Calculus and the early Time Specialist [13], are based on instants as the temporal primitive. Allen's theory of temporal intervals [4], on the other hand, advocates that the interval is the appropriate temporal primitive for reasoning about time.

Although reports of research on temporal representations for CBR in general are scarce, some interesting results have been published. Jaczynski and Trousse propose a method based on so-called time-extended situations [10]. Example applications developed are plant nutrition control and prediction of user behavior for Web navigation. Temporal knowledge is represented as temporal patterns, i.e. multiple streams of data related to time points. The representation holds cases as well as general knowledge, which both are taken into account during retrieval.

Melendez and his group suggest a method for supervising and controlling the sequencing of process steps that have to fulfill certain conditions [16, 18]. Their main domain is the control of sets of recipes for making products, such as plastic or rubber pieces, from a set of ingredients. A case represents a recipe, and the temporal problem is the control of a set of recipes – a batch – in order to fulfill process conditions and achieve a production goal. A deviation from a normally operating condition is called an event, and consists of actions and reactions. Together, the events represent significant points in the history of a product. An episode contains information related to the behavior between two consecutive events. The retrieval method first matches general conditions such as the initial and final sub-processes, and then the initial conditions of the corresponding episodes.

Hansen [8] presents a method for weather prediction in which a point-based representation of whether observations are utilized in a combined case-based and fuzzy set system. Time is included in the similarity metric, together with other weather parameters. Branting and Hasting's knowledge-intensive CBR system for pest management [6] incorporates a method called "temporal projection". The method aligns two cases in time, by projecting a retrieved case forward or backwards in order to match on other parameters (such as the development stage of an insect).

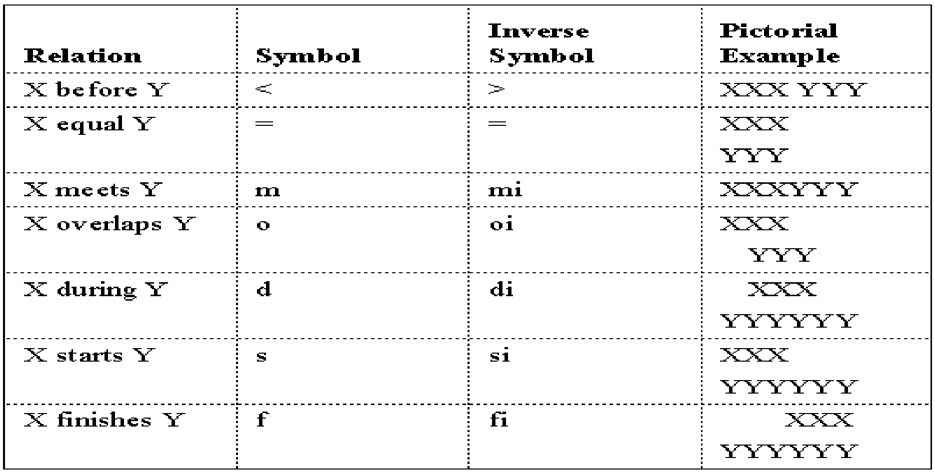
All the above systems are essentially point-based. Research on temporal reasoning in CBR that take an interval-based approach (in the sense of Allen) is scarce. McLaren and Ashley [16] use temporal intervals in the matching of cases for engineering ethics problems. Temporal relations are used for checking time consistency among facts that match on other criteria. Temporality is one of several matching criteria in the second stage of the case matching process. In the method we present in the current paper, temporal relations have a more central role, reflected in the different and more explicit way this knowledge is represented, as well as the different reasoning method.

The type of problems addressed by our research, where we need to deal with large and complex data sets, as well as the explanatory reasoning methods underlying our CBR approach, strongly indicate that a qualitative, interval-based framework for temporal reasoning is preferable. At least this is our hypothesis.

3 Allen's Temporal Intervals

James Allen [4] introduces a way to represent temporal knowledge in an interval-based temporal logic. An important characteristic of Allen's intervals is that they are decomposable; they can always be decomposed into sub-parts, including time points. Intervals may be open or closed. If they are closed they meet each other exactly, if they are open there will be a point between them that has an empty state when neither of them are true. Intervals can be described in a hierarchy connected by temporal relations: *If interval X is during an interval Y, and P holds during X, then P holds during Y.* Within a "during" hierarchy propositions can hence be inherited. This type of hierarchy allows reasoning processes to be constrained so that irrelevant facts are not considered.

Table 1. The thirteen possible relationships. Adapted from [4].



