Discovering Temporal/Causal Rules: A Comparison of Methods

Kamran Karimi and Howard J. Hamilton

Department of Computer Science
University of Regina
Regina, Saskatchewan
Canada S4S 0A2
{karimi, hamilton}@cs.uregina.ca

Abstract. We describe TimeSleuth, a hybrid tool based on the C4.5 classification software, which is intended for the discovery of temporal/causal rules. Temporally ordered data are gathered from observable attributes of a system, and used to discover relations among the attributes. In general, such rules could be atemporal or temporal. We evaluate TimeSleuth using synthetic data sets with well-known causal relations as well as real weather data. We show that by performing appropriate preprocessing and postprocessing operations, TimeSleuth extends C4.5's domain of applicability to the unsupervised discovery of temporal relations among ordered data. We compare the results obtained from TimeSleuth to those of TETRAD and CaMML, and show that TimeSleuth performs better than the other systems.

1. Introduction

We consider the problem of discovering relations among a set of attributes that represent the state of a single system as time progresses. Given a sequence of temporally ordered records, with or without an explicit time attribute, the goal is to identify as many cases as possible where two or more attributes' values depend on each other. We may want to describe the system, predict future behavior, or control some of the attributes by changing the values of other attributes. For example, one observation may be that "(y = 5) is always true when (x = 2)." From this statement, we could predict the value of y as 5 when we see that x is 2. This is an example of *co-occurrence* between two values. Alternatively, the same statement could be interpreted as the rule: **if** {(x = 2)} **then** (y = 5), and use forward chaining to predict that setting the value of x to 2 will result in y becoming 5. This rule is thus interpreted as a *causal relation*.

Mining information from temporal data is an active research subject with different approaches. A *time series* is a time-ordered sequence of observations taken over time [1,2]. In a *univariate time series*, each observation consists of a value for a single attribute, while in a *multivariate time series*, each observation consists of values for several attributes. Most research on time series has assumed the presence of a distinguished attribute representing time and numeric values for all other attributes.

Attempts have been made to fit constant or time-varying mathematical functions to time series data [1].

An *event sequence* is a series of temporally ordered events, with either an ordinal time attribute (which gives the order but not a real-valued time) or no time attribute. Each *event* specifies the values for a set of attributes. A recurring pattern in an event sequence is called a *frequent episode* [18]. Recent research has emphasized finding frequent episodes with varying number of events between the events that identify the event sequence. Algorithms such as Dynamic Time Warping and its variants measure the similarity of patterns that are stretched differently over time [15]. These methods have not been applied to searching for relations in causal data.

Two tools that were designed for performing unsupervised search for causal relations are CaMML [14, 23] and TETRAD [21]. They look for relationships among all attributes, resulting in a non-linear increase in running time as the number of attributes increase. Although these tools were not designed for the exact problem of finding atemporal and temporal rules from temporally ordered data from a single source, they can be applied to this problem and provide the most appropriate existing techniques for comparison purposes.

CaMML is a Minimum Message Length based causal discovery system that uses a Bayesian minimum encoding technique. Given a set of observed attributes, CaMML finds causal relationships between one or more causes and a single effect. As an example, in CaMML's notation (A, $B \rightarrow C$) means that A and B are causes of C. The output of CaMML must be interpreted by a human expert.

TETRAD is a well-known causality discoverer that uses Bayesian networks [4] to find causal relations. It has its own notation for displaying the discovered causal rules. For example, $A \rightarrow B$ means that A causes B and $A \leftrightarrow B$ means that both A and B have a hidden common cause. Unfortunately TETRAD's results are not always precise. $A \bullet\!\!\rightarrow B$ means that either A causes B, or they both have a hidden common cause, $A \bullet\!\!-\!\!\bullet B$ means that A causes B or B causes A, or they both have a hidden common cause (usually considered to mean the same thing as co-occurrence). This ambiguity opens the door to different interpretations of the results. In this paper we attempt to sidestep this ambiguity by using an artificial data set with known relations. TETRAD allows the user to provide it with information regarding the temporal order of the attributes. We exploited this feature in our experiments to give TETRAD all available information.

Interpreting co-occurrence relations as causal relations, as is done by software such as TETRAD, requires justification. The main trend in causality mining involves using the statistical concept of conditional independence as a measure of the control one attribute may have over another [19]. For example, given the three attributes x, y, and z, if x is independent from y given z, that is, $P(x, y \mid z) = P(x \mid z)$, then we can conclude that y is not a direct cause of x. In other words, z separates x and y from each other. This basic concept is used to build Bayesian Networks, which show the conditional dependence of attributes as arcs. These arcs are then *interpreted* as signifying causal relations.

The notion of conditional independence is void of time. The proponents of this popular method use temporal information, if it is available, to place constraints on the relationships among the attributes (if we know x always happens before y, then y cannot

be a cause of *x*), but time is not essential to the working of their algorithms. Leaving out time when dealing with the notion of causality seemed counterintuitive to us. The result was an investigation into the use of a standard tool, C4.5 [19], to discover relations given temporally ordered input data. We chose C4.5 because it is available in source code form, and has been widely used.

In this paper, we describe TimeSleuth [12], an enhanced version of our RFCT software [9], for the problem of finding atemporal and temporal relations in causal data and compare it to two other causality miners. By performing appropriate preprocessing (rotation and flattening) and postprocessing (applying temporal constraints), TimeSleuth extends C4.5's domain of applicability to the unsupervised discovery of temporal relations among ordered nominal or numeric data. It searches for relations among attributes that characterize the behavior of a system. In general, such relations can be causal or acausal, but we consider data sets where causal relations exist, and evaluate the effectiveness of methods at finding these relations. We distinguish between *atemporal relations*, which involve values observed at the same time, and *temporal relations*, which involve values from different time steps. TimeSleuth further divides temporal relations into causal and acausal. TimeSleuth implements the TIMERS method (Temporal Investigation Method for Enregistered Record Sequences), which is formally introduced in [13]. The same reference describes how the method distinguishes between causal and acausal relations.

