Temporal Needleman–Wunsch

Haider Syed

Department of Computer Science
Dartmouth College
Hanover, NH 03755
haider.syed@cs.dartmouth.edu

Amar K. Das

Department of Biomedical Data Science
Geisel School of Medicine at Dartmouth
Hanover, NH 03755
amar.k.das@dartmouth.edu

Abstract—The Needleman–Wunsch algorithm (NW) marked the genesis of a new field of research known as sequence alignment. Its inception was motivated by the growing need for automated methods to find homologous biological sequences. Subsequently, sequence alignment has established itself as a standard approach in bioinformatics, and has also been applied to other domains, including sequences of temporal events. Little prior work has been undertaken on alignment methods in using the temporal information associated with event sequences. In this manuscript, we propose the Temporal Needleman–Wunsch (TNW) algorithm, which modifies the NW approach in a principled manner to use time between events for scoring an alignment. To the best of our knowledge, this is the first formal temporal-extension of a global alignment algorithm. We show that the modification can also be used with the Smith-Waterman algorithm that performs local sequence alignment. We introduce an efficient implementation strategy for the TNW that ensures that the time-complexity is not worse than the NW. We test the abilities of the TNW on event-log data from Electronic Medical Records to identify treatment protocols and to cluster patients based on sequence similarity.

Keywords—Temporal, Needleman-Wunsch, sequence alignment, similarity, data mining, algorithms, Smith-Waterman, cancer care, electronic health records, dynamic programming

I. INTRODUCTION

In the 1970s, the growing need for a way to assess similarity between molecular sequences pioneered a field of research known as sequence alignment [1]. Since its inception, alignment methods have established themselves as the de facto standard for assessing biological sequence homology. The marked success of the Needleman–Wunsch (NW) algorithm [1] motivated the development of a multitude of alignment methods, which have also been adapted for other applications such as sociology [2, 3, 4, 5], natural language processing [6, 7, 8], and healthcare applications [9, 10, 11, 12, 13, 14]. As the methods were created for aligning molecular sequences, which are inherently static in nature, they do not consider temporal information; few efforts have proposed time-based extensions to the approaches [14, 15]. Problems across many domains can be reduced to sequence alignment of discrete dynamic sequences. There is a strong need for principled and efficient sequence alignment approaches which can account for temporality of events during the alignment. Many efforts have developed time-series alignment and matching methods for continuous time-series data [16-20]. However, these methods are meant for continuous data and do not directly and deterministically penalize for missing pieces in the alignment; instead, they use quantized representations for a set of signals to find their best alignment.

Our motivation to develop a temporal sequence alignment method is driven by the need for robust methods that can find patterns in vast amounts of temporal information collected within the healthcare system. In particular, we seek to find homologous patients based on medical histories, where the sequence of and amount of time that elapses between symptoms, diagnoses, and treatments is important for clinical interpretation. If two patients have similar sequences of medical events but one is missing certain diagnoses or treatments, these differences may indicate that the two patients are very different. In developing a sequence alignment method for healthcare data, we need a method that can assess similarity between two sequences by explicitly penalizing for missing events in one of the sequences while also accounting for the time that elapses between events. The method needs to consider the relative timing between events and not the absolute time.

In this manuscript, we introduce the Temporal Needleman–Wunsch (TNW) algorithm, which extends the NW method to include the time between events in the alignment so it can be used for temporal data mining. Previous applications of NW on temporal information have not included temporal extensions to the method. In Section III-C-I, we first present a simplified extension, which we call the memoryless version of the TNW algorithm. The memoryless version considers temporal information in the alignment, but does not account for timing information between events that align with inserted gaps. In Section III-C-II, we then present the naïve implementation of the full TNW algorithm, which considers temporal information when making alignments across the inserted gaps. A naïve implementation of this method has a time-complexity of $O(N^3)$ when aligning a pair of sequences of length N. In Section III-C-III, we present an efficient implementation of the TNW algorithm which has the same time-complexity as the static NW algorithm, $O(N^2)$.

We test the algorithm on data from Electronic Medical Records (EMRs) of breast cancer patients receiving chemotherapy. The timing of treatment events is an important consideration during breast-cancer treatments, and most patients receiving chemotherapy are treated according to recommended protocols of treatment sequences specified by national guidelines. In the first evaluation, we create temporal

sequence representations of the patient treatment histories and for the protocols recommended by clinical guidelines. We use the TNW algorithm to score how well the patient sequences match against recommended protocols. For each patient, we use the highest scoring protocol as the protocol predicted for the patient. Using this approach, we apply the TNW to identify the best-matched chemotherapy protocol and compare our prediction to the protocol identified by manual review, which was performed by one of the authors who is a physician.

II. RELATED WORK

Matching time-series data is a well-studied problem [16-20]. A popular approach for matching continuous time signals is to use a class of algorithms called Dynamic Time Warping [16, 17]. These methods warp the time-axis of the signals in order to find the best alignment between discretized versions of the signals. Some other methods involve Fourier [18] transforms, wavelet [19] transforms or piece-approximations of the signals [20] to represent and subsequently compare them [18]. All these approaches quantize and represent the signals of interest in some manner and try to produce the optimal alignment. However, these methods look for general trends in the data and do not directly and deterministically penalize for specific missing events. For discrete timed-event sequences, such as those found in healthcare, it can be important to penalize for missing events between a pair of sequences; furthermore, the relative timing between events can also be of importance and therefore differences in timing between event occurrences should also be considered during the alignment.

The alternative is to use traditional sequence alignment approaches for discrete sequences, to align discrete temporal sequences. These methods account for the ordering of events, however, they do not use the timing between events in scoring the alignment. A few efforts to extend these approaches to be temporal have been proposed [14, 15]. Hajihashemi and Popescu [14] introduce a temporal extension to the Smith-Waterman (SW) algorithm [21] and apply it to time-series sensor data. Their modification incorporates timing into the similarity score by treating it as a gap; therefore, it does not consider timing information for events that align to each other. In [15], sequence alignment is used to analyze human activities; each activity is encoded as a letter code whereby a single character in a sequence represents a minute of the respective event. The resulting sequences are aligned using a static alignment approach; the temporality is accounted for in the spatial encoding of the sequence by repeating events that span multi-minute time periods.

