

Temporal Search and Replace: A Graphic-Based Solution to Temporal Event Data Wrangling and Incremental Querying

Megan Monroe, Rongjian Lan, Hanseung Lee, Allan Fong,
Catherine Plaisant, PhD, Seth Powsner, MD, Ben Shneiderman, PhD

University of Maryland, College Park, MD

Abstract

Temporal event data is the essence of Electronic Health Records (EHRs), detailing for each patient a history of what happened and when. From this data, clinical researchers try to identify meaningful patterns of events: medication combinations, treatment side effects, disease symptoms, etc. However, these events are rarely recorded to suit the needs of any one study, requiring researchers to modify their datasets to better capture the events of interest. Command-based and spreadsheet-based tools are difficult to learn, and are notoriously ill-suited to manipulating temporal event data. We introduce a graphic-based interface that employs the familiar concept of “search and replace” to help researchers identify and modify event patterns of interest. The Search and Replace interface was used by multiple clinical research groups to facilitate both data wrangling and query construction.

Introduction

Electronic Health Records (EHRs) provide clinical researchers with a detailed history of patient symptoms, treatments, and outcomes. However, a given study is rarely completed using a single, straightforward query against an EHR database. Studies conducted against preexisting records are typically comprised of two phases, which can be done in any order, and repeated as many times as necessary:

- Isolate the subset of patients that is pertinent to the study or the question at hand.
- Transform the events in each patient record to better reflect the perspective of the study.

Both of these steps can require researchers to identify complex patterns of events within a given record. For example, a study might focus on patients who were treated with Medication X for at least 3 months, and then experienced Symptom Y within one month of stopping X. Maybe Medication X is prescribed in month-long intervals, requiring researchers to manually construct the actual duration of exposure by piecing together each prescription. Maybe Symptom Y is not a single event, but two different events that can occur in any order, so long as they occur within a week of each other.

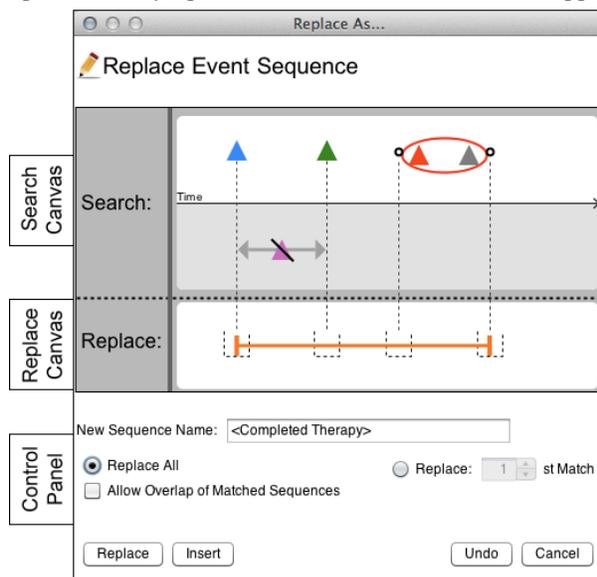


Figure 1: The TSR interface includes a search canvas (top), a replace canvas (middle), and a control panel (bottom). This query is for a blue event, followed by a green event, without an intervening pink event. After that, a red event and a gray event must occur, but in either order. If this sequence is found in a patient record, it is replaced by an orange interval.

Event patterns such as these are usually searched for using command-based and spreadsheet-based tools, which employ query languages that are difficult to learn, are not tailored to temporal event data, and do not provide visual feedback about the impact of the query. In many cases, researchers rely on technical experts to construct their datasets, creating a gap between the data manipulation process and the research domain expertise. To address this problem, we introduce Temporal Search and Replace (TSR), an extension to the advanced search feature of the EventFlow¹ visualization tool that is designed to make querying and modifying temporal event records more accessible and accurate. The interface capitalizes on the fact that most users are familiar with Search and Replace functionality through the use of word processing and spreadsheet applications. We demonstrate its impact on performing common temporal event transformations based on our experience with clinical researchers throughout multiple case studies.