Relation	**Symbol**	**Inverse Symbol**	**Pictorial Example**
X before Y	<	>	XXX YYY
X equal Y	=	=	XXX YYY
X meets Y	m	mi	XXXYYY
X overlaps Y	o	oi	XXX YYY
X during Y	d	di	XXX YYYYYY
X starts Y	s	si	XXX YYYYYY
X finishes Y	f	fi	XXX YYYYYY

Allen proposes a model of thirteen ways in which an ordered pair of intervals can be related with mutually exclusive temporal relations. The temporal relations are shown in Table 1, where the columns of Symbol and Inverse Symbol define the thirteen relations. In the rightmost column future time is to the right and past time to the left. Together, these relations can be used to express any relationship that can exist between two intervals. They are maintained in a network where nodes represent individual intervals. Allen proposes an algorithm that creates missing temporal relations (temporal constraints) from existing knowledge. The set of transitivity relations on all possible subsequent pairs of temporal primitives is presented in Table 2. When new temporal relations are entered in a temporal network all possible temporal relationships can be derived by use of the transitivity table.

In order to reduce the complexity of a growing number of temporal constraints, Allen proposes a generalization method using *reference intervals*. A reference interval represent a cluster of intervals where the temporal constraints have been computed.

4 Prediction of Unwanted Events

We are addressing the prediction problem within oil well drilling, and our example task is to avoid the situation that a drill string gets stuck in the borehole, and stops the drilling process. Due to high hourly cost of drilling operations and the long time involved in freeing a stuck pipe, this is one of the most costly drilling problems [11, 23]. There is abundant literature on this topic describing how to identify a stuck-pipe situation and free it (e.g. [9]). Still, in order to deal with stuck pipe situations and to avoid them, extensive experience by the crew on the platform is needed. The situation when a driller or a system identifies that something may happen is a continuous process, where qualitative changes of particular parameters are strong indicators.

Table 2. Transitivity table for the twelve temporal relations (omitting "="). A, B, and C are the time intervals. The possible relations, r1 and r2, are listed in the first column and the top row, respectively. Table cells list the inferred relations from a combination of A r1 B and B r2 C. Adapted from [4].

B r2 C **A r1 B**	**<**	**>**	**d**	**di**	**o**	**oi**	**m**	**mi**	**s**	**si**	**f**	**fi**
“before” <	<	*no info*	< o m d s	<	<	< o m d s	<	< o m d s	<	<	< o m d s	<
“after” >	*no info*	>	> oi mi d f	>	> oi mi d f	>	> oi mi d f	>	> oi mi d f	>	>	>
“during” d	<	>	d	*no info*	< o m d s	> oi mi d f	<	>	d	> oi mi d f	d	< o m d s
“contains” di	< o m di fi	> oi di mi si	o oi dur con =	di	o di fi	oi di si	o di fi	oi di si	di fi o	di	di si oi	di
“overlaps” o	<	> oi di mi si	o d s	< o m di fi	< o m	o oi dur con =	<	oi di si	o	di fi o	d s o	< o m
“overlapped-by” oi	< o m di fi	>	oi d f	> oi mi di si	o oi dur con =	> oi mi	o di fi	>	oi d f	oi > mi	oi	oi di si
“meets” m	<	> oi mi di si	o d s	<	<	o d s	<	f fi =	m	m	d s o	<
“met-by” mi	< o m di fi	>	oi d f	>	oi d f	>	s si =	>	d f oi	>	mi	mi
“starts” s	<	>	d	< o m di fi	< o m	oi d f	<	mi	s	s si =	d	< m o
“started by” si	< o m di fi	>	oi d f	di	o di fi	oi	o di fi	mi	s si =	si	oi	di

We want our system to give warnings to the user when an unwanted event may be approaching. Two types of system states are defined: *alert state* and *alarm state*. A retrieved case should trigger an *alert state* if a matching past experience indicates an upcoming unwanted event. This happens if the past case was an actual stuck pipe situation, or a situation in which stuck pipe was indicated but avoided. An *alarm state* should be triggered when a seemingly unavoidable unwanted event is about to happen. The alert state is a warning about the alarm state, and ideally the alert state will be discovered in time to avoid the alarm state.

Figure 1 shows a simplified example of some of the parameter-curves in a stuck pipe situation. It shows the magnitude of three different parameters as a function of time. These parameters are continuously being logged by sensors in the well or on the platform. Assume a CBR system that has continuous access to the parameters, and whose task is to identify potential stuck pipe situations based on early indications. The system has two sources of knowledge, the general model and the case-base. The erratic pump pressure at measurement 1300 triggers the system, which starts to look for similar cases in its case-base, and finds two cases above the similarity threshold. Both cases indicate an upcoming occurrence of stuck pipe, and the system executes an alert state warning. One of the two cases contains a cause for the stuck, while in the other case – the best matching case - the cause was not established. The system tries to find a possible explanation for the cause of that case (the latter) within the general knowledge. The explanation structure contains the following explanation:

Erratic Pump Pressure *indicates* Flow Restriction in Annulus
Flow Restriction in Annulus *indicates* Unstable Wellbore
Unstable Wellbore *indicates* Solids Build Up
Solids Build Up *causes* Stuck Pipe

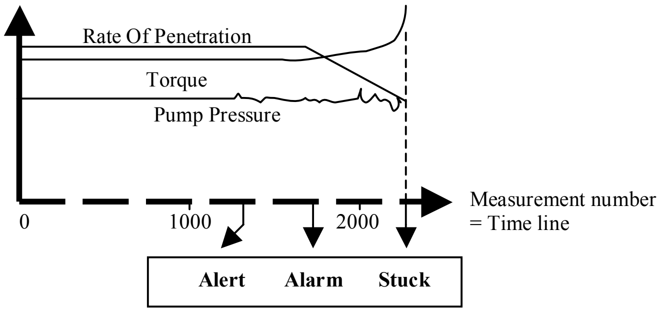


Figure 1. Temporal development of a stuck pipe case.