The remainder of this paper is organized as follows. Section 2 introduces the TimeSleuth method and software. Section 3 describes experiments with synthetic data from a simple problem domain that contains atemporal and temporal/causal relations. The simplicity of the domain allows us to judge the results with little ambiguity. Section 4 presents experiments on real-world data. Section 5 concludes the paper.

2. TimeSleuth

TimeSleuth is a tool for finding and analyzing atemporal and temporal predictive relations in data. It includes a variant of C4.5 to form decision rules. TimeSleuth provides a user interface and processes data before and after running C4.5. Many other rule discoverers could be used instead of C4.5. TimeSleuth assumes the existence of a temporal order among the input records, and based on that assumption, formulates predictive rules among the attributes. These rules are *temporally consistent*, i.e., condition attributes are referred to at or before the time of the decision attribute. TimeSleuth's mathematical model and method are explained in [11].

TimeSleuth requires a patched version of C4.5, which we call C4.5T (**T**emporal), which adds time indications to output rules. Patch files with all the necessary modifications are included with the TimeSleuth package. C4.5T's output can include both temporal decision trees and temporal rules [10]. The relations found by TimeSleuth may or may not be causal. For a discussion on causality, see [3, 5, 16, 22]. TimeSleuth finds relations that have a temporal relationship between a cause and an effect. It provides a metric to distinguish among causal and acausal relations.

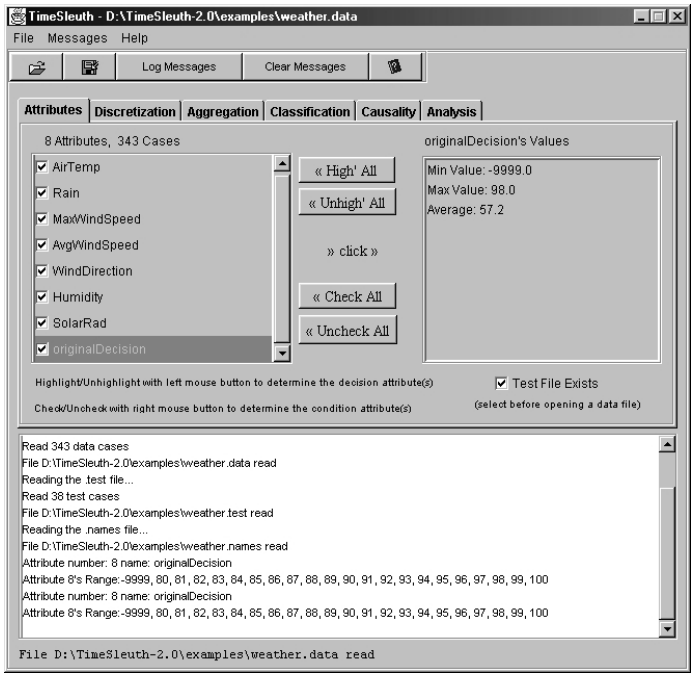

Fig. 1. Screen shot of TimeSleuth

TimeSleuth's graphical user interface is designed to encourage the user to experiment with the data and different scenarios. Unlike C4.5, which requires the user to select a single attribute as the decision attribute, TimeSleuth can iterate over many attributes and invoke C4.5T with a new decision attribute each time. It rotates the fields in the observation data, so that the value of the decision attribute appears last, as required by C4.5. The rules resulting from running C4.5T are sorted by it so that the attributes appear in correct temporal order, and are presented by TimeSleuth in tables that emphasize the temporal characteristics of the rules. The user can thus easily see any temporal relationship among the attributes, and how each attribute is used as time progresses. The user can also opt to have the rules output in the form of Prolog statements [7, 8], thus making the results readily executable. TimeSleuth is thus both a machine learning tool (which outputs machine-usable rules) and a data mining tool (which helps the user to discover temporal relations among data through experimentation). C4.5 assigns a confidence value to the rules it generates. The user can instruct TimeSleuth to prune the output rules so that only rules with a confidence value greater than a threshold are presented. Figure 1 shows TimeSleuth's user interface after loading a file that contains hourly weather observations. The decision attribute (here corresponding to Soil Temperature) is selected by being highlighted. The condition attributes to be used during

rule generation have a check mark next to them. TimeSleuth can discretize real-valued attributes as needed.

TimeSleuth merges consecutive records into one record, which brings previous values together with later observations, helping in the discovery of relationships between temporally distant values. This process is called flattening [6, 13], and the number of records merged is determined by the window size (w). The assumption with bigger window values is that the events' effects have a longer delay. TimeSleuth's Analysis panel presents the output in different ways to help the user in discovering such properties. As an example, Figure 2 shows how often each attribute at each time step has been mentioned in the temporal rules. In this figure, TimeSleuth has generated rules with window values from 1 to 14, and the current window size is set to 5. The user can employ the slider to see the effects of selecting other window size values. In Figure 2 the decision attribute is soil temperature.

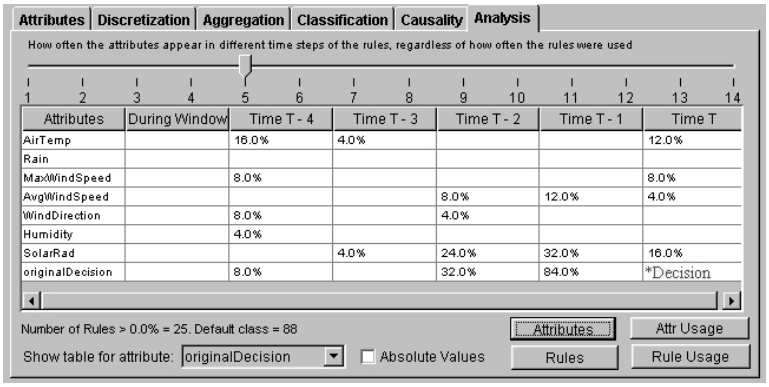

Fig. 2. Attribute usage in the weather data with window size of 5.