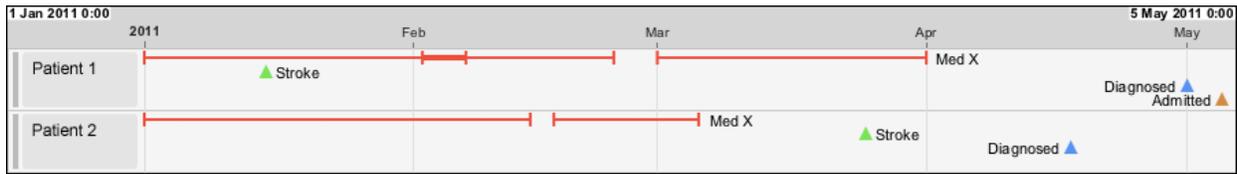


Figure 2: EventFlow’s display of a single patient record. Intervals (such as medication prescriptions) are displayed as a pair of connected bars. Point events (such as diagnoses or reports of symptoms) are displayed as triangles.

Related Work

TSR is based on previous work in both data wrangling and temporal query. Data wrangling is the process of preparing a raw dataset for analysis. This process can be so burdensome and complex that some researchers estimate that it consumes more time than the analysis itself.² Users needing to wrangle temporal event data have two options. The first of these is to master a complex, command-based query language such as SQL or SAS, where data fields can be transformed through creative uses of the “AS” command. The second option is to use prepackaged scripts or stored procedures, which provide shortcuts to common, transformation tasks. However, users in need of transformations that do not exactly conform to these prepackaged scripts are relegated to the first option.

The field of temporal query is comprised of two predominantly disjoint spheres of work, the first of which revolves around incorporating powerful temporal mechanisms into standard, command-based query tools,^{5,6} and the second of which endeavors to make the basic elements of temporal query more accessible to non-technical users.^{3,4} The EventFlow visualization tool bridged this gap with a query system that offered access to a wide range of temporal relationships in an easy-to-learn, usable interface.^{7,9} However, identifying complex temporal patterns still required large and intricately constructed queries. A mistake in one component of such a query would result in a failure of the query as a whole.

The TSR interface and functionality

The TSR interface consists of two canvases, a search canvas and a replace canvas, as well as a small control panel (Figure 1). The search canvas is an exact replica of EventFlow’s advanced search canvas⁷. An event pattern is specified by populating this canvas with event graphics, similar to those used in EventFlow’s individual record display (Figure 2). These graphics are referred to as query elements. Users can click any portion of the search canvas to bring up a short menu where they can specify the details of the query element that they would like to add at that point. They can click on the lower half of the search canvas (in gray) to specify an event that must NOT occur, which are referred to as absence elements. Query elements can be edited or deleted by clicking them.

For each query element that is specified in the search canvas, the replace canvas is populated with a replacement slot, indicated by a dotted line. The replacement slot serves as a placeholder into which users can place one or more replacement events, again by clicking the canvas. Replacement events can be a type of event that already exists in the dataset, or a completely new, TSR-generated marker event type, in which case users can specify both the name of the event, and the color in which it should be displayed. When an event in a patient record is matched to a query element on the search canvas, the timestamp of that event is passed on to any replacement event in that element’s replacement slot. However, there are three types of query elements that require special handling:

- **Absences** – The advanced search allows users to incorporate the absence of an event in their queries (seen in the bottom half of the search canvas). Absences are a semantic requirement that are not matched to any specific event or time, and thus do not provide a timestamp that could be passed on to a replacement event.
- **Permutations** – Users can specify a set of events that can occur in any order. For these query elements, a replacement slot is generated only for the start point and end point of this sequence.
- **Repetitions** – Users can specify that an event or pattern of events must occur a given number of times (e.g. ≤ 3 times). Similar to permutation query elements, replacement slots are generated only for the start point and end point of the repetition sequence.

Once a search pattern and a replacement pattern have been specified, the control panel is used to dictate how the TSR should be executed if the search pattern can be matched to the patient record more than once. Users can specify that a search pattern should be replaced only once, or as many times as possible. If the pattern will only be replaced once, users can choose to replace the first instance of the pattern, or some subsequent instance of the pattern. If the pattern will be replaced multiple times, users can specify whether or not the pattern can overlap with itself. That is, when a replacement pattern is inserted into a record, the search for the next instance of the search pattern can begin either at the beginning of that replacement pattern, or at the end. The former allows patterns to be replaced recursively, with replacement events being matched to elements of the search pattern in subsequent matches. For

example, the interval merging transformation (discussed in the next section) makes use of this option in order to append new intervals to previously merged ones.

In addition to replacements, TSR can be used to delete events entirely, and to insert events into a record without removing anything at all. Event sequences are deleted by simply not specifying a replacement pattern. Insertions can be done by either duplicating elements of the search pattern in the replacement pattern, or by simply using the insert button (Figure 1, lower left). Thus replacement events can be used either to transform data in the patient record, or to generate additional marker events for subsequent analysis, which is discussed in greater detail later. When a TSR is performed, it is added to a stack of TSR modifications. A TSR can be undone only if it appears at the top of this stack. That is, TSR modifications can be undone in the reverse order in which they were performed.