The system accepts Solids Build Up as the strongest supported hypothesis for the failure cause, and gives the user advice on how to avoid it on that basis. In addition it presents the other case whose match is above the threshold, as an alternative possible development to be validated by the user.

During this case, taken from a real drilling process, the drill actually got stuck at measurement 2200. Using a system as described, this event should have been avoided. An advantage of using a CBR system here, compared to existing trend analysis [11], is that the latter is based on single parameter analysis. The complexity of the interaction between a drilling process and its surrounding geological formation is too complex to be effectively handled with such a method.

5 Temporal Representation in Creek

Allen's approach is simple, transparent and easy to implement. It allows knowledge to be imprecise and uncertain which is necessary for problems in open domains. In our representation intervals are stored with temporal relationships inside cases. In the same way as Allen's reference intervals (see section 3), the cases restrict the computational complexity by limiting the amount of temporal relations resulting from transitivity rules.

From a cognitive perspective, the application of the transitivity rules leads to a high degree of regularity that becomes visible through the created relations. It is argued that this does not to correspond to the human, relative way of thinking about time [7]. An extended, more explanation-based way of reasoning about temporal relationships would be preferred. We have made an effort to combine the temporal knowledge inside a case with explanations in the general domain knowledge.

As an illustration of how temporality is represented in the Creek system, consider the following story:

> *When Juan was in the room, the room was tidy. Juan was there for ten minutes. Maria entered the room thirty minutes after Juan. Then the room was a complete mess and Maria's stereo was stolen.*

This story can be represented in a temporal network as shown in Figure 2. We can also formalize the representation as a structured frame with intervals, which in turn is contains their findings, as shown in Table 3. CaseX is related to the intervals, and the intervals are related to the findings, i.e. the observations.

Table 3. Frame object representation of a case and its intervals.

CaseX	Interval1	Interval2	Interval3
hasInterval Interval1	hasFinding JuanInRoom	hasDuration 30Min	hasFinding MariaInRoom
hasInterval Interval2	hasFinding RoomTidy		hasFinding RoomMessy
hasInterval Interval3	hasDuration 10Min		hasFinding StereoStolen

An example of how this frame-based representation looks in a network perspective is illustrated in Figure 3. This is how temporality is represented in Creek. The clarity and representational efficiency of using intervals within cases are clearly seen. The graphs shown are screen dumps from the Creek Knowledge Editor. Each relation in Creek has a corresponding inverse relation (e.g., has interval / interval of), and in the graphs the direction of a relation is from left to right, then top to bottom.

As illustrated in Figure 3, the temporal intervals are connected to each other with temporal relations: Interval1 (I1) *meets* Interval2 (I2) and Interval2 (I2) *meets* Interval3 (I3). An example of an inferred temporal relation is the relation from I1 to I3. When the two meets intervals are added to the system, it infers, by using the transitivity rules, that I1 is before I3. New intervals are incorporated according to the following procedure:

```
For every new interval that is added to the network
(1) Create a <has interval> relationship
(2) Create <has finding> relationships
(3) Create <Temporal Relation> relationships
(4) Infer new <Temporal Relation> relationships
```

There are methods for making intervals with or without dating. When we make a case with time-information, the temporal relations are made automatically (third step). For instance, if we make one interval that lasts from 12:00:00 until 12:10:00 and one that lasts from 12:30:20 until 14:20:30, the system will take care of creating the <*before*> relation between them. In the final step the system uses its transitivity rules to infer possible additional temporal relations between the intervals.

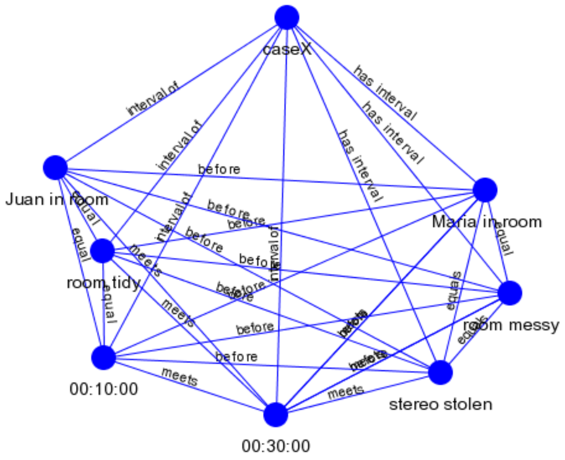


Figure 2. Illustration of the temporal network of a CaseX where all the findings are related to the other findings directly. We can see that even this very simple temporal situation will create a very dense network.