III. METHODS

A. The Needleman–Wunsch algorithm

The NW algorithm [1] is a global sequence alignment method based on dynamic programming. It guarantees the optimal alignment relative to a given scoring schema and gap penalty. For a pair of sequences, $X = x_1, \ldots, x_m$ and $Y = y_1, \ldots, y_n$, the algorithm uses a pre-defined scoring system $S(x, y)$ that quantifies the similarity between sequence elements x and y. The algorithm creates a two-dimensional similarity-score matrix, H, which is defined as:

$$H_{r0} = -rg; \ H_{0c} = -cg \ \forall \ r \in [0,m], c \in [0,n] \quad (1)$$

$$H_{i,j} = \max \begin{cases} H_{i-1,j-1} + S(x_i, y_j), \\ H_{i-1,j} - g, \\ H_{i,j-1} - g \end{cases}, \quad (2)$$

$$\forall \ i \in [1,m], j \in [1,n]$$

where g is a constant gap penalty that is decided by the user. Inserting a gap into the alignment penalizes the score. After the H matrix is setup, the algorithm finds the maximum value in the matrix, which is the final score for the optimal alignment.

A back-tracking step is then used to figure out the optimal alignment of the sequences. In the proposed approach, the temporal footprint of the sequences being aligned is incorporated into the scoring process that builds the H matrix. In order to facilitate the process, we create temporal sequences, which codify the timing information directly into the sequences that will be compared by the algorithm.

B. Pre-Processing: Temporal Sequences

A static set of event-log data is converted into temporal sequences, which are subsequently aligned by the algorithm. The dates of the events in the log data can be used to ascertain the amount of time that elapses between consecutive events, referred to as the transition time; codifying this intra-event time-delay directly into the sequences creates temporal sequences representing the data. As the transition time, t, is the time-distance between two consecutive events, say event A and B, it can either be appended as a suffix to A or as a prefix to B. The two possible encodings would be: A.t B.0 and 0.A t.B; we refer to the former as a suffix-encoded (SE) sequence and the latter as a prefix-encoded (PE) sequence.

C. Temporal Needleman-Wunsch

The proposed TNW uses temporal content in assessing similarity between sequences. Given a pair of timed-event sequences: $X = x_1.t_{x1}, x_2.t_{x2}\ldots, x_m.0$ and $Y = y_1.t_{y1}, y_2.t_{y2}, \ldots, y_m.0$, where t_{xi} and t_{yi} are the transition times for sequence elements x_i and y_j, respectively, for $i \in [1,m-1]$ and $j \in [1,n-1]$. In applications where timing is important, the similarity computation between each event of the two sequences should use the appropriate transition times in the calculation. A traditional NW approach ignores timing completely. We first propose a simple temporal extension to the NW algorithm that does not account fully for transition time between matched event pairs, called the memoryless TNW in Section III-C-I; in Section III-C-II, we detail the TNW algorithm that maintains information on transition times to evaluate temporal alignment across multiple possible matching event pairs.

I. Memoryless TNW

The memoryless TNW algorithm uses the transition-times of the events being compared in computing the similarity score between events of the two sequences. The H matrix is computed using equations (1) and (3).

$$H_{i,j} = \max \begin{cases} H_{i-1,j-1} + S(x_i, y_j) - f(t_{xi}, t_{yj}) \\ H_{i-1,j} - g \\ H_{i,j-1} - g \end{cases}, \qquad (3)$$

$$\forall\, i \in [1, m], j \in [1, n]$$

where $f(t_{xi}, t_{yj})$ is a user-supplied temporal-penalty function (TPF) which reduces the match-score $S(x_i, y_j)$ by an amount that depends on t_{xi} and t_{yj}. In this manuscript, we use the following TPF:

$$f(t_{xi}, t_{yj}) = T_p \frac{|t_{xi} - t_{yj}|}{\max(t_{xi}, t_{yj})} \qquad (4)$$

where T_p is a user-defined heuristic and represents the maximum penalty that will be imposed on $S(x_i, y_j)$ for temporal differences. The temporal-penalty applied to event comparisons is the percentage discrepancy between t_{xi} and t_{yj} multiplied by T_p; therefore, the maximum temporal-penalty imposed is T_p.

This approach ignores the transition times for events that are aligned to gaps and does not use them for computing similarity scores which introduces a limitation that can be illustrated using a pair of example sequences: A.t_1, B.t_2, C.t_3, D.0 and A.t_4, D.0, where $t_4 = (t_1+t_2+t_3)$. Figure 1 visualizes the sequences in time; each event is assumed to occur instantaneously, for the sake of simplicity. The alignment and score computed using the memoryless TNW is shown in the second row of Figure 1, and the ideal alignment and score for this example is shown in the last row of Figure 1. Whenever an event in one sequence aligns to an event in the other sequence, we refer to the aligned-pair as a match pair, as labeled in Figure 1.

The memoryless TNW skips the timing associated with events that align with gaps. For the shown example, using suffix-encoded sequences, the temporal penalty applied to the matching A-pair is $-f(t_1, t_4)$ and zero for the matching D-pair. However, if the same sequences were codified as PE sequences, the temporal penalty would have been zero for the matching A-pair and $-f(t_3, t_4)$ for the matching D-pair. Therefore, the choice of encoding affects the overall score of the alignment as it changes the temporal penalty that is applied, and may also change the alignment that is returned. Therefore, with the memoryless TNW, the results of the alignment can change with the choice of encoding: PE versus SE. It is difficult to decide which of the two results are more representative; since transition time is, by definition, the timing between a 'pair' of events, say event A and B, so there is no clear choice between attaching this time to A or to B.

