DEGREE PROJECT IN COMPUTER SCIENCE AND ENGINEERING, SECOND CYCLE, 30 CREDITS

STOCKHOLM, SWEDEN 2018

Sequential Pattern Mining on Electronic Medical Records for Finding Optimal Clinical Pathways

HENRIK EDMAN

KTH ROYAL INSTITUTE OF TECHNOLOGY
SCHOOL OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE

Sequential Pattern Mining on Electronic Medical Records for Finding Optimal Clinical Pathways

Henrik Edman
henedm@kth.se

Program: Master of Computer Science, Civil Engineer

Royal Institute of Technology, Tokyo Institute of Technology
DA222X, Master Thesis in Computer Science
Supervisor at Royal Institute of Technology: Jeanette Hällgren Kotaleski
Supervisor at Tokyo Institute of Technology: Haruo Yokota
Examiner: Hedvig Kjellström

CSC, KTH

27 April 2018

Abstract

Electronic Medical Records (EMRs) are digital versions of paper charts, used to record the treatment of different patients in hospitals. Clinical pathways are used as guidelines for how to treat different diseases, determined by observing outcomes from previous treatments. Sequential pattern mining is a version of data mining where the data mined is organized in sequences. It is a common research topic in data mining with many new variations on existing algorithms being introduced frequently. In a previous report, the sequential pattern mining algorithm PrefixSpan was used to mine patterns in EMRs to verify or suggest new clinical pathways. It was found to only be able to verify pathways partially. One of the reasons stated for this was that PrefixSpan was too inefficient to be able to mine at a low enough support to consider some items. In this report CSpan is used instead, since it is supposed to outperform PrefixSpan by up to two orders of magnitude, in order to improve runtime and thereby address the problems mentioned in the previous work. The results show that CSpan did indeed improve the runtime and the algorithm was able to mine at a lower minimum support. However, the output was only barely improved.

Referat

Electronic Medical Records (EMRs) är digitala versioner av behandlingshistoriken för patienter på sjukhus. Clinical pathways används som riktlinjer för hur olika sjukdomar borde behandlas, vilka bestäms genom att observera utkomsten av tidigare behandlingar. Sequential pattern mining är en typ av data mining där datan som behandlas är strukturerad i sekvenser. Det är ett vanligt forskningsområde inom data mining där många nya variationer av existerande algoritmer introduceras frekvent. I en tidigare rapport användes sequential pattern mining algoritmen PrefixSpan på EMRs för att verifiera eller föreslå nya clinical pathways. Den kunde dock endast verifiera pathways delvis. En av anledningarna som nämndes för detta var att PrefixSpan var för ineffektiv för att kunna köras med en tillräckligt låg support för att kunna finna vissa åtgärder i en behandling. I den här rapporten används istället CSpan, eftersom den ska överprestera PrefixSpan med upp till två storleksordningar, för att förbättra körningstiden och därmed adressera problemen som nämns i den tidigare rapporten. Resultaten visar att CSpan förbättrade körningstiden och algoritmen kunde köras med lägre support. Däremot blev utdatan knappt förbättrad.

Chapter 1 - Introduction

Sequential pattern mining, as the name suggests, is a version of data mining where the data mined is organized in sequences. The purpose is to find interesting subsequences in a set of sequences. How interesting something is, is defined by its occurrence frequency in the set. The sequences typically consists of itemsets, with information about the item. An itemset could contain an identifier for a customer, what was purchased and a timestamp of when something was purchased.

A common example on area of usage given in several papers is to find customer purchase patterns, which can be useful when making decisions of where to put different items in a physical store or give recommendations on what is commonly purchased together with an item for online stores. In the case of a book trilogy for example, people usually buy the first book and then the second and third. It does not matter if other items or books are purchased in between, this pattern is still supported.

When using sequential pattern mining a minimum support (minsup) is set by the user to decide how many of the sequences that an item have to appear in, in order for it to be considered frequent. Frequent sequences are then constructed from these frequent items to form the interesting patterns that are the output.

Electronic medical records (EMRs), are digital versions of paper charts, used to record the treatment and medical history of different patients. They allow for the data to be stored more accurately and for a patient's medical history to be viewed more easily, instead of having to track down a patient's paper records. This can reduce the cost and time spent.

Clinical pathways are used as guidelines for how to treat different diseases, determined by observing outcomes from previous treatments. Medical workers typically generate clinical pathways themselves, based on their experiences. They are used to standardize the treatment process of patients to reduce variability, improving outcome and reducing costs by the use of evidence based practice [2]. Since they are based on experiences, it can be helpful to verify their

correctness or detect possible improvements to them by computationally mining EMRs and examining frequent sequential patterns.

