An Efficient Single Scan FP-Growth Algorithm for Mining COVID-19 Data in Canada

Connor Bean
Department of Computer Science
University of Manitoba
Winnipeg, MB, Canada
beanw@myumanitoba.ca

Claire Chen
Department of Computer Science
University of Manitoba
Winnipeg, MB, Canada
chenx338@myumanitoba.ca

Eddy He
Department of Computer Science
University of Manitoba
Winnipeg, MB, Canada
hej3456@myumanitoba.ca

James Froese
Department of Computer Science
University of Manitoba
Winnipeg, MB, Canada
froesej9@myumanitoba.ca

***Abstract*—The FP-Growth algorithm performs two transactional database scans in order to complete its mining of frequent k-itemsets. On a large dataset this second scan can significantly affect execution time. In this work, we propose to completely remove the second transaction database scan by modifying the way in which the dataset is represented, as well as how the FP-Tree is constructed. We will then put our proposed algorithm into practice by mining frequent demographic information with regards to the novel COVID-19 virus, using these results to motivate future discussion and potential mitigation strategies.**

Keywords—Data mining, FP-Growth, COVID-19

I. Introduction

The novel corona virus of 2019 dubbed COVID-19 has profoundly altered life on earth in 2020. In addition to the human sickness and death associated with a pandemic, widespread economic shutdowns have been implemented across the globe in an effort to limit the spread of the virus. As students, our lives have been directly impacted as international travel has been vastly reduced, classes moved to online delivery, and personal connections limited. This unique situation provides us with an opportunity to study a problem that is hugely relevant to our day-to-day lives, and where results may become immediately useful in the fight to control this disease.

As a result of these unique circumstances, we have decided to mine Canadian COVID-19 patient data. We pose to determine which demographics are most at risk of hospitalization and death, as well as if demographics most at risk of infection change depending on geographic location. Utilizing data provided by federal and provincial government, we intend to use a variation of the FP-Growth algorithm to determine the most at-risk demographics around the country. Patient data generally includes age, sex, health status, and exposure type which we will mine to determine frequent attributes and draw conclusions with regards to high-risk demographics. In addition, in comparing our results province-by-province we will determine if risk has any region-dependency.

Our contributions will be two-fold. First, we will examine high-risk demographics and how they change depending on province. While risk demographics associated with COVID-19 have been studied regionally in Canada already [2], due to the rapidly evolving nature of an on-going pandemic experiencing a second wave of infections, our project will provide further results on a larger set of cases over additional provinces. As British Columbia, Ontario, and Quebec contained the vast majority of the initial infections in spring of 2020, most research has been conducted on these specific provinces. At the time of writing (December 2020), Alberta, Saskatchewan, and Manitoba are all experiencing significant outbreaks which will add a much larger sample to our mining. Second, we will alter the classic FP-Growth algorithm to suit our particular needs efficiently. To achieve this, we propose removing the second database scan by representing the full database as a tree in memory, reusing available tree nodes where subsequent transactions have the same item prefix. Each path of the tree will then represent a transaction from the database to which the FP-Tree can be constructed. Removing a second pass from the algorithm will reduce computation time and allow for quick frequent pattern mining as the COVID-19 patient data available continues to grow throughout this pandemic.

II. Related Work

A. Covid-19 Demographics

The related and previous work considered in completing this project falls broadly under two categories. The first is the related work that characterizes how demographics are affected differently by COVID-19 and how that research educates the direction of our project. The second is how our selected data mining algorithm has been applied previously and how (and if) our selected changes have been implemented in similar use cases.

While the study of COVID-19 only began in 2020 for obvious reasons, there is significant research already completed that we may observe to direct our own goals for this project. One such preliminary study briefly compares the mortality rate between Chinese and Italian COVID-19 patients and reveals significant disparity in fatality rate between equivalent populations [1]. The fact that distributions of similar people in different geographic locations have different resilience to COVID-19 prompts inspection of this question on a finer regional level. While two countries like China and Italy may have considerable differences, it may be worth examining if such regional

changes in COVID-19 recovery rates exist regionally in Canada.

Examining how demographics react to COVID-19 across Canada has already been attempted in a paper entitled "Demographic Profile of COVID-19 Cases, Fatalities, Hospitalizations and Recoveries Across Canadian Provinces" [2]. This study compares the rates of hospitalization, fatality, and recovery of Canadians across the country by several different demographic characteristics. The authors break risk populations up by sex and age in a given region and found significant differences in the rate of hospitalization and death between provinces. However, this study was published in May of 2020 and is therefore extremely limited to a few select provinces that had substantial outbreaks at the time; namely British Columbia, Ontario, and Quebec. As of the time of writing (December 20, 2020), most provinces have substantial outbreaks ongoing in addition to significant daily hospitalizations and fatalities. With more up-to-date data and a larger sample size, it is likely that we can find additional similar results across the country.