Finding the appropriate window size for mining data may be a challenge for the domain expert. For this reason, TimeSleuth can be run in batch mode, where different window sizes are tried automatically. The user can choose criteria, such as the error rate, to stop the search for an appropriate window size. When applied to sample weather data, this method found the best results with a window size of 3 (hours), as shown in Figure 4. These results are better than standard C4.5's results (w = 1), indicating an advantage of the temporal investigation of the data. TimeSleuth can also be used to distinguish between causal and acausal relations. It considers time to have two directions. The forward direction, which is the natural direction of time, is used to test the data for causality. Here the past observations are used to predict the value of the decision attribute. The backward direction of time is used to test the acausality of the data. In this case the future observations are used to predict the value of the decision attribute. In an acausal relation the value of the decision attribute is in a temporal "co-occurrence" relation with the condition attributes. The results determine the quality of the rules, and thus the verdict of the system. This verdict could be one of causal, acausal, or instantaneous (when window

size 1 gives the best results). Causal and acausal relations are both considered temporal, and in the rest of this paper we will emphasize the temporal aspects of the rules discovered by TimeSleuth. See [13] for a treatment of the distinction between causality and acausality in TimeSleuth.

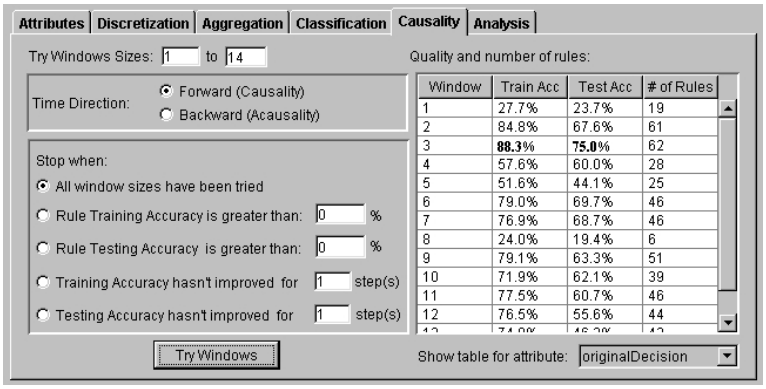

Fig. 3. The quality of the generated rules for different window sizes

Finally, TimeSleuth can use various aggregate values of the attributes in the rules. The sum, mean, median, mode, minimum (or logical-OR), and maximum (or logical-AND) of the values of an attribute at all time steps in the window can be computed and treated as new attributes. If a new attribute is identical to an existing attribute, it is discarded. If a new attribute is used as a condition in a rule, it will be described as "during time window" instead of "at time t." An example rule is: **if**{ *during time window*: *mean*(*a*) > 1 **and** *at time 1*: $x = 1$} **then** *at time 2*: $x = 2$.

3. Experiments on Synthetic Data

We used an Artificial Life [17] simulator called URAL [24] to generate data for the experiments described in this section. URAL is a discrete event simulator where known atemporal and temporal rules govern an artificial environment. Having complete knowledge about the URAL domain allows us to judge the quality of the discovered rules. Other kinds of data would have required interpretations as to the true nature of relations among the attributes, making the process of judging the output more complex and open to debate. The world in URAL is a rectangular, two-dimensional board with an agent living in it. Food exists at specific locations on the board. The agent can sense its position and also the presence of food at its current position. At each time-step, it randomly chooses one of the following actions: left (L), right (R), up (U), or down (D). Left and right correspond to moving along the *x*-axis and up and down to moving along the *y*-axis. The agent can sense which action it takes in each situation. If the agent attempts to move

beyond the boundary of the board, the move is ignored and the location of the agent is left unchanged.

Data from URAL form an event sequence (i.e., records are in temporal order) with attributes $\{x, y, f, a\}$, where x and y give the agent's location coordinates, f indicates whether food was present at this location, and a tells which action was taken at the location. An example record is <2, 3, true, L>. The agent, moving around randomly, can visit the same location more than once.

A single record does not contain the effect of the move action. To bring the possible cause (action) and the possible effect (next position) together in the same record, flattening is required to merge consecutive records into one. A minimum window size of 2 is required to allow a method to find the correct temporal/causal rules for the URAL domain because the complete effect of a move is known in the following time step. An example flattened record with a window size of 2 is <2, 3, true, D, 2, 4, false, L>.

The purpose of the experiments in this section is to measure the effectiveness of TimeSleuth, TETRAD, and CaMML in discovering rules when provided with temporal data. First we apply the methods to unflattened data, which should allow them to discover atemporal rules. Then we apply the methods to data flattened with various window sizes, which should allow them to discover atemporal and temporal rules. The results show how flattening affects the output of the methods.

We applied the three methods to example data from the URAL domain. TETRAD restricts attributes to at most 8 values, so the size of the world was set to 8×8, with x and y ranging from 0 to 7. To ensure fairness to all methods, all results presented in this section are derived from a single run of 10,000 time steps in URAL where food existed at locations (0,6), (1,2), (3,5), (5,5), and (6,3). These results are representative of many experiments we have performed with varying number of time steps and a variety of locations for food. The confidence value of the correct rules generally increased with the number of records.