1.1 Previous Work

This work is based on and a continuation of the work done by Uragaki, Keishiro, et al. in the paper *Sequential Pattern Mining on Electronic Medical Records with Handling Time Intervals and the Efficacy of Medicines* [1]. A sequential pattern mining algorithm called PrefixSpan [6] was extended to handle time differences between different steps of a treatment and was named T-PrefixSpan. Sequential pattern mining is a form of data mining used to find statistically relevant patterns. T-PrefixSpan was used to mine a database containing information from EMRs. The ones used in this paper, and [1], are actual EMRs from the Faculty of Medicine, University of Miyazaki Hospital. The goal stated was to find the optimal clinical pathway for different treatments in order to verify existing pathways or suggest variants or new pathways.

The EMRs used in this work are for the treatments 'Transurethral Resection of Bladder Tumors' (TUR-Bt) and 'Cryptorchidism Fusion Surgery' (CFS). Some treatments have clinical pathways with a fixed flow, such as CFS, and some have flows that are not clearly defined, such as TUR-Bt. The code was run on the data acquired from the EMRs for CFS to compare the results to the fixed flow clinical pathway. Treatment articles not containing medicines, such as nursing tasks, matched well. However, treatment articles containing medicines, such as prescriptions and injections, did not match as well. The reason stated in [1] to be the most probable was that PrefixSpan was too inefficient to be able to mine below a certain threshold, meaning some articles did not occur frequently enough to be considered by the algorithm. A number of more novel and efficient algorithms were suggested to be used instead in order to address this issue. These were CSpan [3], Clospan [4] and CLaSP [5]. The algorithm chosen for this work is CSpan since it outperforms the other algorithms mentioned [3].

The reason that these algorithms are more efficient is that, unlike PrefixSpan, they all consider only closed sequences, which are patterns that are not contained in any longer sequences, or supersequences, with the same support. This is explained in more detail in Chapter 2, Background. CloSpan and CLaSP both have in common that they perform post-pruning on a candidate set of closed sequences to remove sequences that might have been closed but ultimately were not. CSpan supposedly outperforms those algorithms by one order of magnitude because of its use of an occurrence check instead of a candidate set [3]. Closed sequences are detected earlier and less sequences have to be held in memory, resulting in a smaller search space for closure checking of new patterns. This is particularly effective in the case of long patterns and low minsup thresholds.

1.1.1 Sequential Pattern Mining Algorithms

There are several different SPM algorithms, with new and more efficient ones appearing frequently. The ones studied in preparation for this work were PrefixSpan [6], CLaSP [5], CloSpan [4] and CSpan [3].

1.1.2 Data Mining and Medicine

Many studies have explored the use of data mining on EMRs. Wright, A.P., et al. determined whether sequential pattern mining was effective for identifying temporal relationships between medications to accurately predict the next medication to be prescribed for a patient [8]. Huang, Zhengxing, et al. performed clinical pathway pattern mining to find which medical behaviours, and in what order, are critical for clinical pathways [9]. Lin, Fu-ren, et al. developed a data mining technique for discovering time dependency patterns of different activities in clinical pathways [10]. Wakamiya and Yamauchi studied the standard functions of clinical pathways embedded in EMRs [11].

1.2 Problem Statement

In [1] the problem was formulated as: "...verifying existing clinical pathways and recommend variants or new pathways by analyzing historical records ... with handling time intervals between treatments".

This work will focus on creating a version of CSpan with time handling by modifying the mining algorithm in T-PrefixSpan, with the goal of improving the running time to be able to obtain a result that conforms better to the fixed flow clinical pathway. For ease, the new algorithm will be referred to as T-CSpan throughout this paper. Because of this the problem is formulated as:

- Will T-CSpan be able to improve the running time compared to T-PrefixSpan?
- Will T-CSpan more accurately conform to the fixed clinical pathway for a specific treatment compared to T-PrefixSpan?
- Determine T-CSpans use in mining datasets of electronic medical records for the purpose of verifying existing clinical pathways and recommending variants or new pathways with handling of time intervals between treatments.

1.3 Purpose

The purpose of this report is to determine if CSpan can be used to replace PrefixSpan in the previously constructed program T-PrefixSpan to correctly verify existing clinical pathways and recommend variants or new pathways by analyzing EMRs. The running time, as stated in [1], was too inefficient to run the program below the minsup threshold and as such could not accurately enough verify an existing clinical pathway.

If the results prove to conform with the established clinical pathway for CFS, the program could potentially be used to suggest variants of, or new, clinical pathways for treatments which might be especially interesting for those pathways without a fixed flow. This could potentially benefit medical science by improving treatment outcomes and reducing costs.

1.4 Ethical, Societal and Sustainability Aspects

As this work concerns patient medical records and data mining of them, patient confidentiality has to be ensured. For this reason, the identities of the patients in the EMRs have been replaced with anonymous IDs.