A broader study done in Sweden examines finer demographics utilizing data about a patients age, net income, education, civil status, and country of birth [3]. The additional personal information allows for a finer examination of exactly what factors might influence a patient's chances of surviving COVID-19. So, while a simple examination of age and gender yields valuable information, a greater knowledge of an individual's health and socio-economic position will provide much more detailed information on what risk factors are most important to patient death.

Finally, while examining how COVID-19 affects demographics differently is not a novel idea, it may provide novel results given that new data is being received on a daily basis. Moving into our method of mining we can take the lessons learned in previous studies to drive the direction of our data mining efforts.

B. FP-Growth

As our primary goal with this study is to find frequent itemsets that might imply relationships in COVID-19 patient data, our next step is to determine which data mining algorithm best suites our effort to mine frequent itemsets from our data. Jeff Heaton in his paper *Comparing Dataset Characteristics that Favor the Apriori, Eclat or FP-Growth Frequent Itemset Mining Algorithms* examines the above three algorithms and their performance as it relates to frequent dataset mining. While he compares the specific frequency density a maximum transaction size to determine best algorithms, he broadly recommends the FP-Growth algorithm as an appropriate technique for frequent itemset mining [4].

Given our decision to use the FP-Growth algorithm to mine our data for frequent itemsets in COVID-19 patient data, we decided to try and optimize the algorithm to reduce runtime as much as possible. Examining ways in which others have tried to optimize FP-Growth provides a starting point for a new modification.

In a study on oceanographic data, the authors use a parallel method to attempt to optimize their use of FP-Growth to handle large datasets [5]. While this model does result in a faster runtime, the parallelization technique, number of threads used, size of dataset, and itemset characteristics make the ideal parameters to mine with complex. Determining the best optimization using this technique therefore requires significant study and preparation to achieve maximum efficiency. Similarly, in a paper by Chen and colleagues [8], the regular FP-Growth algorithm is sped up through parallelization of data mining. What's different however, is that instead of duplicating the database in order for multiple machines to mine in parallel, the author's algorithm omits the initial building of the FP-tree and instead builds a frequent list of common itemset prefixes. This list is then provided to other nodes for processing. This method results in a reduced FP-Tree that will not overwhelm memory and allow for parallel computation to occur.

In another related paper, *an optimized algorithm for association rule mining using FP tree* by Narvekar and Syed, they offer two proposed optimizations to the FP-Growth algorithm [9]. Firstly, they offer an improved FP-Tree construction method that represents the database as an in-memory tree structure [9]. This is done is a very similar fashion to our proposed algorithm, scanning the transactional database, representing each transaction as a path in what they call the D-Tree [9]. However, they take the optimization one step further by using the D-Tree to create a simplified FP-Tree that allows for more efficient frequent itemset mining [9]. The idea is that they introduce what they call a node table to store spare items in, allowing the FP-Tree to only contain one node per unique item from the database [9]. They then perform a modified mining technique that allows space to be saved by reducing the number of conditional FP-Tree's that are generated in a more authentic FP-Growth algorithm [9]. Comparing the practicality of this algorithm to our proposed idea, this technique would allow FP-Growth to mine larger datasets more efficiently as space is saved on the generation of the FP-Tree itself as well as the mining process.

III. MINING

A. Improved FP-Growth

The contribution we propose involves modifying the original FP-Growth algorithm in order to reduce the number of transactional database (TDB) scans required and thus, create a more efficient mining process. However, this does come at the expense of using more memory; a time-space trade-off.

The original FP-Growth algorithm consists of four main steps: completing an initial scan of the TDB in order to find the frequent 1-itemsets, sorting the frequent 1-itemsets according to a set of criteria/heuristic, completing a second scan of the TDB to construct an FP-Tree, and passing this tree off to the actual FP-Growth mining algorithm to find the

frequent k-itemsets [7]. Our proposal focuses on removing the second TDB scan, thus reducing the transactions scanned in half.

To achieve this, we first focus on the TDB representation before passing it to the FP-Growth algorithm for mining. This will consist of three crucial steps. First, we perform our one and only TDB scan, scanning each transaction and adding each item to a tree structure as a node represented as <*item*, *occurrence_count*>. We will call this tree structure REP-Tree, as it will represent the entire TDB as an in-memory data structure. Each path of the REP-Tree will then represent one transaction from the TDB. To save space, we will re-use tree nodes that have the same prefix as the next transaction from the TDB, incrementing the *occurrence_count* of said node.