Window size	Domain Expert	TimeSleuth	TETRAD	CaMML
1	$f = FoodXY(x, y)$	**if** $\{(x = \alpha)$ **and** $(y = \beta)\}$ **then** $(f = \delta)$.	$x \bullet\!\!-\!\!\bullet f$, $y \bullet\!\!-\!\!\bullet f$	$(x, y \rightarrow f)$
	No relation for a, x, y	No rules	a, x, y	$(\rightarrow a)$, $(\rightarrow x)$, $(\rightarrow y)$
$w \geq 2$	$f_w = FoodXY(x_w, y_w)$	**if** $\{(x_w = \alpha)$ **and** $(y_w = \beta)\}$ **then** $(f_w = \delta)$.	$x \bullet\!\!-\!\!\bullet f_w$, $y \bullet\!\!-\!\!\bullet f_w$	$(x_w, y_w \rightarrow f_w)$
	$x_w = MoveX(x_{w-1}, a_{w-1})$	**if** $\{(x_{w-1} = \alpha)$ **and** $(a_{w-1} = \gamma)\}$ **then** $(x_w = \delta)$.	$x_{w-1} \rightarrow x_w$, $a_{w-1} \rightarrow x_w$	$(x_{w-1}, a_{w-1} \rightarrow x_w)$
	$y_w = MoveY(y_{w-1}, a_{w-1})$	**if** $\{(y_{w-1} = \beta)$ **and** $(a_{w-1} = \gamma)\}$ **then** $(y_w = \delta)$.	$y_{w-1} \rightarrow y_w$, $a_{w-1} \rightarrow y_w$	$(y_{w-1}, a_{w-1} \rightarrow y_w)$
	No cause for a_w	No rules	a_w	$(\rightarrow a_w)$

Table 1. The correct output for TimeSleuth, TETRAD, and CaMML

Table 1 shows the desired output of TimeSleuth, TETRAD, and CaMML in each tool's notation. The domain expert's beliefs about the relations actually present in the simulator are also given. Composite rules, such as "x_{w-2}, a_{w-2}, and a_{w-1} give x_w" are not included because of our preference for shorter rules. *FoodXY*() is an atemporal relation saying whether or not food is present at a given (x,y) location, while *MoveX*() and *MoveY*() are temporal relations describing the effects of moving along the x-axis and y-axis, respectively. For any window size $w \geq 2$, the result of a move depends only on the

position and the move direction in the *w*-1 time step (time starts at 1 and ends at *w* in each flattened record). In Table 1, TETRAD's and CaMML's desired output is represented directly, but TimeSleuth's desired output is given in template form. The actual output has multiple rules, with one rule for each combination of the values for the attributes representing conditional attributes. Also, the actual output does not contain keywords such as *if*, *then*, or *and*, and has specific values instead of the α, β, γ and δ parameters. We expect the atemporal relations that hold for $w = 1$ to also hold for $w > 1$ because atemporal relations do not depend on time.

Window size 1: To test their ability to find atemporal relations, we ran TETRAD, CaMML and TimeSleuth. The results for TETRAD, shown in Table 2, were generated by the "Build" command, assuming the existence of latent common causes, and using the exact algorithm. The best results were found at the significance levels from 0.001 to 0.01, where rules *a*, $x \bullet\!\!-\!\!\bullet f$ and $y \bullet\!\!-\!\!\bullet f$ are correct, because the action has no discernible cause and food is associated with both *x* and *y*. Other rules are incorrect in the context of a single record, because no other causal or atemporal rules exist among the attributes in a single record.

Significance Levels	Correct Rules	Incorrect rule
0.001, 0.005, 0.01	$f \bullet\!\!-\!\!\bullet x,\ y \bullet\!\!-\!\!\bullet f, a$	$x \bullet\!\!-\!\!\bullet y$
0.05, 0.1		$y \leftrightarrow f,\ y \leftrightarrow x, x \rightarrow f, a \bullet\!\!\rightarrow x$
0.2	$f \bullet\!\!-\!\!\bullet x,\ y \bullet\!\!-\!\!\bullet f, a$	$y \bullet\!\!-\!\!\bullet a, y \bullet\!\!-\!\!\bullet x, a \bullet\!\!-\!\!\bullet x$

Table 2. Atemporal rules discovered by TETRAD ($w = 1$).

For the same data, CaMML gave the results shown in Table 3. CaMML found all the correct atemporal rules. In fact, *x* and *y* do not cause *f* in URAL, but without any domain knowledge, it is reasonable to interpret this relationship as a causal one.

Correct Rules	Incorrect rules
$(\rightarrow a), (\rightarrow x), (\rightarrow y), (x, y \rightarrow f)$	

Table 3. CaMML's results with unflattened records ($w = 1$)

TimeSleuth gave the results shown in Table 4. The first two entries for predicting the value of *f* show that the C4.5rules component of TimeSleuth eliminated unneeded condition attributes when creating rules. In general, if the value of *x* is sufficient to predict the outcome regardless of the value *y*, the generated rule will not include *y* and vice versa. We accept all the rules generated for predicting *f* because together they correctly predict the presence or absence of food.

TimeSleuth's desired output for other attributes would be no rules, which can be achieved by not setting *x*, *y* or *a* as the decision attribute. For the sake of completeness we did try TimeSleuth with these attributes, with the results shown in Table 4. As a classifier, C4.5 creates rules regardless of whether or not they make semantic sense. However, the results were interesting. There is a strong co-occurrence between *f*, *x*, and *y*, and this is exploited by TimeSleuth for predicting the value of these attributes with high confidence

levels. The value of *a*, however, is unrelated to any other observable attribute, and this is reflected in low confidence levels for the rules. From an atemporal, co-occurrence point of view, the results in Table 4 are satisfactory.