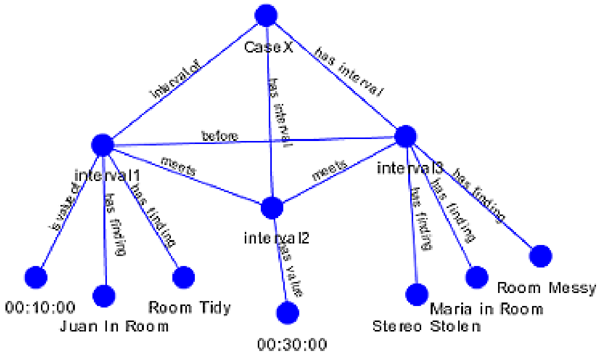


Figure 3. Illustration of the temporal network of CaseX. All the findings are related to CaseX via intervals. The temporal network is very small but still keeps all of the information of the network in Figure 2.

6 Temporal Paths and Dynamic Ordering

During case matching in the original (non-temporal) Creek system, three similarity values are computed: The *activation strength* is based on direct index matching, the *explanation strength* is based on similarity explained in the general domain model (where each relation has a certain explanatory strength), while the *matching strength* combines the two former into a resulting similarity degree.

For temporal cases, we refer to the additional temporal similarity measurement as the *temporal path strength*. For each such path the matching degree of corresponding findings is calculated (examples are given in section 7. Comparing the temporal paths is not trivial because there are often many possible temporal paths for a combination of two primitive relations (see table 2). The search space is reduced by a goal-directed search, guided by what to predict. What we in particular are interested in for our prediction task is to foresee possible warning states as early as possible. We developed an algorithm referred to as *dynamic ordering* [12] that enables intervals to be related to each other so that a similarity assessment of parameters can be made in order to predict particular states. The dynamic ordering algorithm uses two procedures in order to follow the temporal paths *{getNextInterval}* and *{getSameTimeIntervals}*. The dynamic ordering algorithm is as follows, where IC is the input case and CC the current case:

```
(1) Find first interval in IC and CC (intervalIC and intervalCC)
(2) Check intervalIC and intervalCC for matching or explainable findings
(3) If match - Update temporal path strength
(4) Check {getSameTimeIntervals} for new information and special situations
        If special situations - Perform action
(5) {getNextInterval} from CC and IC
(6) Unless {getNextInterval} is empty - Go to (2)
(7) Return temporal path strength
```

(1) The system finds the starting point. Figure 4 illustrates the temporal relationships related to past and future from the viewpoint of one interval. The first interval is found by selecting the interval that ends most to the left on the scale on Figure 4, and has no temporal relationships to intervals more to the left of the scale. The primary preference is a <*Before*> interval without any intervals connected to it with <*Before*>, <*Meets*>, <*Contains*>, <*Overlaps*> or <*Finished-by*> relationships. This interval is guaranteed to be the first one in time, since no other intervals can have an earlier starting point.

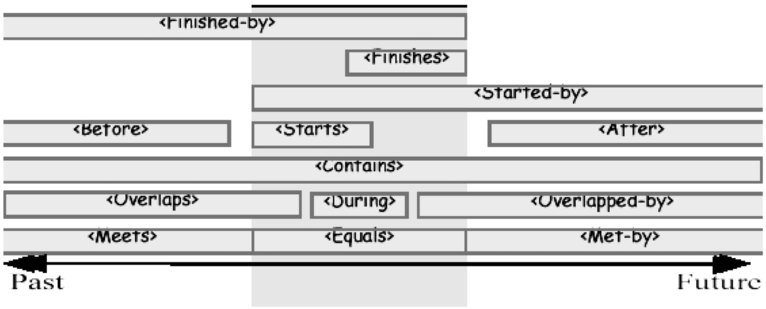


Figure 4 The thirteen possible relationships between intervals placed on a timeline. The marked square in the middle is the interval connected to, and the two white areas are intervals connected from. The relation names inside the bars show the relationship from an interval to the interval in the middle square.

(2) After recognition of the first intervals within IC and CC, the two intervals are checked for matched findings. The activation strength, the explanation strength and the matching strength are computed as they are in non-temporal cases. If the matching shows that the intervals are not similar, the second interval in CC is compared with the first in the input case. The process of finding the first similar interval may continue all through the CC-case, or a limit can be set for how far it is relevant to try, depending on the type of domain and on the duration of cases. (3) If a match is found, the matching strength, representing the path strength of this temporal path, is added to

the temporal path strength. (4) *{getSameTimeIntervals}* finds all the intervals in CC that are at the 'same time' as intervalIC. That is, all findings in intervalCC which share some time-points with intervalIC. This step is special for temporal cases. It keeps the current time-perspective of the input case while matching corresponding findings. It looks for the intervals of CC that are fully contained in or cover any of the temporal relationships inside the middle square of Figure 4 (<*Contains*>, <*During*>, <*Equals*>, <*Starts*>, <*Started-by*>, <*Finishes*> and <*Finished-by*>). (5) The procedure *{get-NextInterval}* selects the interval that is closest to the current in future time, using the scale in Figure 4. (6) The dynamic ordering continues until *{getNextInterval}* returns an empty value and the case is completely compared with all its intervals. When the input case is fully compared, the accumulated temporal path strength is normalized.