Secondly, our algorithm will traverse the REP-Tree, and create a table consisting of <*item*, *occurrence_count*> pairs. In this step, any item whose occurrence count does not meet the user specified minimum support threshold with be pruned from the table. The table will then be sorted according to a heuristic. The heuristic we will be using in our implementation and following example will be based on a frequency descending order. Resolving occurrence count ties based on alphabetical order of the item.

Finally, our algorithm will then convert our REP-Tree into an FP-Tree in preparation for mining. To execute this, we simply need to extract each path from the REP-Tree. Since each path represents a transaction from the TDB, we can remove items that are not present in our header table and add the resulting transaction to an FP-Tree according to the standard FP-Growth algorithm. Once the FP-Tree is constructed it can be passed to FP-Growth for mining. At this point the REP-Tree is no longer required and can be removed from memory.

To illustrate the above description, we will perform a step-by-step example using the data in Table 1 below.

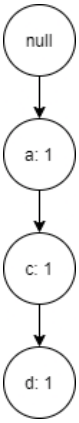
Table 1. Example Transactional Database

Transaction ID	Transaction
t1	a, c, d
t2	b, c, e
t3	a, b, c, e
t4	b, e

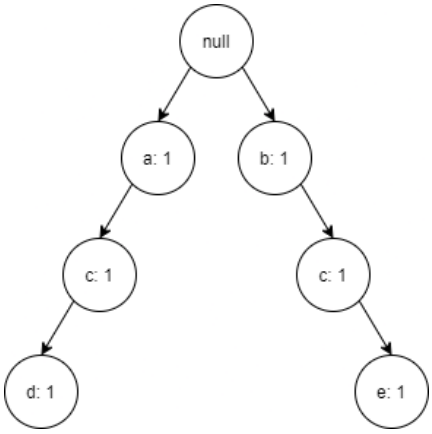
minsup = 2

Step 1: Scan the TDB and represent each transaction in our REP-Tree.

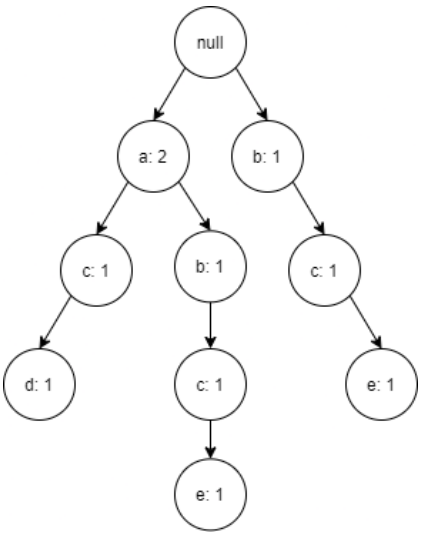
Scan t1



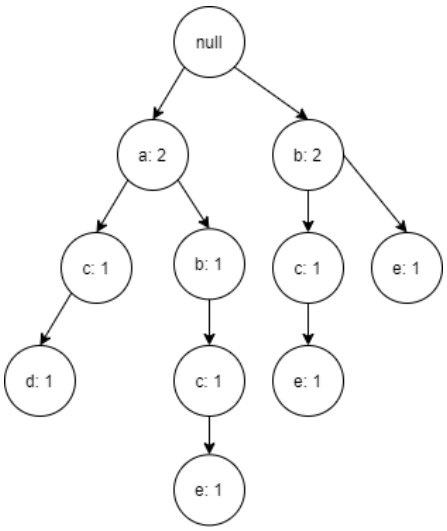
Scan t2



Scan t3



Scan t4



Following the first step, the result is a REP-Tree. Each path of the tree now represents a transaction from the original TDB.

Step 2: Scan the REP-Tree, sum the item occurrence count, and re-order based on our heuristic.

Item	Occurrence Count
a	2
b	3
c	3
~~d~~	~~1~~
e	3

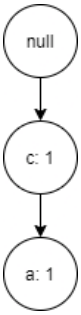
Re-order in frequency descending order

Item	Occurrence Count
b	3
c	3
e	3
a	2

Step 3: For each path in the REP-Tree, sort in header table order, remove any item that does not appear in the header table, and construct an FP-Tree.