Decision Attribute	Condition Attribute(s)	Number of Rules	Correctness	Min / Max Confidence	Training Accuracy
f	*x*	3	correct	99.9% / 99.9%	100%
	y	4	correct	99.9% /99.9%	
	x, y	20	correct	99% / 99.4%	
x	*y, f*	4	incorrect	52.9% / 99.4%	19.9%
y	*x, f*	5	incorrect	99% / 99.4%	21.2%
a	*x, y*	7	incorrect	30.3% / 33.9%	26.4%

Table 4. TimeSleuth's results with unflattened records ($w = 1$)

TimeSleuth can prune the rules based on the confidence level, and thus only the rules with a confidence level greater than a certain threshold will be displayed to the user.

Window size 2: Flattening using a window size of $w = 2$, produced records with 8 attributes $\{x_1, y_1, f_1, a_1, x_2, y_2, f_2, a_2\}$. TETRAD, CaMML, and TimeSleuth were applied to determine whether they could discover the *FoodXY*() atemporal function and the *MoveX*() and *MoveY*() temporal functions.

TETRAD's output is summarized in Table 5. TETRAD has a mechanism to let the user specify the temporal order among the input attributes, and this information was provided in the input. Flattening the records to give TETRAD more relevant data resulted in many more rules being discovered, most of which are incorrect. TETRAD was thus unable to exploit the extra information provided to it. The rule "$f_1 \; x_2$#" means that TETRAD considered the input information concerning the relation between to f_1 and x_2 to be contradictory (some observations supporting f_1 to be the cause of x_2, and some vice versa).

Significance Level(s)	Correct Rules	Incorrect Rules
0.001	a_2	$y_1 \leftrightarrow f_1$, $y_1 \bullet\!\!\rightarrow y_2$, $y_1 \leftrightarrow f_2$, $f_1 \leftrightarrow x_1$, $f_1 \leftrightarrow y_2$, $f_1 \leftrightarrow x_2$, $a_1 \leftrightarrow y_2$, $a_1 \leftrightarrow x_2$, $x_1 \leftrightarrow f_2$, $x_1 \bullet\!\!\rightarrow x_2$, $y_2 \leftrightarrow f_2$, $f_2 \leftrightarrow x_2$,
0.005, 0.01, 0.005	a_2	$y_1 \leftrightarrow f_1$, $y_1 \bullet\!\!\rightarrow y_2$, $y_1 \leftrightarrow f_2$, $f_1 \leftrightarrow x_1$, $f_1 \leftrightarrow y_2$, $f_1 \leftrightarrow x_2$, $a_1 \leftrightarrow y_2$, $a_1 \leftrightarrow x_2$, $x_1 y_2$#, $x_1 \leftrightarrow f_2$, $x_1 \rightarrow x_2$, $y_2 \leftrightarrow f_2$, $x_2 \rightarrow f_2$
0.1	$x_1 \rightarrow x_2$,	$f_1 \bullet\!\!\rightarrow y_1$, $y_1 \leftrightarrow x_1$, $y_1 \leftrightarrow y_2$, $y_1 \leftrightarrow f_2$, $y_1 \leftrightarrow x_2$, $f_1 x_2$#, $f_1 \leftrightarrow y_2$, $f_1 \leftrightarrow x_2$, $a_1 \leftrightarrow y_2$, $a_1 \bullet\!\!\rightarrow f_2$, $a_1 \leftrightarrow x_2$, $x_1 \rightarrow y_2$, $x_1 \leftrightarrow f_2$, $x_1 \leftrightarrow a_2$, $y_2 \leftrightarrow f_2$, $f_2 \leftrightarrow x_2$,
0.2		$f_1 \bullet\!\!\rightarrow y_1$, $y_1 \leftrightarrow x_1$, $y_1 \leftrightarrow y_2$, $y_1 \leftrightarrow f_2$, $y_1 \leftrightarrow x_2$, $f_1 \leftrightarrow x_1$, $f_1 \leftrightarrow y_2$, $f_1 \leftrightarrow x_2$, $a_1 \leftrightarrow y_2$, $a_1 \bullet\!\!\rightarrow f_2$, $a_1 \leftrightarrow x_2$, $x_1 \rightarrow y_2$, $x_1 \leftrightarrow f_2$, $x_1 \leftrightarrow a_2$, $y_2 \leftrightarrow f_2$, $y_2 \leftrightarrow a_2$, $f_2 \leftrightarrow x_2$,

Table 5. The rules discovered by TETRAD from the flattened records ($w = 2$)

The results of applying CaMML with a window size of 2 appear in Table 6. For CaMML, we were unable to discover any means of specifying a temporal order among the input attributes. It continued to discover the same relations as it had found with unflattened records, which is desirable, as the relations that existed in the previous case continue to exist in the flattened data. However, it failed to discover the relationship

between the previous location and action, and the current location. It also discovered the rule (x_1, y_1, x_2, $y_2 \rightarrow a_1$), which is a valid relationship among these attributes. However, it is temporally invalid, as it refers to a time in the future to predict the past.

Correct Rules	Incorrect rules
$(\rightarrow x_1)$, $(\rightarrow y_1)$, $(x_1, y_1 \rightarrow f_1)$, $(x_2, y_2 \rightarrow f_2)$, $(\rightarrow a_2)$	$(x_1, y_1, x_2, y_2 \rightarrow a_1)$, $(\rightarrow x_2)$, $(\rightarrow y_2)$

Table 6. CaMML's rules for the flattened records (w = 2)

The results of applying TimeSleuth to the same data are given in Table 7. In our 8×8 board, 4 possible actions and 8 distinct values exist for each of x_1 and y_1. In this example, the agent has explored the entire world, and TimeSleuth created 32 (8×4) rules for predicting the next value of each of x_2 and y_2. It correctly pruned y_1 from the rules for x_2, because the rules for moving along the x-axis are independent of the value of y. Similarly, it pruned x_1 from the rules for y_2. The rules for predicting f_2 are the same as the rules given in Table 4 for predicting f in the atemporal case, because the rules for food are not dependent on time. TimeSleuth was also tried on the a_2 attribute, which is not caused by any of the observable attributes. We did not use x_1, y_1, f_1, and a_1 as decision attributes. Consistent with the results with unflattened data, the rules have low confidence levels and can be pruned out completely.