7 An Example of Prediction in Oil-Well Drilling

The exemplified domain is Stuck Pipe, and the cases are reconstructed on the basis of logged data from a Statoil well in the North Sea [25]. The case retrieval step incorporates the prediction goal of being warned about a possible drill-sticking situation sufficiently early to be able to prevent it.

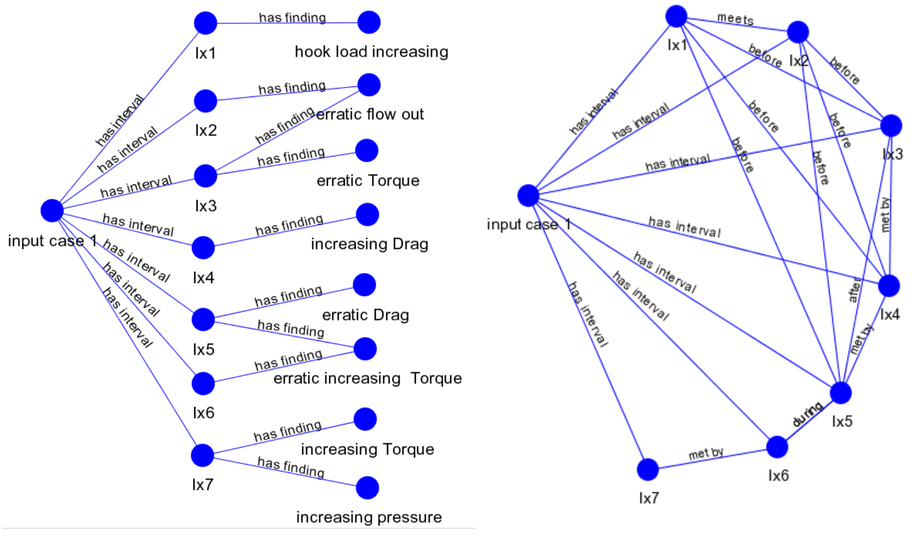


Figure 5 The input case with findings (left) and temporal relations (right), split in two for readability.

Figure 5 shows the input case after the raw data have been transformed into qualitative findings (currently a manual process). The intervals are named IX1, IX2, IX3, IX4, IX5, IX6 and IX7 (the numbers are just labels and do not necessarily correspond to temporal sequence). The relationship, IX1 *has-finding* HOOK LOAD INCREASING describes that the finding when hook load is increasing is in the time of interval Ix1, and IX2 *has-finding* ERRATIC FLOW OUT describes that the finding when flow out is erratic is in the time of interval Ix2. The relationship IX1 *meets* IX2 describes that the end of IX1 is the time-point before IX2. The entities of the two intervals can hence be temporally described as; HOOK LOAD INCREASING *meets* ERRATIC FLOW OUT.

The system starts by retrieving similar cases from the case-base in a non-temporal way. It retrieves two cases that have matching strength above a set threshold, from which the case "SPCase#1" has the highest strength. The two retrieved cases are then compared in order to find the temporal path strength, by means of dynamic ordering. It turns out that SPCase#1 also has the strongest temporal path strength. We will now trace the temporal matching of findings in the two cases. For simplicity, the activation strength for direct value matching is set to 1.0, while explained matches are set to 0.5.

The process starts by comparing the first intervals in time, and finds the matching finding HOOK LOAD INCREASING. The first interval of SPCase#1 also has the findings ERRATIC INCREASING TORQUE and DECREASING PRESSURE (see Figure 6, note that the inverse of the "has finding" relation is "reminding"), but they are neither directly matched nor explained by input case 1. The temporal path strength is now updated with the value of a direct match for one finding, i.e. it is increased with 1.0.

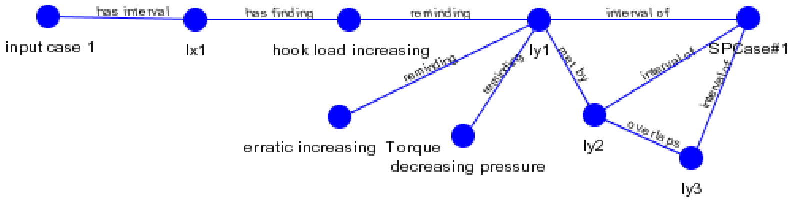


Figure 6 Comparing the first intervals of the input case and retrieved case.