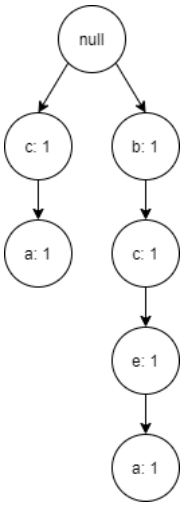
Path 1 = a – c – d becomes c – a

FP-Tree



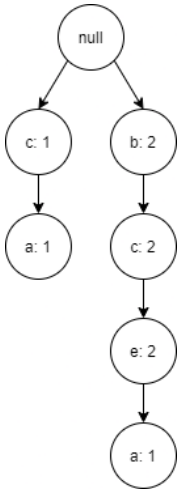
Path 2 = a – b – c – e becomes b – c – e – a

FP-Tree



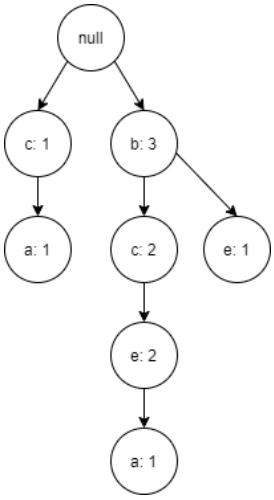
Path 3 = b – c – e becomes b – c – e

FP-Tree #3



Path 4 = b – e becomes b – e

FP-Tree #4



Upon completing this construction process, we can pass the resulting FP-Tree to the FP-Growth algorithm for mining the remaining frequent k-itemsets. This requires only one scan of the transactional database at the cost of using more memory. However, in a practical scenario, if FP-Growth is the algorithm being used for mining, the TDB must fit into memory to be represented as an FP-Tree. If this is possible, representation as a REP-Tree should also be feasible.

Below we illustrate, in pseudocode, the three main modifications to FP-Growth required to apply our REP-Tree.

Code 1. Pseudocode for main mining:

```
OPEN database
for each transaction:
   ADD transaction to REP-Tree

PROVIDE REP-Tree to FP-Growth mining algorithm
```

Code 2. Pseudocode for adding a transaction to the REP-Tree:

```
ASSIGN the Current Node as the Root Node

for each item in the transaction:
   if the item is in the roots children:
      INCREMENT the nodes count
      ASSIGN this child node as the Next Node
   else:
      CREATE a new node as <item, 1>
      ASSIGN the new node to be a child of the current n
ode
      ASSIGN this new node as the Next Node

   ASSIGN the Current Node as the Next Node
```

Code 3. Pseudocode for getting support values of items in a REP-Tree:

```
CREATE a dictionary of the form <item, count>
CREATE a queue
ADD the root node to the queue

while the queue is not empty:
   dequeue node from the queue

   if the node contains an item:
      ADD an entry to the dictionary with the nodes item
      and increment the count by its occurrence count

      ADD each child of the node to the queue

RETURN the generated dictionary
```

B. Mining Procedure

Our proposed FP Growth Algorithm is used to mine the chosen Canadian COVID-19 case details database after reducing some unnecessary information like *ObjectId*, *latitude*, *longitude*, etc. The mining for most vulnerable groups is done in two parts; one is to mine and compare with results from every month starting on the 1st, and the other is to compare with results mined for each province.

Mining frequent itemsets over different periods of time is performed on the entire database and sub databases. The sub databases are generated based on transactions whose *case_status* is Deceased, and *exposure* is Close Contact and Travel-Related. This is done in order to mine different representations of most vulnerable group formed by age and gender. The age and gender group with highest support is chosen to be the most vulnerable group. A minimum support threshold of 3% is chosen to mine the entire database as this will ensure each month have frequent 2-itemsets where the most frequent age and gender group only takes up 3.28% for May. Furthermore, a minimum support of 5% is used to mine the sub databases. The results for January and February are excluded as COVID-19 just began around that time in Canada. This makes the number of cases, 9 for January and 58 for February, not large enough to contribute to our discussion in a meaningful way.

Mining for different provinces is performed by taking the *age_group*, *gender*, and *province* columns from the database. We set the minimum support threshold to 10 as this gives us a more consistent result among other support values experimented with. When mining the most vulnerable age group within each province, we find all frequent 2-itemsets and their occurrence count containing both age-group and province. The age group with the largest count is defined as the most vulnerable group for each province. When mining the most vulnerable age group, including gender within each province, we find all frequent 3-itemsets and their occurrence count containing age-group, province and gender. We define the most vulnerable group as the age group with the largest occurrence count recorded, separating

this for male and female patients. For the most vulnerable group within each gender, the following 3 provinces/territories: Prince Edward Island, New Brunswick, and Newfoundland and Labrador do not have enough cases or only have one gender recorded. These transactions are therefore excluded from the results to avoid any biased conclusion.