For aligning temporal sequences of patient-treatment records, which is the original motivation for the proposed method, the timing of interest is the time that elapses between the match-pairs. It is possible to extend the memoryless TNW to compute the temporal penalties using the total transition time between consecutive match pairs as exemplified in the third row of Figure 1. Using the total transition time also circumvents the aforementioned problem of encoding choice as it produces the same result independent of encoding style.

Computing the temporal penalty based on total transition time for the example case, the penalty becomes $-f(t_1+t_2+t_3, t_4) = -f(t_4, t_4) = 0$, as shown in the third row of Figure 1. The full TNW can correctly compute this penalty. The algorithm is formalized in the proceeding section, and extends the memoryless approach to mollify the limitation of not using the total transition time between match pairs in computing temporal penalties.

II. The full TNW

As exemplified in the third row of Figure 1, it is desirable to use the total time between the match pairs to compute the temporal penalty. It is, therefore, not enough to base the penalty on the transition time of the match pair alone, $-f(t_{xi}, t_{yj})$, as was done in the memoryless TNW; instead the penalty becomes $-f(t_{S1}, t_{S2})$ where t_{S1} and t_{S2} are transition times associated with the first and second sequence, respectively. For the example in Figure 1, the temporal penalty for event A, given SE sequences, would need to use $t_{s1} = t_1+t_2+t_3$ and $t_{s2} = t_4$, therefore, the algorithm has to retrieve all the transition times associated with events that align to gaps between the matched-A pair and matched-D pair. However, this poses a challenge. When comparing event A of the two sequences, we do not know a priori that the alignment of events that proceed the matching pair of event A will have two indels as we have not computed the alignment for events that proceed A yet. Therefore, when computing the temporal penalty for the A-pair, it is impossible to include the transition times for those proceeding events that will subsequently align to gaps, as we do not yet know the alignment of events that proceed event A. However, our goal is to compute the time penalty for differences in timing that exist between the matched A-pair and matched D-pair, so it is equivalent to attach this penalty to the matched D-pair instead of calculating it when scoring the A-pair. Doing it this way, we now need to look at the alignment of events preceding the D-pair to figure out which transition times to include in computing the penalty. This circumvents the problem we faced when trying to compute the penalty when aligning the A pair, as that would require looking ahead to see if the alignment for proceeding events will feature gaps which is not possible.

Therefore, the limitation was artificially introduced as SE sequences were being used: A.t_1, B.t_2, C.t_3, D.0 and A.t_4, D.0, and can be overcome by using the equivalent PE representation of the sequences: 0.A t_1.B t_2.C t_3.D and 0.A t_4.D. With PE sequences, it is possible to compute the temporal-penalty for match pairs using the total intra-event time-delay between the current and preceding match pairs. Our desired alignment score for the example is: $S(A,A) - 2g - \{S(D,D) - f(t_1+t_2+t_3, t_4)\}$ which is computed by the full TNW algorithm presented next.

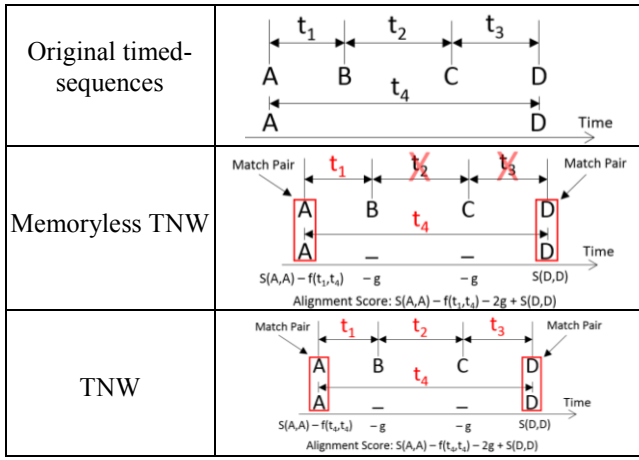| Original timed-sequences	
Memoryless TNW	
TNW	

Fig. 1. Aligning and scoring two example sequences (top row) based alignment and score computed using the Memoryless TNW (middle row) and based on ideal alignment and score using the TNW (bottom row).

a) The Naïve Algorithm

Given a pair of prefix-encoded timed-event sequences: $X = t_{x1}.x_1, t_{x2}.x_2 \ldots, t_{xm}.x_m$ and $Y = t_{y1}.y_1, t_{y2}.y_2, \ldots, t_{ym}.y_m$, where t_{xi} and t_{yi} are the transition times for sequence elements x_i and y_j, respectively, for $i \in [1,m-1]$ and $j \in [1,n-1]$. The naïve implementation of the TNW algorithm uses a score matrix, H, a trace-back matrix, D, and variables c_i and c_j to compute the global alignment. The H matrix is created using (1) and (5).

$$H_{i,j} = \qquad (5)$$

$$\max \left\{ \begin{array}{l} H_{i-1,j-1} + S(x_i, y_j) - f\left(\sum_{q=0}^{c_i} t_{x(i-q)}, \sum_{r=0}^{c_j} t_{y(j-r)} \right) (*) \\ H_{i-1,j} - g \\ H_{i,j-1} - g \end{array} \right\},$$

$$\forall i \in [1,m], j \in [1,n]$$

where c_i and c_j are determined using the following algorithm; the matrix D, is defined by (6):

current_direction = D(i-1,j-1);

$c_i = 0$; $c_j = 0$;

if current_direction equals '-1'

  $c_i = c_i + 1$

else if current_direction equals '1'

  $c_j = c_j + 1$

while current_direction ≠ '0' {

  if current_direction equals -1

    $c_i = c_i + 1$;

  if current_direction equals 1

    $c_j = c_j + 1$;

  current_direction = D(i-c_i, j-c_j)