The next step is to compare the set of intervals retrieved from the procedure *{getSameTimeIntervals}* in order to improve the match and look for occurrences of situations related to the prediction goal. From Figure 5 we see that Ix1 only contains <*before*> and <*meets*> relationships. Hence there are no intervals that share time with Ix1, and no additional intervals are retrieved.

The next intervals in the two cases are then found (procedure *{getNextInterval}*). The next interval to Ix1 is Ix2, determined by the *meets* relationship. One matching finding, ERRATIC FLOW OUT, is found in the second interval of the retrieved case (Iy2). The temporal path strength is updated with the value of one matching finding, and becomes 2.0. Ix2 and Iy2 are then compared in *{getSameTimeIntervals}*. Iy3 and Iy2 in SPCase#1 is found to share a time span, due to their <*overlaps*> relationship. All the findings in interval 2 and 3 are then compared, but no similarities are found. Interval Ix3 and Iy3 are then found to be similar, as they both have the findings ERRATIC FLOW OUT and ERRATIC TORQUE, see Figure 7. The combined value of two matching findings is used to update the path strength, which becomes 4.0. Iy4 has the finding HOOK LOAD INCREASING and Ix4 has the finding INCREASING DRAG. They are explained similar by the relationship INCREASING DRAG *causes* HOOK LOAD INCREASING. The temporal path strength is increased with the value of the explanation, and becomes 4.5.

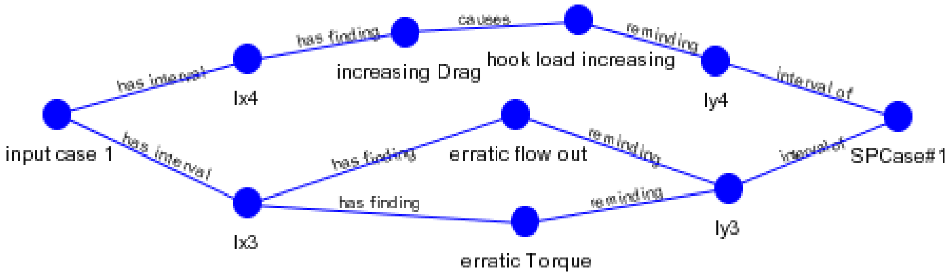


Figure 7 Direct and explained matches of Ix3/Iy3 and Ix4/Iy4.

Ix5 and Iy5 has one matching finding ERRATIC INCREASING TORQUE and one explained similar finding, INCREASING DRAG *causes* INCREASING HOOK LOAD. The temporal path strength is updated with 1.5, and becomes 6.0.

When *{getSameTimeIntervals}* is used on Ix5 and Iy5, and these are checked for goal-related situations, an alert state is found. Iy7 (linked by a <*during*> relation from Iy5) has a warning related to it, indicating that at this stage of the past case there was a possibility of an upcoming stuck pipe situation. The following intervals are found with similar time points: Iy5 *during* Iy7, Iy5 *overlaps* Iy6, Ix6 *during* Ix5 (see figure 8).

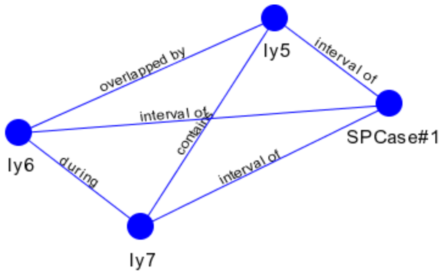


Figure 8 Temporal relations between I5, I6, and I7 in the retrieved case.

The set of intervals from input case 1 (Ix5 and Ix6) and from SPCase#1 (Iy5, Iy6 and Iy7) are then compared. The following findings are found similar in the two cases: Ix5 and Iy5 has one explained finding; ERRATIC DRAG *causes* HOOK LOAD ERRATIC, and one direct matched finding; ERRATIC INCREASING TORQUE.

Ix5 and Iy7 has one explained finding; ERRATIC DRAG *causes* HOOK LOAD ERRATIC, and one direct matched finding; ERRATIC INCREASING TORQUE.

Iy5 and Ix6 have a direct matched finding; ERRATIC INCREASING TORQUE, and Ix6 and Iy7 the direct match ERRATIC INCREASING TORQUE.

From the combination of the similar findings, Creek has computed a temporal path strength that indicates that the cases are similar at this stage and decides that the warning in Iy7 should be given to the user. Creek warns the user about a possible upcoming stuck pipe situation. Note that by using *{getSameTimeIntervals}* the system is able to find the alert state before Ix7 is directly compared with Iy7.

Relationships between SPCase#1 and input case 1 are visualized in Figure 9. We see that Iy7 and Ix7 have the two direct matching findings INCREASING TORQUE and INCREASING PRESSURE, which further increase the temporal path strength by 2.0. It finally becomes 8.0. The path strength is now divided on all the features of SPCase#1 that have been compared with the input case (22 features are counted). It gives the value 0.36 of the temporal path strength of SPCase#1.