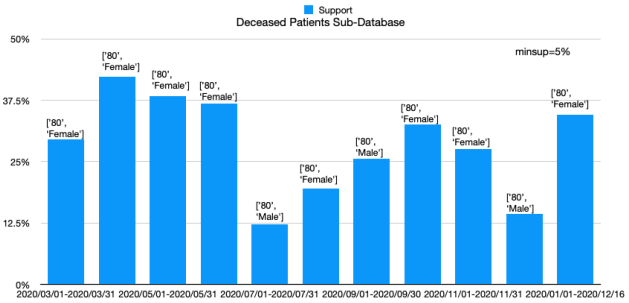
IV. Results

The dataset used with our modified FP-Growth algorithm has the following relevant attributes: *gender*, *case_status*, *exposure type*, *age-group*, and *health_region*. Given these attributes we will discuss the results obtained from our dataset.

A. Mortality Risk

To begin, we are interested in risk groups for mortality over time. Discovering if the highest risk groups for death have changed over the course of the pandemic could potentially help to educate our response as the pandemic progresses.

Chart 1. Highest Risk of Death Patients



As illustrated in Chart 1 above, the patients with highest risk of death due to COVID-19 consistently remains patients in their 80s, both female and male. While these results are not unexpected, given the higher prevalence of co-morbidities in older patients, continuing to monitor these numbers as new treatments are utilized might inform changes in infected demographics.

B. Exposure Type

Secondly, examining the demographics most frequently associated with various forms of infection leads us to the following results, see charts 2 and 3.

Chart 2. Close Contact Exposure by Demographic

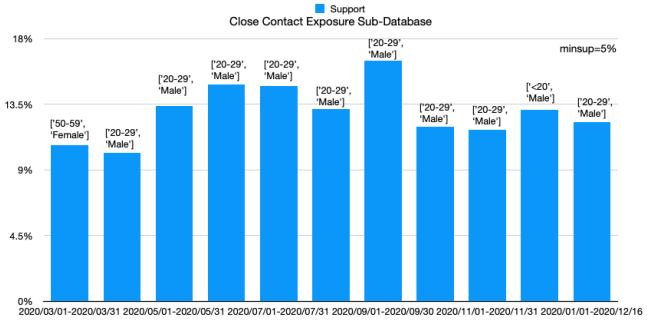
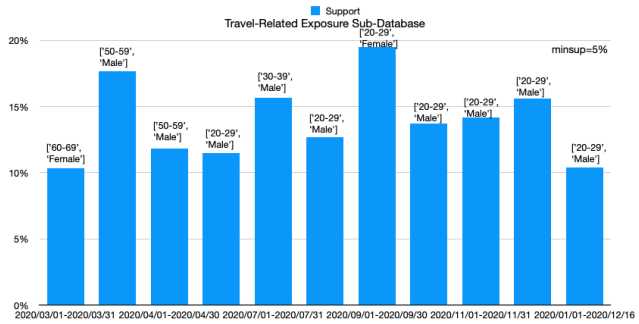


Chart 3. Travel Exposure by Demographic



A look at the above charts shows the demographic most likely to contract COVID-19 through either a close contact or travel. The results of particular interest are the shift in at-risk demographic over time seen in Chart 3. During the months of March through May we see older adults being the primary vector for COVID-19 infections through travel. After May however, young adults of both genders consistently account for the most frequent infections through travel.

A simple explanation for this abrupt shift could be the fact that young adults are in a low-risk demographic for serious COVID-19 complications. Since there is relatively little danger of serious personal danger, young adults therefore do not feel the need to limit travel as much as those in older age-groups.

Similarly, a look at Chart 2 shows that aside from the beginning of the pandemic, the demographic most frequently infected through close contact is young males ages 20-29 years old. A single data point in March 2020 shows females ages 50-59 most frequently at risk; this is the only outlier to this trend. While this single data point is not enough to illustrate a total shift in frequent close contact infection demographics, presumably due to imposed COVID-19 restrictions, it does support the results discussed in Chart 3. Namely, that government-imposed restrictions since the beginning of the pandemic have been more effective with older adults and less effective at reducing spread through multiple infection vectors with young adults, particularly males.