}

The D matrix is the trace-back matrix which records the outcome of the three comparisons at each cell of $H_{i,j}$. As we compute each cell of H, we choose between three possibilities: a match between the events, an indel inserted in the first sequence or an indel inserted into the second sequence. The trace-back matrix, D, which is part of the original NW method records this choice and is used to determine the alignment computed by the algorithm; the D matrix is computed using (6).

$$D(i,j) = \left\{ \begin{array}{l} 0, if\ H_{i,j} = * \qquad (6) \\ -1, if\ H_{i,j} = H_{i-1,j} - g \\ 1, if\ H_{i,j} = H_{i,j-1} - g \end{array} \right.$$

$$\forall i \in [1,m], j \in [1,n]$$

The intuition behind the method is that whenever we have a matched pair between sequences, transition times for all events that align to gaps and directly precede the match pair must be used in computing the temporal penalty for the matched events. Therefore, whenever we encounter a match, we follow the path for the alignment backwards until we reach the previous matched pair; as we traverse the path, we separately accumulate the total number of gaps encountered in sequence 1, c_i, and sequence 2, c_j. In order to compute the temporal penalty for a matched pair, we then accumulate the total transition time between the current matched-pair and the previous matched pair with the aid of the c_i and c_j values. Using this implementation, at every cell, we must follow the path backward until the last matched pair is encountered; this leads to a time-complexity of $O(m.n.max(m,n))$ or $O(n^3)$ if both sequences under comparison are of length n.

For the example shown in Figure 1, the resulting H matrix is shown in Figure 2. In each cell of H, we also show the value of D for that cell inside brackets; for simplicity, we represent a 0 in the D matrix using a 'd', a -1 in the D matrix using a 'v' and a 1 in the D matrix using an 'h'. Furthermore, we have also super-imposed the value of the variables c_i and c_j in the cells where we have a match pair. The $H_{1,1}$ represents a match between the A events; to compute c_i and c_j for this cell, we check the value of $D_{0,0}$ which is a 'd' since $H_{0,0}$ is stored as a match so c_i and c_i are both zero and the temporal penalty for $H_{1,1}$ reduces to $-f(t_{x0}, t_{y0}) = -f(0,0) = 0$. The $H_{4,2}$ cell is a also a match; the direction for this cell is therefore a 'd'. Therefore, we move to the cell that is diagonally across (3,1) and check the direction, $D_{3,1}$. The direction is 'v' so we increment c_i by 1. The direction stored in cell (3,1) was 'v' so we move to the cell vertically above it and check the direction stored in cell (2,1), $D_{2,1}$, since it is 'v', we increment c_i once again and move to the cell vertically above (2,1) and arrive at cell (1,1). The direction for (1,1), $D_{1,1}$ is 'd' which means we have found the preceding match pair so we return the value of c_i and c_j as 2 and 0, respectively.

		0.A	t4.D
	0 (d)	-0.5 (h)	-1 (h)
0.A	-0.5 (v)	1 (d) $c_i = 0$, $c_j = 0$	0.5 (h)
t1.B	-1 (v)	0.5 (v)	0 (v)
t2.C	-1.5 (v)	0 (v)	-0.5 (h)
t3.D	-2 (v)	-0.5 (v)	1 − * (d) $c_i = 2$, $c_j = 0$

* $f(t_1 + t_2 + t_3, t_4)$

Fig. 2. The H matrix computation using the Naïve implementation of the TNW, for the example shown in Figure 1. In this matrix, we also show the corresponding values of D in each cell, where 'd' represents a 0, 'v' represents -1 and 'h' represents 1 in the D matrix computed using (6).

a) Efficient Implementation

Each cell in the H matrix has a path that leads to $H_{0,0}$, this path must go through one of the mxn cells of the H matrix. Therefore, multiple paths from different cells will traverse the same cells of H; i.e. there are many paths with shared sub-paths. The implementation described for the TNW in the previous section suggests traversing the path backward from a specific cell until we find the previous match pair and to then compute the cumulative transition time between match pairs to find the temporal penalty. The time-complexity of this algorithm can be reduced using dynamic programming. Looking at Figure 1, it is possible to see that whenever we insert a gap, the transition time for the event that aligns with a gap will eventually be involved in a temporal penalty calculation if we encounter a match pair eventually. A second observation is that whenever two events align, transition times associated with the matching events and transitions for events prior to them will never be involved in computing temporal penalties for events that proceed the matched pair. Using these facts, an efficient implementation of the algorithm accumulates the total transition time associated with events that align to gaps and subsequently uses the cumulative time to compute the temporal penalty when a match-pair is encountered; upon matching two events, the accumulated transitions are reset to zero. The proposed approach uses an H matrix, a trace-back matrix, D, and two additional matrices TR and TC to compute the global, temporal alignment between two sequences. In this case, H is computed using (1) and (7):

$$H_{i,j} = \qquad (7)$$

$$\max \left\{ \begin{array}{l} H_{i-1,j-1} + S(x_i, y_j) - f(t_{xi} + TR_{i-1,j-1}, t_{yj} + TC_{i-1,j-1}) \\ H_{i-1,j} - g \\ H_{i,j-1} - g \end{array} \right\},$$

$$\forall\, i \in [1, m], j \in [1, n]$$

where the TR and TC matrices accumulate the transition times for events that align to gaps in sequence X and Y, respectively; these timings are reset to zero whenever a match-pair is encountered; TR and TC are computed using:

$$TR_{i,j} = \begin{cases} 0, & \text{if } D(i,j) = 0 \\ TR_{i,j-1}, & \text{if } D(i,j) = 1 \\ TR_{i-1,j} + t_{xi}, & \text{if } D(i,j) = -1 \end{cases} \qquad (8)$$

$$TC_{i,j} = \begin{cases} 0, & \text{if } D(i,j) = 0 \\ TC_{i,j-1} + t_{yj}, & \text{if } D(i,j) = 1 \\ TC_{i-1,j}, & \text{if } D(i,j) = -1 \end{cases}$$