Decision Attribute	Condition Attribute(s)	Number of Rules	Correctness	Min/Max confidence	Training Accuracy
f_2	x_1	3	correct	99.9% / 99.9%	100%
	y_1	4		99.9% / 99.9%	
	x_2, y_2	20		99% / 99.4%	
x_2	x_1, a_1	32	correct	99.7% / 99.4%	100%
y_2	y_1, a_1	32	correct	99.5% / 99.6%	100%
a_2	x_1, a_1, y_2	16	incorrect	30.2% / 45.7%	29.8%
	x_1, y_1, a_1,	18			
	x_1, y_1, y_2	6			
	x_1, x_2, y_2	7			
	x_1, y_1, x_2	4			
	x_1, y_2	3			
	x_1, y_1	2			
	x_1, a_1	1			
	a_1, y_2	1			

Table 7. TimeSleuth's results for the flattened records (w = 2)

Window Sizes Larger Than 2: TimeSleuth obtained good results with larger window sizes. As previously demonstrated [6], C4.5 effectively handles the bigger input records that are created by larger window sizes by pruning irrelevant attributes. TETRAD yielded very unsatisfactory results, shown in Table 8, for w = 3 with a significance level of 0.01, and these results are typical of the results for significance levels from 0.001 to 0.2.

Significance Level	Correct Rules	Incorrect Rules
0.01	$a_3, y_2 \rightarrow y_3$	$y_1 \leftrightarrow f_1, y_1 \leftrightarrow y_2, y_1 \leftrightarrow f_2, y_1 \leftrightarrow y_3, y_1 \leftrightarrow f_3, y_1 \leftrightarrow x_3, f_1 \leftrightarrow x_1, f_1 \leftrightarrow y_2, f_1 \leftrightarrow x_2, f_1 \leftrightarrow y_3, f_1 \leftrightarrow f_3, f_1 \leftrightarrow x_3, a_1 \leftrightarrow y_2, a_1 \leftrightarrow x_2, a_1 \leftrightarrow y_3, a_1 \leftrightarrow x_3, x_1 \leftrightarrow y_2, x_1 \leftrightarrow f_2, x_1 \leftrightarrow x_2, x_1 \leftrightarrow f_3, x_1 \leftrightarrow x_3, y_2 \leftrightarrow f_2, y_2 \leftrightarrow f_3, x_2 \rightarrow f_2, , f_2 \leftrightarrow y_3, , f_2 \leftrightarrow x_3, a_2 \leftrightarrow y_3, a_2 \leftrightarrow x_3, x_2 \rightarrow y_3, x_2 \rightarrow f_3, x_2 \bullet\!\!\rightarrow x_3, y_3 \leftrightarrow f_3, f_3 \leftrightarrow x_3$

Table 8. The rules discovered by TETRAD from flattened records ($w = 3$)

CaMML could not process the more complex data because the version we used could not handle the increased number of combinations. For $w > 3$, TETRAD's running time was too long (no results after days of computing). Table 9 shows the results obtained with window sizes from 3 to 10.

Window Size	Domain Expert	TimeSleuth	TETRAD	CaMML
$3 \le w \le 10$	*FoodXY*(x_w, y_w)	**if** $\{(x_w = \alpha)$ **and** $(y_w = \beta)\}$ **then** $(f_w = \delta)$.	Poor results with $w = 3$; running time too long for $w > 3$	Not tested because it cannot handle the input size
	MoveX(x_{w-1}, a_{w-1})	**if** $\{(x_{w-1} = \alpha)$ **and** $(a_{w-1} = \gamma)\}$ **then** $(x_w = \delta)$.		
	MoveY(y_{w-1}, a_{w-1})	**if** $\{(y_{w-1} = \beta)$ **and** $(a_{w-1} = \gamma)\}$ **then** $(x_w = \delta)$.		

Table 9. Test results for larger window sizes

The overall results of our experiments are summarized in Table 10, where the number of correct and incorrect relations/rules discovered by each system, for a variety of window sizes, is given. For TETRAD, we use the 0.05 significance level and for TimeSleuth we assumed a threshold confidence of 80%.

Window Size	TETRAD		CaMML		TimeSleuth	
	Correct	Incorrect	Correct	Incorrect	Correct	Incorrect
$W = 1$	3	1	1	3	15	1
$W = 2$	3	15	1	7	79	0
$W = 3$	3	39	N/A		79	0
$4 \le w \le 10$	N/A		N/A		79	0

Table 10. Summary of experimental results

4. Experimental Results on Weather Data

We also tested TimeSleuth on a weather data set from the Louisiana AgriClimatic Information System [25], which contains observations of 8 environmental attributes gathered hourly from 22/7/2001 to 6/8/2001. The first 343 observations were used for training, and the last 38 hours' data were used for testing the rules. The result of trying

TimeSleuth on all available attributes using window sizes from 1 to 10 hours appears in Table 11. "W" represents the window size, "Tr" the training accuracy, and "Ts" the testing (predictive) accuracy, all in percentages.