C. Age Demographics

Chart 4. Cases by Age

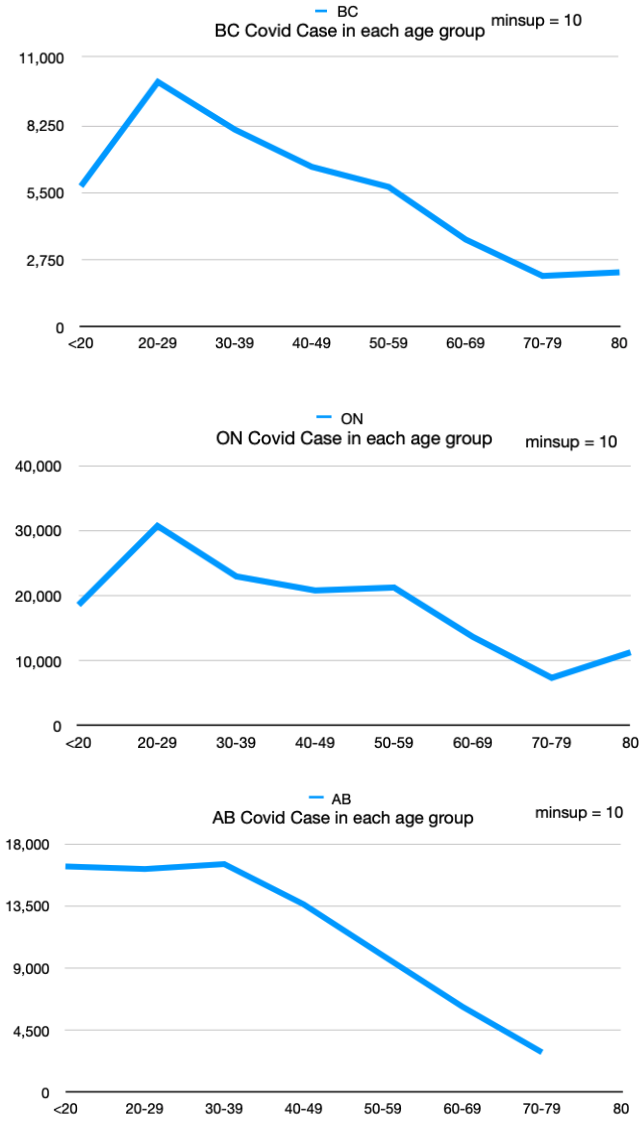
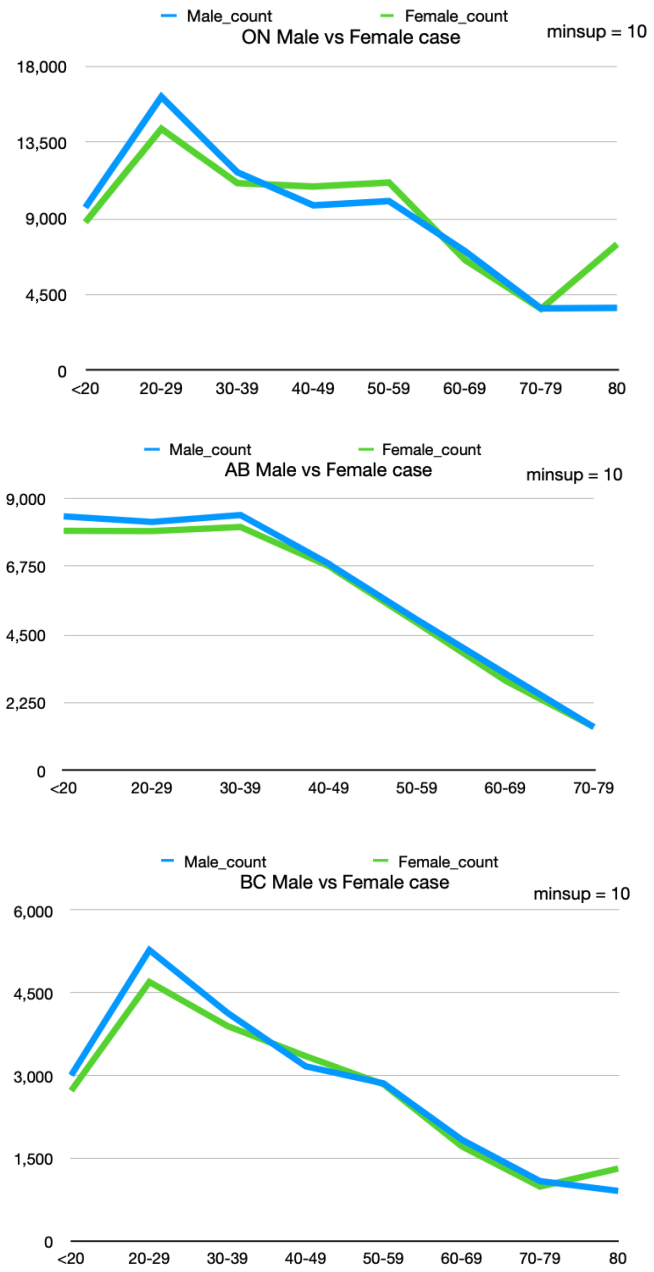