$$\forall\, i \in [1, m], j \in [1, n]$$

In this proposed implementation, the transition time for gapped events are accumulated as a running total and used up whenever match pairs are encountered. To test the match possibility, the algorithm no longer traces back the path to accumulate transitions for events that aligned to gaps. Instead these transitions have already been accumulated in the TR and TC matrices, and the algorithm has to retrieve a single accumulated value from the relevant cell in TR and TC; and use the respective values to compute the correct temporal penalty and match score as shown in (7). As computing the value for a single cell of the H matrix is still O(1), the overall time-complexity of the proposed approach is O(mn) which is the same as the original NW; furthermore, the space-complexity of the algorithm is also O(mn), which is the same as the NW algorithm. However, the constant attached the order of the space complexity with the efficient implementation is 3, whereas it is 1 for the static NW. This is because we store the TR and TC matrices which each contain mn entries. One observation is that when the H matrix is being computed, the value of any cell (i,j) depends on the values of TR and TC corresponding to the current row, the previous row and the previous column only; i.e. the i-th row, (i-1)-th row and (i-1)-th column. Therefore, instead of maintaining the complete TR and TC matrices, it is sufficient to store two rows and one column from each. Furthermore, if we store vectors and orient the H matrix so that the events of the longer sequence are placed along the row of H, this means that the space required to store the two rows and single column will be 2*min(m,n) + max(m,n). Therefore, the space required for the algorithm will now be O(m,n) + 2*min(m,n) + max(m,n) so the space-complexity now becomes O(m,n) with a scalar of 1, which is exactly the same as the static NW algorithm.

For the example shown in Figure 1, the resulting H matrix is shown in Figure 3. As the efficient implementation of the algorithm recovers the same alignment and score as the naïve implementation, the scores in H and the direction matrices are identical to those shown in Figure 2. In Figure 3, for any cell $H_{i,j}$, we have superimposed the values of $TR_{i,j}$ and $TC_{i,j}$. For simplicity these values are represented as coordinates ($TR_{i,j}$,

$TC_{i,j}$). The $TR_{i,j}$ and $TC_{i,j}$ values represent the accumulated transition times for events that are gapped in the first sequence (0.A t_1.B t_2.C t_3.D) and second sequence (0.A t_4.D), respectively. In any cell (i,j) where the direction is 'v', which represents a gap in the first sequence, the values for ($TR_{i,j}$, $TC_{i,j}$) = ($TR_{i-1,j}$, $TC_{i-1,j}$) + (t_{xi},0). For example, in $H_{2,0}$, the events did not align and the direction is 'v'; therefore ($TR_{2,0}$, $TC_{2,0}$) = ($TR_{1,0}$, $TC_{1,0}$) + (t_{x2},0) = (0,0) + (t_1,0) = (t_1,0). What we have done is to store the transition time of the event B, t_1, because it was aligned to a gap. Similarly, for cell (3,0): ($TR_{3,0}$, $TC_{3,0}$) = ($TR_{2,0}$, $TC_{2,0}$) + (t_{x3},0) = (t_1,0) + (t_2,0) = (t_1+t_2,0). Once again, we have added the transition time of the event C, t_2, to the appropriate cell of TR. Therefore, we maintain a running total of all events that align to gaps in the first sequence using TR and similarly used TC to accumulate the transition times for all events in sequence two that align to gaps. Whenever we encounter a match, we recover the value of TR and TC for the cell that is diagonally across, add it to the transition times of the events being compared and use the sum to compute the time penalty. For example, for cell (4,2), which is a match, the temporal penalty is computed using ($TR_{3,1}$, $TC_{3,1}$) + (t_{x4}, t_{y2}) = (t_1+t_2, 0) + (t_3, t_4) = (t_1+t_2+t_3, t_4). Note that once a match-pair is encountered, the value for (TR, TC) for that cell is reset to 0 as shown for cell (4,2); this represents the fact that we have already computed a temporal penalty based on the transition times for any preceding events that align to gaps and the transition times for the events that just matched. Therefore, subsequent cells that involve this cell in their alignment do not need to use the transition time for the events in this cell or any prior event in computing their temporal penalty.

III. Choosing Parameters of the TNW

The TNW algorithm requires a user-defined scoring scheme and gap penalty, which are also needed for the NW algorithm. Choosing an appropriate scoring scheme depends on the application. In the work presented in this manuscript, we use a simple scoring scheme where exact-matching events get a score of 1, events that do not match get a score of -1.1 and a gap penalty of 0.5. The score of -1.1 for mismatches reflects the preference that instead of aligning mismatching events, we prefer to insert one gap in each sequence. This is achieved because aligning mismatching events would incur a penalty of -1.1, whereas inserting a gap in each sequence would incur an overall penalty of 1. Furthermore, the TNW requires a user-defined temporal-penalty function. We use a simple penalty function that uses the percentage discrepancy between transitions of the events being compared to compute a time penalty. The function requires choosing a heuristic T_p which is the maximum penalty that can be imposed for temporal differences. All the experiments and results presented in this paper have been computed with $T_p = 0.25$. By choosing T_p to be lower than the gap penalty and lower than the absolute difference between distinct scores in the scoring scheme, we encode the fact that the alignment between a pair of sequences should not generally change when computed with and without the temporal distances; although the scores computed in each case will be different.