Data wrangling using TSR

With temporal event data, data wrangling can take many forms. Consider the example presented in the introduction where patients are to be selected for a study based on the following event pattern: were treated with Medication X for at least 3 months, then experienced Symptom Y within one month of stopping X. However, Medication X prescriptions appear in the patient records as month-long intervals that can overlap or occur with a gap between them, depending on when the patients happened to pick up their prescription. In this case, the raw data format does not accurately reflect the intention behind this criterion.

To account for this, the first thing a researcher might want to do is to transform those individual prescriptions into new, consolidated intervals that represent the actual exposure time. In fact, this process of interval merging is used by many different research groups as a stored procedure, however these stored procedures are difficult to customize and are limited to this single functionality. Figure 3 demonstrates the use of TSR to perform interval merging and event type merging (combining two different types of events into a single type), two common transformations that, along with countless others, can be customized and executed in a single interface.

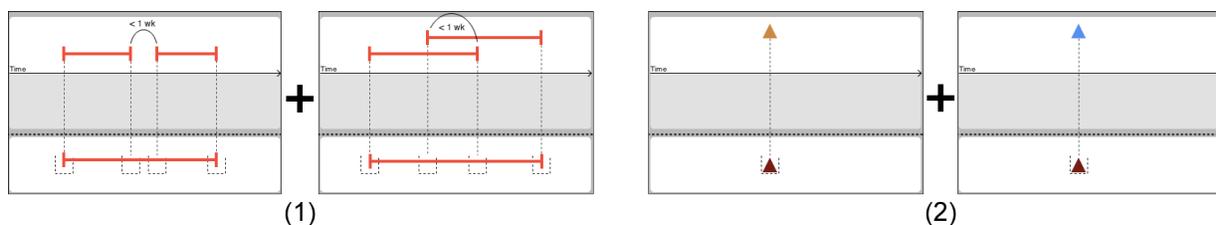


Figure 3: Two common data wrangling transformations are accomplished with two replacements each in TSR. (1) intervals of Medication X that either overlap by less than a week, or are separated by less than a week are merged into a single interval. (2) two different event types (shown in orange and blue), are both replaced by the same event type (shown in dark red).

Incremental temporal query using TSR

Temporal query is critical for isolating sets of patient records that contain a particular event pattern. One of the best strategies for executing these complex queries without making errors is to perform them incrementally, narrowing in on the population of interest step by step. In command-based query tools, this is accomplished using the “AS” statement to save fields and tables as new entities. However, the majority of these command-based tools are designed for relational data structures, and do not provide direct support for temporal event sequences.

TSR allows users to take a similar incremental approach to executing complex queries. Separable components of the query can be executed independently and flagged with TSR-generated marker events. These marker events can then be incorporated into subsequent queries that dictate how the marker events must relate to each other. Returning to our earlier example, consider Symptom Y, which is comprised of two different events that can occur in any order, but must occur within a week of each other. Identifying Symptom Y could begin with a query that replaces any permutation of these two events with a marker interval that spans the duration between them. Records containing a marker interval with duration of less than a week could be queried for and isolated. This marker event, as well as the merged Medication X events discussed in the previous section could then be combined to formulate the final query for qualifying patients. This process, in its entirety, is shown in Figure 4.

This technique of executing queries incrementally using TSR was used substantially in case studies involving researchers at the US Army, Pharmacovigilance Center (PVC), Children’s National Medical Center, and University of Maryland School of Medicine. The TSR interface was also evaluated for learnability and usability with graduate student researchers at the University of Maryland, who ranked the interface with an average score of 5.5 on a 1-7 Likert scale, with a score of 7 representing the best possible experience. The researchers, who were previously limited to only command-based tools, were particularly impressed not just with the ease of use of the TSR interface,

but also with its integration into the EventFlow software. They commented that the interactive visualizations made it extremely easy to see the effects of their TSR modifications as they performed their queries.