Chart 4 shows the cases seen in three provinces by age-group. Alberta, Ontario, and British Columbia were displayed as they were the only provinces in the dataset with case numbers in the thousands. While each province shows slight variation in the precise ratio of cases per-age group, the overall trend remains consistent. Overall, younger people represent the most COVID-19 cases in each province. The total number of cases trends downward as age group increases. This affirms the results found with regards to frequent demographics infected via travel and close contact. Examining Chart 5 shows an extremely similar distribution of cases by age demographic between sexes. While males are, generally, slightly more likely to be infected across age ranges, the difference is minimal and as a result, the conclusions we draw with regards to at-risk demographics for contracting COVID-19 has more to do with age than gender.

Chart 5. Cases by Age and Gender



D. Modified Algorithm Comparison

Comparing the original FP-Growth algorithm to our new proposed FP-Growth algorithm, we noticed some interesting performance differences. We compared these two algorithms using a dataset of 1, 2, 7, and 13 unique items on a database size of 10000, 50000, 100000, 500000, and 1000000. We further experimented with a minimum support threshold of 3, and a much larger minimum support threshold of 10%.

Table 2 and table 3 below illustrate the time (in seconds) that it took to execute each algorithm with a minimum support threshold equal to 10% and 3, respectively. These tables are broken down into time taken to complete the first database scan, the time taken to complete the frequent

itemset mining, and the total execution time for each algorithm.

Table 2. Time Comparison of FP Growth and Our New FP Growth for minsup = 10%

Minsup=10%

		FP Growth			New FP Growth		
Number of Items	Number of Transactions (x10000)	Time for 1st DB scan(s)	Time for mining with 2nd DB scan(s)	Total time(s)	Time for 1st DB scan(s)	Time for mining without 2nd DB scan(s)	Total time(s)
1	1	0.006849	0.026743	0.033592	3.778615	8.794226	12.572841
	5	0.029959	0.108287	0.138246	3.495327	9.578382	13.073709
	10	0.047384	0.235302	0.282686	3.867308	8.961116	12.828424
	50	0.494167	1.292675	1.786842	3.321386	8.739933	12.061319
	100	1.332434	2.599547	3.931981	2.952844	9.338865	12.291709
2	1	0.008389	0.038381	0.04677	3.905153	9.089948	12.995101
	5	0.036291	0.148517	0.184808	3.649052	8.829931	12.478983
	10	0.053233	0.304074	0.357307	3.723290	9.008787	12.732077
	50	0.872262	1.609933	2.482195	3.635585	9.357383	12.992968
	100	1.175910	3.290898	4.466808	3.290351	8.177903	11.468254
7	1	0.014880	0.082485	0.097365	3.846366	8.773406	12.619772
	5	0.068728	0.342797	0.411525	3.564453	8.943261	12.507714
	10	0.416831	0.670876	1.087707	3.896373	8.548642	12.445015
	50	0.932749	3.829312	4.762061	3.336239	9.305300	12.641539
	100	1.603243	6.466473	8.069716	3.119535	8.280690	11.400225
13	1	0.023138	0.211170	0.234308	3.716773	8.868879	12.585652
	5	0.108843	0.615759	0.724602	3.574015	8.852497	12.426512
	10	0.202382	1.102556	1.304938	3.982898	8.465991	12.448889
	50	1.398579	5.630854	7.029433	3.437229	9.453105	12.890334
	100	2.768687	10.518168	13.286855	3.349542	8.856066	12.205608