If a version of T-CSpan would ever be released as a tool used for examining clinical pathways, the database containing EMRs would have to be secure so they are not readily available to those who do not need them, which is the same requirement that hospitals handling EMRs have.

By being able to verify or improve on clinical pathways, this could help both hospitals and patients by further reducing variability in treatments, improving the outcome and reducing costs of treatments.

Chapter 2 - Background

The Span in PrefixSpan stands for **S**equential **pa**ttern mining. Sequential pattern mining has been around for a while and is a common research topic in data mining with many new variations on existing algorithms being introduced frequently [3,4,5,6]. It was first introduced by Agrawal and Srikant [7].

Database	Item 1	Item 2	Item 3	Item 4
Sequence 1	a	b	c	d
Sequence 2	a	c	d	e
Sequence 3	a	b	c	

Table 1. A sequence database.

2.1 PrefixSpan

PrefixSpan works by first finding all frequent sequences of length 1. A sequence is frequent if it satisfies the minsup threshold, which is normally set by the user. Minsup represents how many sequences in the database that a pattern has to occur in for it to be considered frequent. For a minsup of 2/3 when considering the sequences in Table 1, all patterns except for *e* would be frequent subsequences of length 1, since they appear in 2 sequences out of the total 3 in the database. Projected databases are then created for the frequent subsequences and mined recursively. A projected database contains only the sequences that the frequent subsequences appear in. This way, only the relevant sequences are mined in each iteration.

As *a* occurs in all sequences in this database it makes no difference from mining the original database, however for *b* only two thirds of the sequences in the database would have to be mined. The projected database for the 'prefix' *a* would consist of (*b, c, d*), (*c, d, e*), (*b, c*). The projected database is used to extend *a* with frequent items to create longer frequent subsequences. The length 2 frequent subsequences would be (*a,b*), (*a,c*), (*a,d*). A projected database is then created for each of these new 'prefixes'. It repeats these steps until the sequences can not be extended anymore and all frequent subsequences have been found. The problem with PrefixSpan is that it adds all of these frequent sequences to the result. In the example, both the subsequence *(a, b)* and the supersequence *(a, b, c)* would be added as both are frequent. However, as they occur with the same frequency the supersequence contain the same information as the subsequence. In PrefixSpan duplicates and subsequences of larger supersequences are added to the result, even though they are not statistically relevant. Closed sequential pattern mining, which is what CSpan does, does not mine these sequences and hence reduces the number of patterns produced while still having access to the same information.

2.2 CSpan

One of the algorithms mentioned by Uragaki, Keishiro, et al. as being able to improve the results was CSpan. The reason that it is more efficient is that it utilizes an occurrence check in order to only add unique sequences to the result, called closed sequences. A frequent sequence that does not occur in a supersequence and has the same support, is a closed sequential pattern. If the support of the supersequence is lower, the smaller sequence is still closed since it appears more often. Hence, only the longest frequent sequence is added and any subsequences of it are ignored, unless they occur more frequently than the supersequence. Since the same information can be extracted from the supersequences, all subsequences with the same support are efficiently the same as duplicates.

Instead of mining the complete set of sequential patterns, only closed sequential patterns are mined which reduces the number of patterns examined. The benefit to this is that fewer projected databases have to be generated, which can greatly reduce running time. With a low minsup, bigger databases and longer sequences, the effect of this is more apparent.

2.2.1 Occurrence Check

```
Occurrence Check
Input: FreqSequence: a frequent sequence,
       D: a projected database containing the sequences from the original database that
       FreqSequence occurs in,
       FreqItemSet: a list of all frequent items that occur after the frequent sequence
       FreqSequence in the database D
Output: determines whether or not to add the FreqSequence to the output set ClosedSequences
for each FreqItem in FreqItemSet do
       distance = 0
       temp = 0
       counter = 0
       if support(FreqSequence) == support(FreqItem) then
              for each Sequence in D do
                     counter++
                     if Sequence.contains(FreqSequence) and Sequence.contains(FreqItem) then
                            posA = Sequence.positionOf(FreqSequence[size-1])
                            posB = Sequence.positionOf(FreqItem)
                            temp = distance
                            distance = posB - posA
                            if distance != temp and temp > 0 then
                                   break
                            end if
                     end if
                     if counter >= D.size then
                            return
                     end if
              end for
       end if
end for
ClosedSequences.add(FreqSequence)
closure(FreqSequence)
return

Subroutine: closure(FreqSequence)
for Sequence in ClosedSequences do
       if Sequence.contains(FreqSequence) and support(Sequence) >= support(FreqSequence) do
              ClosedSequences.remove(FreqSequence);
       else if FreqSequence.contains(Sequence) and support(FreqSequence)>=support(Sequence) do
              ClosedSequences.remove(Sequence);
       end if
end for
```

Algorithm 1, Occurrence check.