		0.A	t_4.D
	0 (d) (0,0)	-0.5 (h) (0,0)	-1 (h) (0, t_4)
0.A	-0.5 (v) (0,0)	1 (d) (0,0)	0.5 (h) (0, t_4)
t_1.B	-1 (v) (t_1,0)	0.5 (v) (t_1, 0)	0 (v) (t_1, t_4)
t_2.C	-1.5 (v) (t_1+t_2,0)	0 (v) (t_1+t_2,0)	-0.5 (h) (t_1+t_2,t_4)
t_3.D	-2 (v) (t_1+t_2+t_3, 0)	-0.5 (v) (t_1+t_2+t_3, 0)	1 – * (d) (0,0)

* $f(t_1+ t_2+ t_3, t_4)$

Fig. 3. The H matrix computation for the efficient implementation of the TNW, for the example shown in Figure 1. In this matrix, we show the corresponding values of the D matrix in each cell, where 'd' represents a 0, 'v' represents -1 and 'h' represents 1 in the D matrix computed using (6). Furthermore, we superimpose the values of the corresponding TR and TC matrices computed using (8), as coordinates whereby for cell (i,j), the TR and TC values are presented as ($TR_{i,j}$,$TC_{i,j}$)

D. Extending the Approach to Create a Temporal Smith-Waterman Algorithm

The method proposed can be extended to the SW algorithm which is a local sequence alignment method that guarantees the optimal alignment under a given scoring system. The SW is similar to the NW algorithm with a few modifications. It has four main differences in its implementation:

For the SW algorithm, there are no negative values in the score matrix, H. To implement this, when computing a cell of H, if the best score for the cell is negative, the cell is assigned a value of 0.

The first row and column of H are set to 0, instead of using (1) to initialize their values.

The final alignment score is the maximum value in the H matrix.

The trace-back process in SW starts from the maximum score in the H matrix and traces back the path from that cell, using the D matrix, until it reaches a cell with a score of 0 in the H matrix. By contrast, the NW algorithm always starts the trace-back from the $H_{n+1, m+1}$ cell and ends at the $H_{0,0}$ cell.

Extending the methods presented in this paper to SW, we develop the memoryless-version of a temporal SW, and the efficient implementation for a temporal SW. For both implementations: the first row and column of H are set to 0, the final alignment score is the maximum value of H, and the trace-back procedure used is the one detailed above, in bullet (4). Other modifications are also required which are presented in Sections D-I and D-II.

I. Memoryless Temporal Smith-Waterman

The only modification needed for the memoryless Temporal SW is that when building the H matrix, $H_{i,j}$ is assigned the maximum of the three quantities mentioned in (3) and zero.

II. Efficient Implementation of the Temporal Smith-Watermans Algorithm

Once again, the H matrix is built using (5), but instead of $H_{i,j}$ being the maximum of the three quantities in (5), it is the maximum of the three quantities and zero.

The other change that is necessary is in the equations for TR and TC, which become:

$$TR_{i,j} = \begin{cases} 0, & if\ D(i,j) = 0\ or\ H(i,j) = 0 \\ TR_{i,j-1}, & if\ D(i,j) = 1 \\ TR_{i-1,j} + t_{yj}, & if\ D(i,j) = -1 \end{cases} \quad (9)$$

$$TC_{i,j} = \begin{cases} 0, & if\ D(i,j) = 0\ or\ H(i,j) = 0 \\ TC_{i,j-1} + t_{xi}, & if\ D(i,j) = 1 \\ TC_{i-1,j}, & if\ D(i,j) = -1 \end{cases}$$

$$\forall\ i \in [1,m], j \in [1,n]$$

E. Experiments and Results

I. Dataset

a) Patient Sequences

Our dataset comprises of event logs for breast cancer patients who received chemotherapy treatments. The patient records were extracted from the institutional EMR of Dartmouth-Hitchcock Medical Center on November 1, 2014 and were de-identified for research use according to an approved human-subject study plan. Each of the 178 selected patients underwent chemotherapy treatments with approved chemotherapy agents for invasive breast cancer (Stage I-IV). We encoded each chemotherapy agent used for breast cancer as a unique single-letter code. For example, Cyclophosphamide is denoted as C. The time-stamped patient event logs specify the drugs they received during their treatments; we convert these records into temporal PE sequences.

b) Recommended Chemotherapy Protocol Sequences

Patients who have cancer are often treated according to nationally defined standards of chemotherapy regimens that are based on clinical trials research. Chemotherapy regimens comprise of single or multiple drug agents administered at set time intervals and given over multiple cycles. There are often several recommended treatment regimens that have the same agent or agents but have different sequences and cycles. One of the pressing clinical research questions using cancer treatment data derived from EMRs is (1) determining which chemotherapy protocols patients are receiving, since EMRs do not capture treatment planning, and (2) evaluating how effective they are in real-world practice compared to clinical-trial studies. To understand how well our TNW method can apply to such questions, we test the method with the treatment data we have collected on breast cancer patients. We first create a knowledge base of recommended treatment protocols by reviewing the NCCN breast cancer treatment guidelines for the years 2011 through 2014 [22-25], covering the period of treatment data we derived from the EMR. We identified 43 distinct protocols during this period and encoded the protocols as timed-event sequences, using single-letter codes for the chemotherapy drugs and PE sequence representation. For example, if a protocol recommends four weekly cycles of C, the sequence would be 0.C 7.C 7.C 7.C.

For each of the 178 patients, one of the authors (AKD, a physician researcher) manually annotated the protocol the patient most likely underwent. We found that 115 patients could be clearly matched to one of the 44 protocols. The remaining patients either received combinations of drugs not found in the NCCN guidelines, which may represent clinical trials, or were not given enough cycles of drug agents to be matched to a specific protocol.

In Section E-II, we score each patient in the cohort against the encoded protocol sequences to autonomously identify their treatment regimen and compare our predicted protocol against the manual annotation.