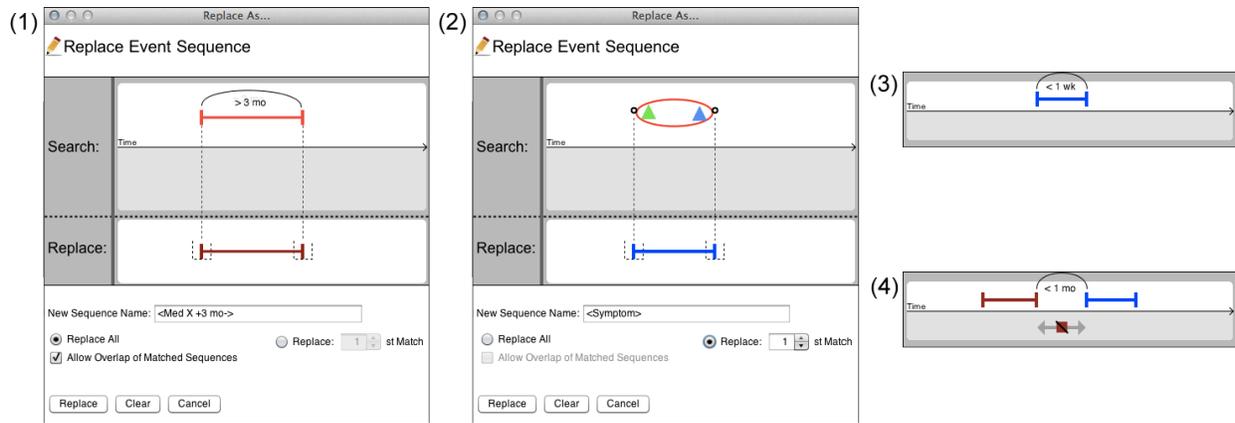


Figure 4: A query is executed incrementally by first replacing Medication X prescriptions exceeding 3 months with a dark red marker intervals (1). Then, both permutations of the two events that comprise Symptom Y are replaced with a blue marker interval (2). Finally, two queries are performed, first to find all the blue marker intervals that do not exceed a week (3), and then to specify the relationship between the dark red and blue marker intervals (4).

Conclusion

Temporal Search and Replace (TSR) provides clinical researchers with expanded capability to transform temporal event data, allowing event sequences to be easily inserted, removed, and modified. The interface greatly facilitates the process of data wrangling, both in performing common operations, as well as complex, customized data alterations. It also allows researchers to break down complex queries into small, manageable increments, making this notoriously difficult and error-prone process more accessible to non-technical experts. Clinical researchers found TSR to be a significant improvement in both usability and capability over standard command-based and spreadsheet-based tools. TSR, and the encapsulating EventFlow application, allowed these researchers to create, explore, and share datasets that better reflected the objectives of their studies.

Acknowledgements

We appreciate the partial support of the Oracle Corporation and the UMIACS Center for Health Informatics and BioImaging.

References

1. EventFlow Webpage: www.cs.umd.edu/hcil/eventflow.
2. S. Kandel, A. Paepcke, J. Hellerstein, and J. Heer. Wrangler: interactive visual specification of data transformation scripts. In *Proc. of ACM Conf. on Human-Computer Interaction (CHI 2011)*, 3363–3372, 2011.
3. Vrotsou, K., Johansson, J., and Cooper, M. Activitree: Interactive visual exploration of sequences in event-based data using graph similarity. In *IEEE Transactions on Visualization and Computer Graphics*, 945–952, 2000.
4. Jin, J., and Szekely, P. Interactive querying of temporal data using a comic strip metaphor. In *2010 IEEE Symposium on Visual Analytics Science and Technology (VAST)*, 163–170, 2010.
5. Jensen, C., Clifford, J., et al. The TSQL Benchmark. In *Proceedings of the International Workshop on an Infrastructure for Temporal Databases*, QQ1–QQ28, 1993.
6. Snodgrass, R. The temporal query language TQuel. In *ACM Transactions on Database Systems (TODS)*, 247–298, 1987.
7. Monroe, M., Lan, R., Morales del Olmo, J., Shneiderman, B., Plaisant, C., Millstein, J., The Challenges of Specifying Intervals and Absences in Temporal Queries: A Graphical Language Approach, *Proc. of ACM Conf. on Human-Computer Interaction (CHI 2013)*, 2349–2358, 2013.
8. Monroe, M., Lan, R., Lee, H., Plaisant, C., Shneiderman, B., Temporal Event Sequence Simplification, to appear in *Proc. IEEE VAST conference* (2013).
9. Meyer, T., Monroe, M., Plaisant, C., Lan, R., Wongsuphasawat, K., Coster, T., Gold, S., Millstein, J., Shneiderman, B., Visualizing Patterns of Drug Prescriptions with EventFlow: A Pilot Study of Asthma Medications in the Military Health System, *Proc. of Workshop on Visual Analytics in HealthCare* (2013)