---

*A sequential pattern **X** is not closed if a frequent item **y** exists such that (1) **y** appears in every sequence of **X**'s projected database and (2) the distance between **X** and **y** is identical in every sequence of **X**'s projected database.*

Lemma 1, Occurrence checking

The occurrence check used in this paper, Algorithm 1, was written based on Lemma 1 from [3], seen above. It takes a frequent sequence, its projected database and a set containing all frequent items that are found to occur after it in the sequences of the original database. For each item in the set, the distance of that item from the frequent sequence is checked for all sequences in the projected database. If the distance is the same in all sequences, the frequent sequence is not closed and should not be added, so the occurrence check can be interrupted. However, if the distance is different for any sequence, the inner loop can be interrupted since it means that the sequence is closed with regards to that item, and the next item is checked. If the sequence is found to be closed for all items, it will be added to the set of closed sequences. However, there is a risk that it is not closed with regards to the closed set. Therefore, before it can be added, another check is done to assure closure within the closed set.

2.3 Time and Medicine Handling

Each treatment item is associated with a *Time*, the date it was administered. In a sequence, all items are sorted according to their time. If items occur on the same date, they are sorted alphabetically. Each item is also associated with both a *Code* and a *Name*. *Code* is the medicinal code representing the efficacy of the medicine and *Name* is the name of the specific medicine. A medicinal classification table can be seen in Figure 1 below. Non-medicinal items do not have a code associated with them.

Medicinal classification table	
Code	Efficacy that corresponds to Code
112	Hypnotics and sedatives, antianxietics
114	Antipyretics, analgesics and antiinflammatory agents
223	Expectorants
225	Bronchodilating preparations
231	Antidiarrheals, intestinal regulators
235	Purgatives and clysters
239	Other agents affecting digestive organs
243	Thyroid and para-thyroid hormone preparations
331	Blood substitutes
441	Antihistaminic
449	Other antiallergic agents
613	Antibiotic preparations acting mainly on gram-positive and gram-negative bacteria
614	Antibiotic preparations acting mainly on gram-positive bacteria and mycoplasma

Figure 1, Medicinal classification table [1].

In T-PrefixSpan two different modes for mining were used. *Normal* and *Focus on Efficacy*. The difference is that with *Normal* the names of medicines are considered, meaning medicines with different names are treated as different items. With *Focus on Efficacy* only the code is considered, meaning different medicines with the same efficacy will be considered as the same item. Since several different medicines can have the same efficacy and new medicines appear constantly, it was deemed more relevant for the purpose of this paper to only consider the efficacy. If the name would be considered, medicines with different names but the same efficacy would be considered as different items, resulting in more but rarer sequences.

Chapter 3 - Method

T-CSpan and T-PrefixSpan was run on a dataset consisting of electronic medical records of Cryptorchidism Fusion Surgery treatments on several patients using *Focus on Efficacy*. The runtime and results of T-PrefixSpan was compared to those of T-CSpan. The goal was to be able to run the program quicker and to mine at a lower minsup than 0.02, which was reported in [1] to be the limit for T-PrefixSpan, and achieve a result that conforms better to the clinical pathway for CFS. Only the *Focus on Efficacy* mode was used and tests was run on the CFS dataset solely, as the algorithm has to accurately find the clinical pathway of a fixed flow before variants or new flows can be suggested, e.g. for TUR-Bt.

3.1 Test Machine

The machine is a Lenovo G50 (PC) laptop. The hardware specifications of the computer are the following:

- Processor: Intel Core i3 5005U, 2.0 GHz
- Memory: 4 GB RAM, DDR3L-1600 MHz
- Hard Disk Drive (HDD): 500 GB
- Operating System: Windows 10
- Java version: 1.8.0_121

3.2 Test Data

The dataset that will be used consists of EMRs of Cryptorchidism Fusion Surgery treatments on several patients between the years 2005-2015. The patients' identities are protected as anonymous patient IDs are used. These EMRs were obtained from an EMR system called *WATATUMI*, used by the Faculty of Medicine, University of Miyazaki Hospital.

3.3 Extracted Flow

The output is in the form of a number of closed frequent sequences, containing medicinal and non-medicinal items, sorted by the day they occurred. To compile all these sequences into one sequence that will represent the extracted flow, the output was run through a sorting algorithm.

The sorting algorithm summarizes each item into one row, giving the treatment info and a count of how many times it appears in the result.

Chapter 4 - Result

The results of the study will be presented in this section. T-PrefixSpan and T-CSpan was run on the Cryptorchidism Fusion Surgery dataset with the Focus on Efficacy option. The dataset used is the same as was used in [1].

4.1 Dataset

	Cryptorchidism Fusion Surgery
	Focusing on Efficacy
Sequences	271
Max length	549
Min length	10
Average length	27.39

Table 2. The dataset properties.