II. Matching Patients to Chemotherapy Protocols

We evaluate the performance of the TNW algorithm for automatically identifying the chemotherapy protocol a patient is undergoing based on their chemotherapy treatment history. As mentioned earlier, 115 of the patients were manually matched to a clinically recommended protocol from the NCCN guidelines; the remaining patients could not be matched uniquely to one of the recommended NCCN protocols. For the 115 manually annotated patients, we use the TNW algorithm to automatically predict the protocol they were following. The process involves scoring each patient against every recommended protocol using the TNW. The predicted protocol for a patient is the highest scoring protocol; we accurately recognize the patient's protocol for 107 patients, which represents 93% accuracy. Instead of using the top-scoring protocol as the predicted regimen for a patient, we also used the two highest scoring protocols for a patient as possible protocols, and we find the correct match in 98% of cases. We only miss 2 of the 115 patients in matching her treatment history to the correct protocol. As mentioned, the recommended protocols consist of treatment patterns with identical sequences of drug events but different transition times between events, such as biweekly versus monthly. Without a temporal sequence alignment algorithm, such as the TNW, we would not be able to discern between protocols that only differ in their temporal footprints.

F. Discussion

The proposed method, TNW, shows strong promise for automatically identifying the chemotherapy protocol that a patient is following, as seen in Section E-II. For the patient data, we correctly match the patient to the correct manually annotated protocol in 93% of cases. However, it should be noted that the manual annotation of the patients found only 14 of the 44 recommended protocols represented in the patient data. Furthermore, five of the 14 protocols were only seen in a single patient. This may a reflection of the fact that physicians have a preference for certain protocols; however, the proposed method needs to be tested on a larger dataset of patients in order to fully test efficacy for matching patients to clinical guidelines. It should be noted there are several protocols in the clinical-recommendations where a pair of protocols have identical events but different temporal signatures. Such protocols were seen in 24 patients; a static sequence alignment algorithm would have no possibility of uniquely identifying the protocol the patient received, as the protocols only differ in their temporal footprint. However, for all 24 patients, the TNW algorithm was able to correctly identify the protocol being followed.

An alternative to the approach presented in this paper is to use temporal alignment methods [16-20] for time-series data. Such approaches align time-sampled signals, such as sensor data, to find the best match between signals. Such approaches do not have established penalties for mismatching specific events and consider absolute time during the alignment and not relative time between events. Therefore, time-series methods have limited utility for aligning discrete temporal sequences for some applications, such as trying to match chronological patient treatment records to each other or to established guidelines. The temporal extension to the SW algorithm proposed by Hajihashemi and Popescu [14] can also be adapted to the NW algorithm to create a variant of the suggested TNW algorithm. However, their approach is to include temporal distance in the alignment by penalizing for temporal differences only when an event aligns to a gap; therefore, it does not account for temporal distance between aligned events in the two sequences. Ideally, temporal variations for aligned events would be considered. In particular, for patient treatment records, if two sequences have identical events, their score should still be affected by temporal variations that exist in the sequence. Using the method in [14], the two sequences would receive a score identical to the one that would computed by a static NW or SW.

The method proposed by Shoval and Isaacson [15] performs temporal sequence alignment by codifying multi-minute activities as multiple, consecutive occurrences of the same character. As a temporal resolution must be chosen during the sequence encoding, only discrete time-steps can be represented so the method cannot support truly continuous values. Furthermore, the method generally increases the length of the encoded sequences, worsening computation-time by a factor dependent on the underlying temporal-footprints of the sequences in question. Moreover, since the time is treated in conjunction with scoring the event similarity, it cannot be completely decoupled from the underlying event-scoring function and so cannot support non-linear temporal-penalty functions. By contrast, our proposed TNW approach does not change the length of the sequences in question, treats time as a separate dimension from the spatial event-matching, which allows the use of temporal-penalty functions that are independent of the spatial-alignment task, and supports continuous timing. Moreover, the TNW can be implemented to have the same time-complexity as the static NW algorithm albeit with a slower run-time, which is independent of the timing of the events.

G. Conclusions

Sequence alignment methods for static data are pervasive. However, temporal data mining requires alignment methods that use timing in computing the alignment of the sequences. Very few efforts have tried to create such methods. In some domains, such as healthcare records, the relative time between events is clinically important for the interpretation and evaluation of data. Using static sequence alignment approaches on these event sequences ignores the timing information, which is a significant factor that should be involved in assessing similarity. In healthcare records such as patient treatment histories, merely assessing similarity based on treatments shared by a pair of patients may not be sufficient as the absence of treatments can be a clinical indication that the patients suffer from different underlying ailments. One option to align such sequences is to use time-series alignment methods that use temporal information in the alignment, however, such methods do not apply a deterministic penalty for missing events and further use absolute time in computing the alignments instead of the relative-time between treatments which is the measure of interest in treatment sequences. Therefore, there is a need for a temporal sequence alignment approach that includes the transition time between events in computing the alignment for the sequences. We present the first temporal global sequence alignment algorithm. The efficient implementation of the temporal method proposed has the same time and space complexity as the static NW method. Furthermore, we also show how to extend the proposed approach to create a temporal SW algorithm that also has the same time and space-complexity as its static counterpart.

The method is used for autonomous chemotherapy-protocol recognition based on patient treatment histories. Physicians follow standardized and optimal clinically recommended protocols, however, medical records often do not record the specific protocol being followed. It is important to know what regimens a patient has received as it has therapeutic and diagnostic value. Manually abstracting the protocol from a patient's medical record can be a cumbersome and time-intensive process. The method proposed in this manuscript can be used to identify chemotherapy protocols using a patient's event log; the results show the algorithm has an accuracy of 93% if considering the top match as the predicted protocol or 98% in cases where the second best match is considered.

H. Future Work

The TNW algorithm provides a way for temporal sequence alignment when relative-timing information is important. In this manuscript, the algorithm was tested on a cohort of cancer data, however, the method can be applied for problems in other

domains where the relative timing between sequence events should be included in scoring the alignments; the algorithm will be tested on other datasets within health-care and for problems in other domains. An alternative application of this algorithm could be for sentence-matching. The temporal distance between different words could be encoded for different sentences. When comparing two sentences, the difference in timing could be used to affect the similarity score based on the distance between the words of interest in each case.