W	Soil Temp		Air Temp		Humidity		Wind Direc		AvgWindSp		MaxWindSp		Solar Rad	
	Tr	Ts	Tr	Ts	Tr	Ts	Tr	Ts	Tr	Ts	Tr	Ts	Ts	Tr
1	27.7	15.8	39.7	7.9	43.4	23.7	**4.4**	**100**	73.2	57.9	44.3	**44.7**	44.3	**42.1**
2	85.1	48.6	**48.8**	16.2	45.9	29.7	2	100	71.3	**67.6**	46.2	37.8	43.3	40.5
3	**88.6**	**61.1**	42.2	16.7	51	**33.3**	1.5	100	**72.4**	66.7	**46.9**	30.6	43.4	38.9
4	57.6	48.6	35.9	14.3	30	22.9	1.5	100	51.8	51.4	34.4	37.1	43.5	37.1
5	51.6	41.2	22.1	11.8	20.4	20.6	1.5	100	66.4	64.7	N/A	N/A	**44.5**	32.4
6	79	42.4	40.8	24.2	48.2	21.2	1.5	100	69.2	60.6	38.5	27.3	43.2	33.3
7	76.9	40.6	39.2	**28.1**	46.9	15.6	1.5	0	51.3	46.9	46.9	21.9	43	31.2
8	24.1	25.8	38.7	22.6	**53**	22.6	3.3	0	65.8	58.1	43.2	22.6	42.9	32.3
9	79.1	36.7	42.4	23.3	27.6	10	2.1	0	67.5	53.3	41.8	20	42.7	33.3
10	71.9	37.9	12.9	0	50.3	24.1	2.1	0	69.5	58.6	38.9	20.7	42.5	34.5

Table 11. Traing and testing accuracy for all attributes except rain (in percentages).

The training accuracy for Rain was 99.7% and the predictive accuracy was 100%. The reason for the high numbers is that there were few rainy days in the training data, and simply selecting the default value of "no rain" gave very good results. As evident in Table 11, Soil Temperature, Humidity, and Air Temperature have their highest accuracy for window sizes 3, 2, and 8, respectively. In these three cases, the results are better than for $w = 1$ (atemporal case). The user can interpret these results to mean that these attributes are temporally dependent on the previous values of the attributes, while the Wind Direction, Average Wind Speed, Max Wind Speed, and Solar Radiation attributes have little temporal dependence on previous observations.

These observations agree with common sense expectations. For example, we expect the soil temperature to be temporally related to the values of the available attributes at the current time and during the previous several hours. Other attributes such as wind direction and speed depend on factors that may not be available locally (difference of temperature between two regions). TimeSleuth's temporal rules for soil temperature will thus be more accurate than the atemporal rules that C4.5 could form from the unflattened data.

The available data allows a temporal investigation of the values of the Soil Temperature, Air Temperature, and Humidity attributes. The user can thus relate the current values to the previous values of the available attributes. For a window size of 3, a total of 62 rules were discovered. Table 12 summarizes the composition of the rules for predicting Soil Temperature. Each entry shows the percentage of rules in which a particular attribute at a particular time appears. Since the window size is 3, the current hour is time 3, the previous hour is time 2, and two hours ago is time 1. For example, the Air Temperature at time 2 appears in 14.5% of the rules for predicting the value of the Soil Temperature at time 3. The usage of attributes in rules with other window sizes is roughly consistent with Table 12, because most attributes that appear in rules are drawn from the current and previous two time steps. The general concentration of the attributes in the last 3 time steps can be seen also in Figure 2, which shows the attribute usage when the window size is 5.

Attribute	Time T-2	Time T-1	Time T
Air temperature	6.4%	14.5%	17.7%
Rain	0%	0%	0%
MaxWindSeed	1.6%	0%	9.6%
AvgWindSpeed	3.2%	4.8%	8.0%
WindDirection	8.0%	4.8%	6.4%
Humidity	3.2%	1.6%	3.2%
Solar Radiation	19.3%	9.6%	20.9%
Soil Temperature	19.3%	88.7%	N/A

Table 12. Attribute usage when predicting the value of Soil Temperature with a window size 3.

We tried TETRAD with the same weather data (unflattened). The results appear in Appendix 1. Because weather is very complicated and one can argue that "to some degree" everything is related to everything else, we have refrained from categorizing the output as either correct or incorrect.

5. Concluding Remarks

We introduced a new unsupervised learning tool called TimeSleuth, which is based on C4.5 and relies on straightforward methods to extend its abilities. C4.5 is a supervised learning tool that requires the user to identify one attribute as the decision attribute. By applying C4.5 with each decision attribute in turn to flattened input, TimeSleuth was able to discover atemporal and temporal rules, which can be interpreted as causal rules.

TimeSleuth is specifically meant for cases where data is temporally ordered and generated by a single source. The results in this paper indicate that it is effective in finding atemporal and temporal/causal rules when presented with such data. On synthetic data with strong causal relationships among the attributes, TimeSleuth found all correct temporal rules. On the same data, TETRAD and CaMML discovered many incorrect rules and also stopped working when the input size increased. On real weather data, TimeSleuth's results agreed with common sense knowledge of weather. However, despite TimeSleuth's better performance in this paper, it is not meant to replace software such as TETRAD, as its domain of applicability is more restricted (temporally ordered output of a single source versus data generated by many sources, regardless of their temporal order). As well, we used a preliminary version of CaMML, and its authors may now have a better version. We will next apply TimeSleuth to many years worth of weather data.

TimeSleuth is written in Java and runs in any environment that supports the Java runtime environment and has a graphical user interface such as Microsoft Windows or X Window. The package includes full source code and online help, and is available from http://www.cs.uregina.ca/~karimi/downloads.html.