In Table 2 the properties of the dataset are described. There are a total of 271 sequences in the dataset. While the max length is 549, the average is relatively low at around 27 items.

4.2 Runtime Comparison

The runtime for T-PrefixSpan and T-CSpan is compared.

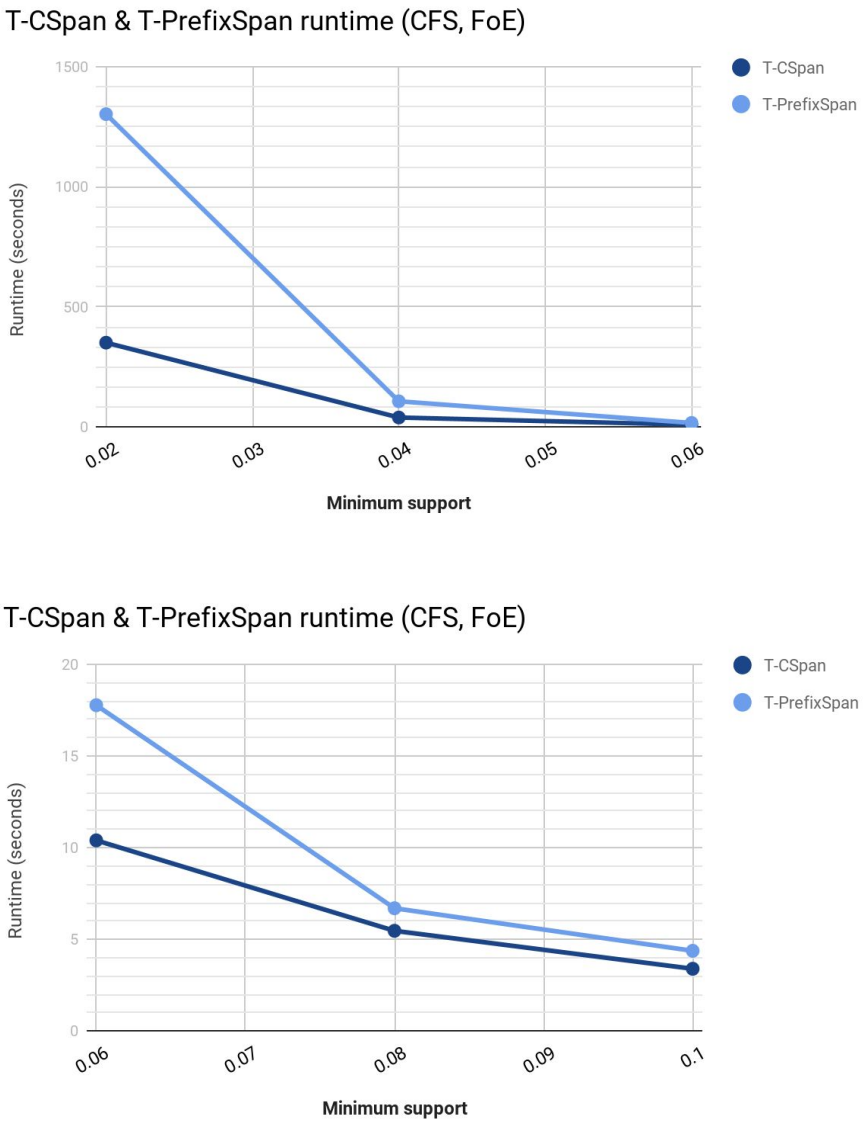

Figure 2.1 & 2.2, Runtime (seconds) for T-CSpan vs T-PrefixSpan using different minimum support (on the CFS dataset, focusing on efficacy).

minsup	T-CSpan	T-PrefixSpan
0.1	3.4 s	4.38 s
0.08	6.47 s	6.7 s
0.06	10.4 s	17.77 s
0.04	40.22 s	107.68 s (~2 mins)
0.02	351.62 s (~6 mins)	1302.62 s (~22 mins)
0.015	15642.03 s (~4h 20 mins)	-

Table 3. Runtime values (seconds).

The graphs in Figures 2.1 and 2.2 were split into two at 0.06 minsup in order to better visualize the difference in runtime for higher minimum supports, since the runtime at 0.02 differs greatly compared to other values. The runtimes can also be seen numerically in Table 3. As the minsup goes down, the runtime of T-CSpan becomes significantly lower than that of T-PrefixSpan. However, below 0.02 the runtime increased substantially. T-CSpan was able to mine at 0.015, although it took 4 hours and 20 minutes.

4.3 Flow Comparison

The extracted flow for CFS is compared, for both algorithms, to the fixed flow.