The algorithm requires a user-specified temporal-penalty function. The exact choice of function is application specific; however, it is possible to create temporal-penalty functions that encapsulate *a priori* information about how timing affects the similarity. Future work will use semantically-driven temporal-penalty functions to evaluate the performance of the algorithm.

The scoring scheme is another user-specified heuristic in the algorithm. The scoring scheme used in this paper is a very simple scheme, however, if an ontology representing the relations between the events of interest is available, it is possible to derive semantically-meaningful scoring schemes based on the distance between different events that may be encountered. For the breast-cancer dataset presented, future work will focus on using ontology-based scoring schemes to compute the alignment scores.

REFERENCES

[1] Needleman, S.B. and Wunsch, C. 1970. A general method applicable to the search for similarities in the amino acid sequence of two proteins. Journal of Molecular Biology. 48, 3 (1970), 443-453.

[2] Blair-Loy, M. 1999. Career Patterns of Executive Women in Finance: An Optimal Matching Analysis. *American Journal of Sociology*. 104, 5 (1999), 1346-1345.

[3] Abbott, A. and Hrycak, A. 1990. Measuring Resemblance in Sequence Data: An Optimal Matching Analysis of Musicians' Careers. *American Journal of Sociology*. 96, 1 (1990), 144.

[4] Poole, M. and Holmes, M. 1995. Decision Development in Computer-Assisted Group Decision Making. *Human Communication Research*. 22, 1 (1995), 90-127.

[5] Abbott, A. and Barman, E. 1997. Sequence Comparison Via Alignment and Gibbs Sampling: A Formal Analysis of the Emergence of the Modern Sociological Article. *Sociological Methodology*. 27, 1 (1997), 47-87.

[6] Regina Barzilay and Lillian Lee. 2002. Bootstrapping lexical choice via multiple-sequence alignment. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing - Volume 10* (EMNLP '02), Vol. 10. Association for Computational Linguistics, Stroudsburg, PA, USA, 164-171. DOI=10.3115/1118693.1118715 http://dx.doi.org/10.3115/1118693.1118715

[7] Figueroa, A. and Atkinson, J. 2009. Intelligent answering location questions from the web using molecular alignment. *Journal of Intelligent Information Systems*. 35, 1 (2009), 75-90.

[8] Regina Barzilay and Lillian Lee. 2003. Learning to paraphrase: an unsupervised approach using multiple-sequence alignment. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1* (NAACL '03), Vol. 1. Association for Computational Linguistics, Stroudsburg, PA, USA, 16-23. DOI=10.3115/1073445.1073448 http://dx.doi.org/10.3115/1073445.1073448

[9] Mans, R.S., Schonenberg, M.H., Song, M., van der Aalst, W.M.P., Bakker, P.J.M.: Application of process mining in healthcare - a case study in a Dutch hospital. Biomedical Engineering Systems and Technologies. Communications in Computer and Information Science, vol. 25, pp. 425–438. Springer, Heidelberg (2009)

[10] Meng, F., D'Avolio, L. W., Chen, A. A., Taira, R. K., & Kangarloo, H. 2005. Generating Models of Surgical Procedures using UMLS Concepts and Multiple Sequence Alignment. *AMIA Annual Symposium Proceedings* 520–524.

[11] Lee, W.N., Das A.K. Local alignment tool for clinical history: temporal semantic search of clinical databases. *AMIA Annu Symp Proc.* pp. 437–441 (2010)

[12] Hares B., Automated plan-recognition of chemotherapy protocols. *AMIA Annu Symp Proc.* 108–114 (2011)

[13] Popescu M., An Ontological Fuzzy Smith-Waterman with Applications to Patient Retrieval In *Electronic Medical Records proceedings of IEEE World Congress on Computational Intelligence* (Barcelona, Spain, July 18-23, 2010).

[14] Hajihashemi Z. and Popescu M. An Early Illness Recognition Framework Using a Temporal Smith Waterman Algorithm and NLP. *Proc AMIA Fall Symp* 2013:549–57

[15] Shoval, N. and Isaacson, M. 2007. Sequence Alignment as a Method for Human Activity Analysis in Space and Time. *Annals of the Association of American Geographers*. 97, 2 (2007), 282-297.

[16] Sakoe, H. and Chiba, S. 1978. Dynamic programming algorithm optimization for spoken word recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*. 26, 1 (1978), 43-49.

[17] Velichko, V. and Zagoruyko, N. 1970. Automatic recognition of 200 words. *International Journal of Man-Machine Studies*. 2, 3 (1970), 223-234.

[18] Faloutsos C., Ranganathan M., and Manolopoulos Y.. 1994. Fast subsequence matching in time-series databases. In *Proceedings of the 1994 ACM SIGMOD international conference on Management of data* (SIGMOD '94), Richard Thomas Snodgrass and Marianne Winslett (Eds.). ACM, New York, NY, USA, 419-429.

[19] Chan K.P., and Fu A.W.C. Efficient time series matching by wavelets. In *ICDE (Sydney, Australia*, March 1999).

[20] Keogh, E., Chakrabarti K., Pazzani M., Mehrotra S. Dimensionality reduction for fast similarity search in large time series databases. *Journal of Knowledge and Information Systems. 2000*;3(3):263–286

[21] Smith, T. and Waterman, M. 1981. Identification of common molecular subsequences. *Journal of Molecular Biology*. 147, 1 (1981), 195-197.

[22] Carlson R.W., Allred D.C., Anderson B.O., et al., Invasive breast cancer. *J Natl Compr Cancer Netw.* 9:136–222 (2011)

[23] National Comprehensive Cancer Network. Breast cancer. NCCN Clinical Practice Guidelines in Oncology, version 1.2012 (2012)

[24] National Comprehensive Cancer Network. Breast cancer. NCCN Clinical Practice Guidelines in Oncology, version 1.2013 (2013)

[25] Breast Cancer Version 3.2014. *J Natl Compr Cancer Netw.* 12:542–90 (2014)