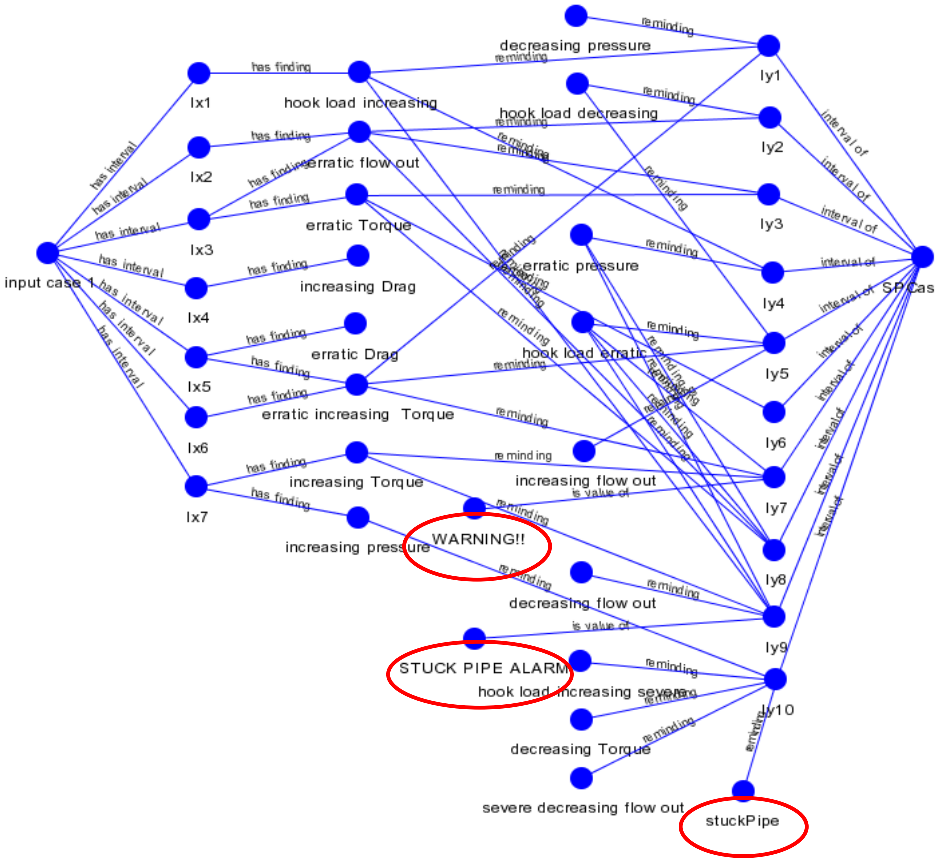


Figure 9 SPCase#1 and input case 1 compared with each other

8 Conclusion

We have presented a method for representation of temporal intervals in a CBR system, and how the representation is utilized in case retrieval. The method is aimed at supporting prediction of events for industrial processes. The paper has shown how Allen's temporal intervals can be incorporated into the semantic network representation of the Creek system. An algorithm for matching a sequence of findings that occur during an oil well drilling process was described, and exemplified. The research presented indicates that the interval-based approach is feasible for qualitative temporal reasoning in this context. An advantage of using intervals is that the representation becomes close to the way the human expert reasons in domains where qualitative changes of parameters over time are important. Another advantage is that it facilitates integration into a model-based reasoning system component, as shown in our example. However, as shown by the two-step retrieval method of Creek, the interval-based approach will also work with pure syntax-based retrieval methods (i.e. the activate step only).

A weakness of the representation is that it only enables one fixed layer of intervals in the cases. A more flexible way of handling intervals, in terms of intervals with sub-intervals, etc., would be interesting for some applications. We have done initial research into enabling dynamic structuring of Creek cases [15] that may be useful for that. So far, the system has not undergone a systematic evaluation. Even if Allen's algorithm in principle has nice complexity properties, its integration into the Creek method needs to be more thoroughly investigated. The system, implemented in Java, has only been tried on small knowledge bases.

To run the system in a real operational setting, the problem of transforming raw data into the qualitative changes used by our method needs to be solved. A strong motivation for our method is that many processes are too complex to be predicted by standard trend analysis. Temporal interval-based CBR methods, as exemplified in this paper, should have a high potential for revealing such complex problems.