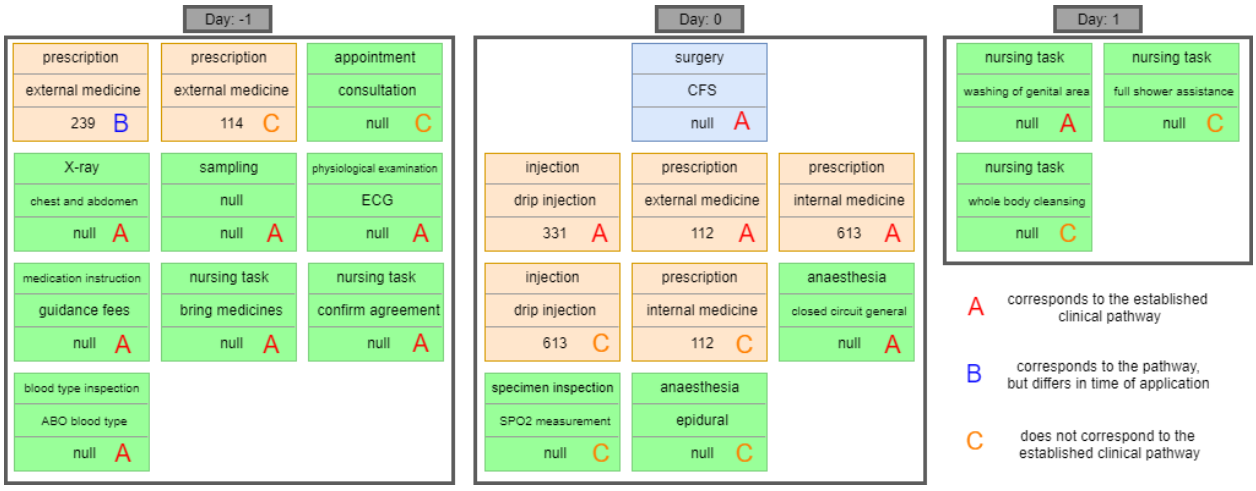

Figure 3.1, Extracted flow for CFS using T-PrefixSpan with minsup 0.02.

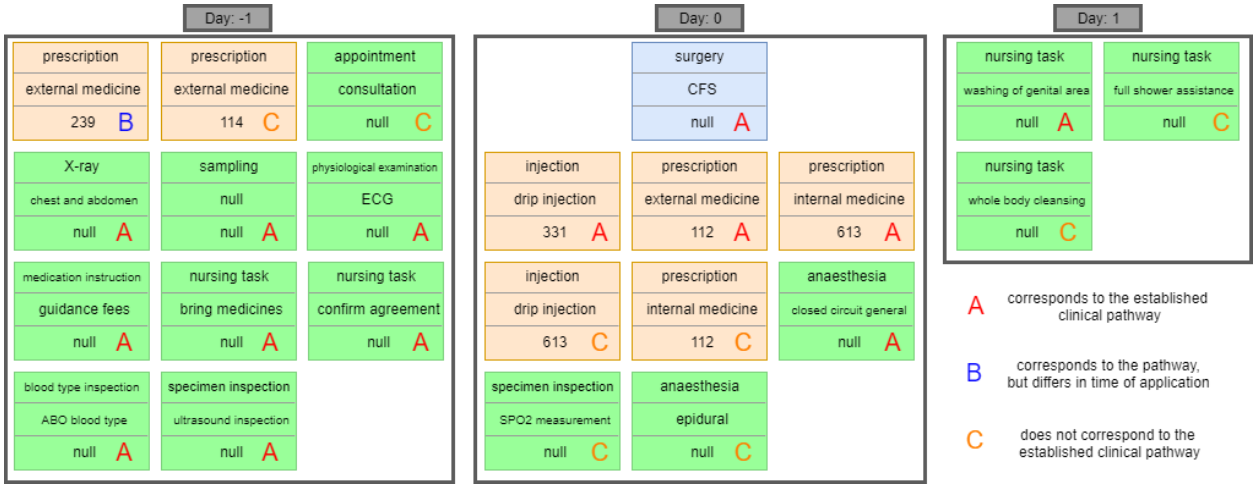

Figure 3.2, Extracted flow for CFS using T-CSpan with minsup 0.015.

Figure 3.1 and 3.2 only show one difference. T-CSpan managed to find one more item that corresponds to the established flow when mining at a lower minsup, 0.015. However, no additional treatments including medicines were found. As can be seen in Figure 3.3, it is mostly the treatments including medicines that are not found for both algorithms. The top item on day 0 is the surgery, prescriptions and injections are medicinal treatments and the rest are non-medicinal treatments. Items that correspond to the fixed flow and on the correct day are marked with an *A* and those that correspond to the flow, but on the incorrect day are marked with a *B*. Items that do not correspond at all are marked with a *C*.