References

1. Berndt, D. J. and Clifford, J., Finding Patterns in Time Series: A Dynamic Programming Approach, *Advances in Knowledge Discovery and Data Mining*. U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, et al. (eds.), AAAI Press/ MIT Press, 1996, pp. 229-248,
2. Chatfield, C., *The Analysis of Time Series: An Introduction*, Chapman and Hall, 1989.
3. Freedman, D. and Humphreys, P., *Are There Algorithms that Discover Causal Structure?*, Technical Report 514, Department of Statistics, University of California at Berkeley, 1998.
4. Heckerman, D., *A Bayesian Approach to Learning Causal Networks*, Microsoft Technical Report MSR-TR-95-04, Microsoft Corporation, May 1995.
5. Humphreys, P. and Freedman, D., The Grand Leap, *British Journal of the Philosophy of Science 47*, 1996, pp. 113-123,
6. Karimi, K. and Hamilton, H.J., Finding Temporal Relations: Causal Bayesian Networks vs. C4.5, *The Twelfth International Symposium on Methodologies for Intelligent Systems (ISMIS'2000)*, Charlotte, NC, USA, October 2000, pp. 266-273.
7. Karimi, K. and Hamilton, H.J., Learning With C4.5 in a Situation Calculus Domain, *The Twentieth SGES International Conference on Knowledge Based Systems and Applied Artificial Intelligence (ES2000)*, Cambridge, UK, December 2000, pp. 73-85.
8. Karimi, K. and Hamilton, H.J., Logical Decision Rules: Teaching C4.5 to Speak Prolog, *The Second International Conference on Intelligent Data Engineering and Automated Learning (IDEAL 2000)*, Hong Kong, December 2000, pp. 85-90.
9. Karimi, K. and Hamilton, H.J., RFCT: An Association-Based Causality Miner, *The Fifteenth Canadian Conference on Artificial Intelligence (AI'2002)*, Calgary, Alberta, Canada, May 2002, pp. 334-338.
10. Karimi, K. and Hamilton, H.J., Temporal Rules and Temporal Decision trees: A C4.5 Approach, *Technical Report CS-2001-02*, Department of Computer Science, University of Regina, Regina, Saskatchewan, Canada, December 2001.
11. Karimi, K. and Hamilton, H.J., Discovering Temporal Rules from Temporally Ordered Data, *The Third International Conference on Intelligent Data Engineering and Automated Learning (IDEAL 2002)*, Manchester, UK, August 2002, pp. 334-338.
12. Karimi, K., and Hamilton, H.J. TimeSleuth: A Tool for Discovering Causal and Temporal Rules, *The 14th IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2002)*, Washington DC, November, 2002, pp. 375-380.
13. Karimi, K., and Hamilton, H.J., Distinguishing Causal and Acausal Temporal Relations, *The Seventh Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD'2003)*, Seoul, South Korea, April/May 2003.
14. Kennett, R.J., Korb, K.B., and Nicholson, A.E., Seabreeze Prediction Using Bayesian Networks: A Case Study, *Proc. Fifth Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD'01)*. Hong Kong, April 2001.
15. Keogh, E. J. and Pazzani, M. J., Scaling up Dynamic Time Warping for Data Mining Applications, *The Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2000)*, August 2000.
16. Korb, K. B. and Wallace, C. S., In Search of Philosopher's Stone: Remarks on Humphreys and Freedman's Critique of Causal Discovery, *British Journal of the Philosophy of Science 48*, 1997, pp. 543-553,
17. Levy, S., *Artificial Life: A Quest for a New Creation*, Pantheon Books, 1992.

18. Mannila, H., Toivonen, H. and Verkamo, A. I., Discovering Frequent Episodes in Sequences, *Proceedings of the First International Conference on Knowledge Discovery and Data Mining*, 1995, pp. 210-215,
19. Pearl, J., *Causality: Models, Reasoning, and Inference*, Cambridge University Press. 2000.
20. Quinlan, J. R., *C4.5: Programs for Machine Learning*, Morgan Kaufmann, 1993.
21. Scheines, R., Spirtes, P., Glymour, C. and Meek, C., *Tetrad II: Tools for Causal Modeling*, Lawrence Erlbaum Associates, Hillsdale, NJ, 1994.
22. Spirtes, P. and Scheines, R., Reply to Freedman, In McKim, V. and Turner, S. (editors), *Causality in Crisis*, University of Notre Dame Press, 1997, pp. 163-176,
23. Wallace, C. S., and Korb, K. B., Learning Linear Causal Models by MML Sampling, *Causal Models and Intelligent Data Management*, Springer-Verlag, 1999.
24. http://www.cs.uregina.ca/~karimi/downloads.html/URAL.java
25. http://typhoon.bae.lsu.edu/datatabl/current/sugcurrh.html. Contents change.

Appendix 1

Significance Level	Relations
0.001	airt •–• maxwnd, airt •–• humid, maxwnd •–• avgwnd, avgwnd •–• wnddir, solar •–• soilt, rain
0.005	Maxwnd •→ airt, Humid •→ airt, maxwnd •→ avgwnd, wnddir •→ avgwnd, solar •–• soilt, rain
0.01	Maxwnd •→ airt, humid •→ airt, maxwnd •→ avgwnd, wnddir •→ avgwnd, solar •–• soilt, rain
0.05	airt ↔ maxwnd, airt → avgwnd, humid •→ airt, maxwnd ↔ avgwnd, maxwnd → wnddir, avgwnd wnddir#, solar •–• soilt, rain
0.1	airt ↔ maxwnd, airt → avgwnd, humid •→ airt, maxwnd ↔ avgwnd, maxwnd → wnddir, avgwnd wnddir#, solar •–• soilt, rain
0.2	airt ↔ maxwnd, airt → avgwnd, humid •→ airt, rain •→ maxwnd, maxwnd → avgwnd, maxwnd → wnddir, avgwnd → wnddir, humid •–• soilt, solar •–• soilt

TETRAD's results with the weather data.

Airt = Air Temperature
Maxwnd = Max Wind Speed
Avgwnd = Average Wind Speed
Wnddir = Wind Direction
Solar = Solar Radiation
Soilt = Soil Temperature
Humid = Humidity
Rain = Rain