References

1. Agnar Aamodt, Enric Plaza: "Case-based reasoning: Foundation issues, methodological variation, and system approaches", AICOM, Vol. 7, no. 1 (Mar. 1994) 39-59.2.
2. Agnar Aamodt: Explanation – driven-case-based reasoning. Topics in case-based reasoning, edited by S. Wess et al., Springer Verlag (1994) 274-288.
3. Agnar Aamodt, Helge A. Sandtorv, Ole M. Winnem: Combining Case Based Reasoning and Data Mining - A way of revealing and reusing RAMS experience. In Lydersen, Hansen, Sandtorv (eds.), Safety and Reliability; Proceedings of ESREL '98, Trondheim, June 16-19, 1998. Balkena, Rotterdam, 1998. pp 1345-1351.
4. James F. Allen: Maintaining Knowledge about Temporal Intervals. Communications of the ACM 26, 11, 832-843. The University of Rochester. Nov. 1983.
5. Ralph Bergmann, Hector Muñoz-Avila, Manuela Veloso and Erica Melis: CBR applied to planning. In Case-Based Reasoning Technology; From Foundations to Applications, (chapter 7), pages 169--200. 1998.
6. Branting, L.K., Hastings, J.D.: An empirical evaluation of model-based case matching and adaptation. *Proceedings of the Workshop on Case-Based Reasoning, AAAI-94.* Seattle, Washington, July 1994.
7. Christian Freksa: Temporal reasoning based on semi-intervals. In Artificial Intelligence 54 (1992) pp 199-227. 1992.
8. Bjarne K. Hansen: Weather Prediction Using Case-Based Reasoning and Fuzzy Set Theory, Master of Computer Science Thesis, Technical University of Nova Scotia, Halifax, Nova Scotia, Canada. 2000
9. Inrock Drilling Fluids: IDF Training Manual, Chapter Six - Drilling Problems, Rev2 1989.
10. Michel Jaczynski: A Framework for the Management of Past Experiences with Time-Extended Situations. In ACM Conference of Information and Knowledge Management (CIKM'97), Las Vegas, November 1997. INRIA Sophia-Antipolis,

France. 1997.

11. Jardine, McCann and Barber: An Advanced system for the early detection of Sticking pipe. Sedco Forex Schlumberger ADC/SPE 23915. 1992.
12. Martha Dørum Jære: Time sequences in case-based reasoning. Master of Computer Science Thesis, Norwegian University of Science and Technology, Trondheim, Norway, 2001.
13. K. M. Kahn and A. G. Gorry: Mechanizing temporal knowledge. Artificial Intelligence 9,2 (1977). pp 87-108.
14. E.T. Keravnou: Modeling medical concepts as time-objects. In M. Stefanelli and J. Watts, eds., Lecture Notes in Artificial Intelligence, vol. 934. Springer, 1995, pp. 67-90.
15. Ellen Lippe: Learning support by reasoning with structured cases. M.Sc. Thesis, Norwegian University of Science and Technology, Trondheim, Norway, 2001.
16. Bruce M. McLaren and Kevin Ashley: Helping a CBR program know what it knows. In David W. Aha and Ian Watson, eds., Lecture Notes in Artificial Intelligence, vol. 2080. Springer, 2001, pp. 377-391.)
17. Joaquim Melendez, Josep Lluis de la Rosa, Daniel Macaya, Joan Colomer: Case Based Approach for generation of recipes in Batch Process Control. Institut d'Infor-matica i Aplicacions -Universitat de Girona, Spain. 1999.
18. Joaquim Melendez, Daniel Macaya, Joan Colomer: Case Based Reasoning Methodology for Supervision. eXiT Group IliA & LEA-SICA, Univeristat de Girona, Spain. 2000.
19. S. Rougegrez. Similarity evaluation between observed behaviors for the prediction of processes. In Topics in Case-Based Reasoning, vol 873 of Lecture Notes in Artificial Intelligence, pp 155-166. Springer 1994.
20. Katarina Sachse: Domain model of drilling fluid problems. (Diploma thesis) Department of Petroleum Engineering, Norwegian University of Science and Technology. Dec. 2000.
21. Rainer Schmidt, Bernard Pollwein, Lothar Gierl: Medical multiparametric time course prognoses applied to kidney function assessments. In International Journal of Medical Informatics 53 (1999), pp 253-263.
22. Yuval Shahar: Timing is everything: Temporal Reasoning and Temporal Data Mainte-nance Medicine. In W Horn et al. (Eds.): AIMDM'99, LNAI 1620, pp30-46, 1999. Springer Verlag.
23. Pål Skalle, Agnar Aamodt and Jostein Sveen: Case-based reasoning - a method for gaining experience and giving advise on how to avoid and how to free stuck drill strings. Proceedings of IADC Middle East Drilling Conference, Dubai, Nov. 1998.
24. Paal Skalle, Josein Sveen and Agnar Aamodt: Improved Efficiency of Oil Well drill-ing through Case Based Reasoning. Proceedings of PRICAI 2000, The Sixth Pacific Rim International Conference on Artificial Intelligence, Melbourne August-September 2000. Lecture Notes in Artificial Intelligence, Springer Verlag, 2000. pp 713-723.
25. Statoil, Boreavdeling, Stavanger, Mud logging data, Gullfax, 1994.
26. Luis Vila: A survey of temporal reasoning in artificial intelligence. AI Communications, Volume 7, Number 1, March 1994, pp 4-28.