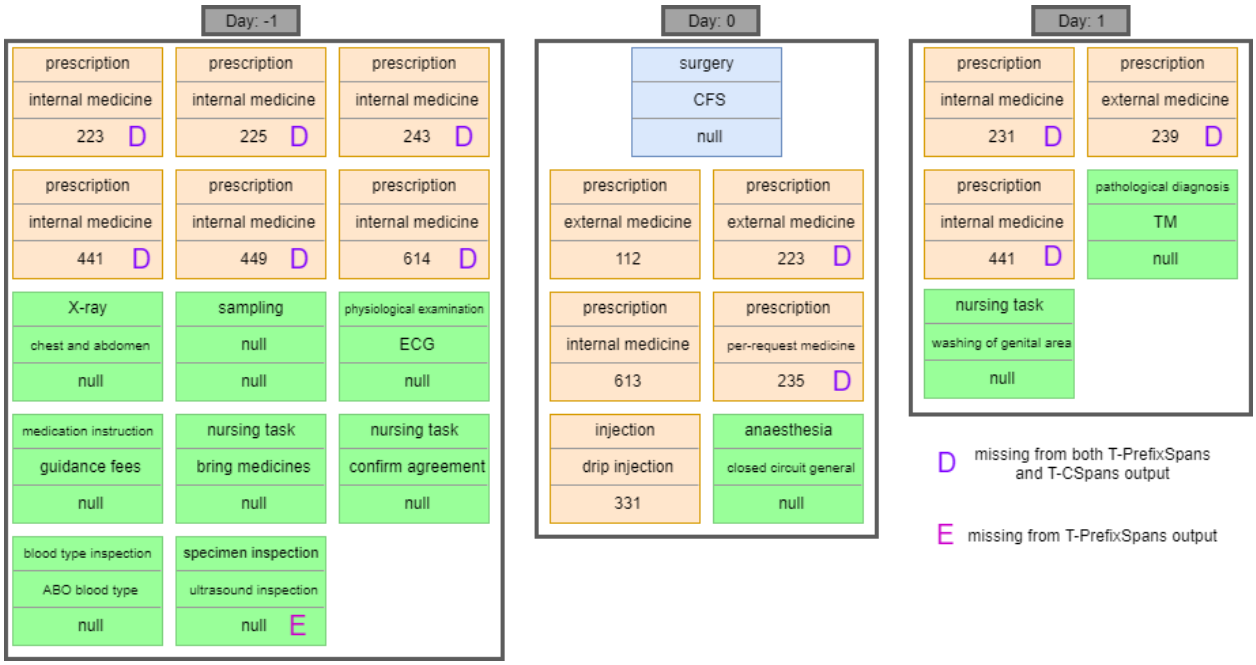

Figure 3.3, Existing flow of the clinical pathway for CFS.

When comparing the results to the fixed flow, Figure 3.3, it can be seen that the medicinal items still do not match well and the results are very similar. A *D* marks items that did not appear in neither T-PrefixSpan's nor T-CSpan's output. An *E* marks items that were missing from T-PrefixSpan's output, but not T-CSpan's.

Chapter 5 - Discussion

T-CSpan turned out to be faster than T-PrefixSpan and at low support it was quite a lot faster. However, I expected it to be even more efficient. The only information I could find about CSpan was in the paper that presented it, which merely consisted of a short explanation. There was no pseudocode for the occurrence check itself. Therefore I do not know how close my version is to the original and it might be possible to improve the runtime considerably.

Because of the improved runtime, it was possible to mine below a minsup of 0.02. For every item to be considered frequent, a minsup of 1/271 = 0.0036 would have to be used, assuming there are items that only occur once in the dataset. If every item occurs in at least 5 different sequences however, a minsup of 0.01845 would suffice.

The output was however, barely improved. One more item was found compared to T-PrefixSpan but it was not a medicinal one and it only occurred once in the result. It appears that no new medicinal items are considered frequent even when mining at 0.015, meaning they do not even appear in 5 different sequences as there are 271 in the dataset.

So I decided to look more closely at a specific medicinal item. Code 225 appears in 5 different sequences, meaning it should be considered frequent when the minsup is set to 0.015, which it does not. While running the program, I found that this item reached the point where it is used to extend frequent sequences, however, those extended sequences were never passed to the occurrence check. This leads me to the believe that the algorithm, for some reason, is having trouble extracting sequences containing medicinal items.

A second thought that I have about this is the fact that many medicinal items seem to appear very rarely in the dataset. Since these items are supposed to belong to a fixed flow, I expected them to appear in most of the cases. The item with code 225, for example, appear in less than 2% of the treatments. That means that items occurring on different days than specified by the fixed flow are more common than these medicinal items, in the dataset used.

Chapter 6 - Conclusion

Because of the improvements made to T-PrefixSpan, runtime was improved and it was possible to mine at a lower minsup, which was a part of the goal of this paper. However, this did not produce any interesting differences in the result contrary to the expected outcome.

It seems that the result will not be improved regardless of how low the minsup is set to as a medicinal item that appears in 5 sequences was not extracted even when the algorithm was run at the appropriate minsup. Also, the dataset might not be representative, as many medicinal items that are part of the fixed flow occur very rarely. The site where Uragaki et al. acquired their dataset is not accessible anymore and when this problem was discovered, it was too late to look for alternatives as EMRs are not readily available.

6.1 Future Work

Further work on this subject should, first and foremost, be to make sure medicinal items are extracted by the algorithm, and perhaps to get a new dataset. Possibly a bigger dataset, as it appears that the data might not be representative, since items that belong to the fixed flow should be considered frequent yet they barely appear in the dataset. Furthermore, this implementation of CSpan is most likely rather different from the ‘real’ CSpan and can probably be considerably improved.

References

1. Uragaki, Keishiro, et al. "Sequential pattern mining on electronic medical records with handling time intervals and the efficacy of medicines." Computers and Communication (ISCC), 2016 IEEE Symposium on. IEEE, 2016.
   Accessed at [http://ieeexplore.ieee.org/abstract/document/7543708/] on [13/7/2017]

2. Panella, M., S. Marchisio, and F. Di Stanislao. "Reducing clinical variations with clinical pathways: do pathways work?." International Journal for Quality in Health Care 15.6 (2003): 509-521.
   Accessed at
   [https://academic.oup.com/intqhc/article/15/6/509/1823636/Reducing-clinical-variations-with-clinical] on [13/7/2017]

3. Raju, V. Purushothama, and GP Saradhi Varma. "Mining closed sequential patterns in large sequence databases." International Journal of Database Management Systems 7.1 (2015): 29.
   Accessed at [http://airccse.org/journal/ijdms/papers/7115ijdms03.pdf] on [13/7/2017]

4. Yan Xifeng, Jiawei Han, and Ramin Afshar. "CloSpan: Mining: Closed sequential patterns in large datasets." Proceedings of the 2003 SIAM International Conference on Data Mining. Society for Industrial and Applied Mathematics, 2003.
   Accessed at [https://www.cs.ucsb.edu/~xyan/papers/cloSpan.pdf] on [13/7/2017]

5. Gomariz, Antonio, et al. "Clasp: An efficient algorithm for mining frequent closed sequences." Pacific-Asia Conference on Knowledge Discovery and Data Mining. Springer Berlin Heidelberg, 2013.
   Accessed at
   [https://pdfs.semanticscholar.org/9966/a1e8a67e1534a4b0377e23a303e43eeed13d.pdf] on [13/7/2017]

6. Han, Jiawei, et al. "Prefixspan: Mining sequential patterns efficiently by prefix-projected pattern growth." proceedings of the 17th international conference on data engineering. 2001.
   Accessed at [http://hanj.cs.illinois.edu/pdf/span01.pdf] on [13/7/2017]

7. Agrawal, Rakesh, and Ramakrishnan Srikant. "Mining sequential patterns." Data Engineering, 1995. Proceedings of the Eleventh International Conference on. IEEE, 1995.
Accessed at [https://pdfs.semanticscholar.org/d6a0/e0b04a020ac6422b98b8e63027a6178060fd.pdf] on [13/7/2017]

8. A. P. Wright, A. T. Wright, A. B. McCoy, D. F. Sittig, "The use of sequential pattern mining to predict next prescribed medications", Journal of Biomedical Informatics, vol. 53 pp. 73-80 2015.
Accessed at [http://www.sciencedirect.com/science/article/pii/S1532046414002007] on [13/7/2017]

9. Huang, Zhengxing, Xudong Lu, and Huilong Duan. "On mining clinical pathway patterns from medical behaviors." Artificial intelligence in medicine 56.1 (2012): 35-50.
Accessed at [http://www.sciencedirect.com/science/article/pii/S0933365712000656] on [13/7/2017]

10. Lin, Fu-ren, et al. "Mining time dependency patterns in clinical pathways." International Journal of Medical Informatics 62.1 (2001): 11-25.
Accessed at [http://www.sciencedirect.com/science/article/pii/S1386505601001265] on [13/7/2017]

11. Wakamiya, Shunji, and Kazunobu Yamauchi. "What are the standard functions of electronic clinical pathways?." International journal of medical informatics 78.8 (2009): 543-550.
Accessed at [http://www.sciencedirect.com/science/article/pii/S1386505609000409] on [13/7/2017]

12. Chen, Yen-Liang, Mei-Ching Chiang, and Ming-Tat Ko. "Discovering time-interval sequential patterns in sequence databases." Expert Systems with Applications 25.3 (2003): 343-354.
Accessed at [http://www.sciencedirect.com/science/article/pii/S0957417403000757] on [13/7/2017]