Table 3. Time Comparison of FP Growth and Our New FP Growth for minsup = 3

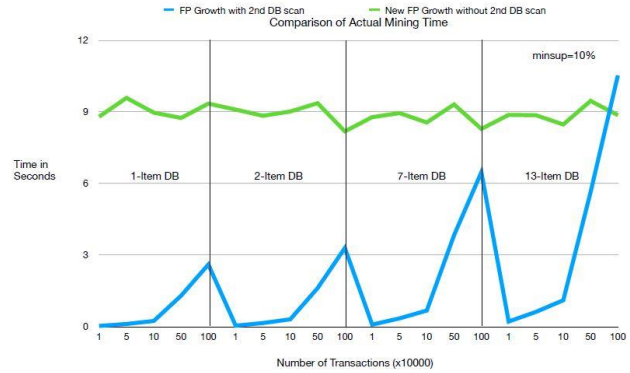
Minsup=3

		FP Growth			New FP Growth		
Number of Items	Number of Transactions (x10000)	Time for 1st DB scan(s)	Time for mining with 2nd DB scan(s)	Total time(s)	Time for 1st DB scan(s)	Time for mining without 2nd DB scan(s)	Total time(s)
1	1	0.005284	0.030063	0.035347	4.642311	38.280339	42.92265
	5	0.759468	1.654519	2.413987	3.805805	39.004925	42.81073
	10	0.047271	0.931621	0.978892	3.935213	39.181218	43.116431
	50	1.150560	1.578380	2.728940	3.749621	38.104467	41.854088
	100	1.355186	3.318962	4.674148	3.515855	38.472821	41.988676
2	1	0.007266	0.045599	0.052865	4.323616	39.899828	44.223444
	5	0.031211	0.815281	0.846492	3.901515	39.098532	43.000047
	10	0.060578	0.328245	0.388823	4.045049	38.799143	42.844192
	50	0.929988	2.465082	3.39507	3.799792	39.360025	43.159817
	100	0.851098	3.947662	4.79876	3.299029	37.052258	40.351287
7	1	0.016745	2.107256	2.124001	4.589452	39.974353	44.563805
	5	0.070759	5.534523	5.605282	3.596223	38.585937	42.18216
	10	0.125754	6.351060	6.476814	4.089940	38.418112	42.508052
	50	1.178134	35.005443	36.183577	3.876174	38.577713	42.453887
	100	1.810166	40.649590	42.459756	3.569203	38.471611	42.040814
13	1	0.023739	30.869394	30.893133	3.844074	36.131663	39.975737
	5	0.109407	1495.664736	1495.77414	3.325896	37.135383	40.461279
	10	0.229147	1558.931990	1559.16113	3.709270	38.539893	42.249163
	50	1.577674	1803.046532	1804.62420	4.324835	38.067326	42.392161
	100	3.046375	3978.968727	3982.01510	6.200273	36.726959	42.927232

The time taken to complete the first database scan in the original FP-Growth algorithm represents the time required to scan each transaction in the database and construct a simple list of transactions. With respect to our new FP-Growth algorithm, this time represents the time taken to scan each transaction in the database and construct our REP-Tree (a tree representing each transaction in the database). Furthermore, the second column of data represents the time taken to complete the second database scan, determine the frequent 1-itemsets, and complete the mining process to determine the remaining frequent k-itemsets with respect to the original FP-Growth algorithm. This column represents the time taken to iterate over the REP-Tree, determine the frequent 1-itemsets, and complete the mining process to determine the remaining frequent k-itemsets for our new proposed FP-Growth algorithm.

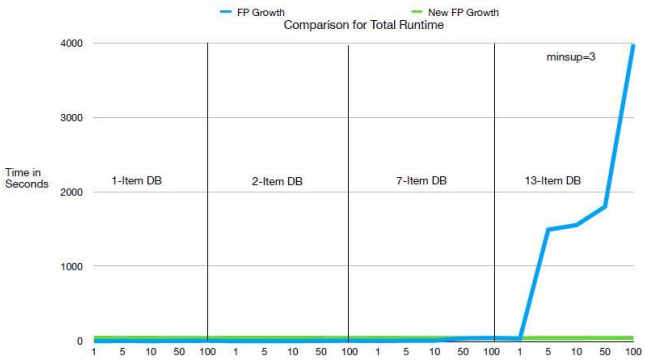
Looking at the results for each of these algorithms, we can observe that the original FP-Growth algorithm outperforms our new FP-Growth algorithm with large minimum support thresholds (see table 2). This is largely due to the fact that there is a large overhead to construct and iterate over the REP-Tree when a lot of items will inevitably be pruned due to not meeting the minimum support threshold requirement. It should be noted that our new FP-Growth algorithm performs all mining in a relatively constant time due to the tree traversal process. However, we see the opposite result when using a low minimum support threshold (see table 3). In this scenario, the original FP-Growth algorithm must now iterate over almost all the possible itemsets, reading them off disk to construct the FP-Tree. While FP-Growth still outperforms our new FP-Growth algorithm on datasets with few unique items, our proposed algorithm significantly outperforms the original FP-Growth algorithm when we begin mining on large datasets with many unique items.

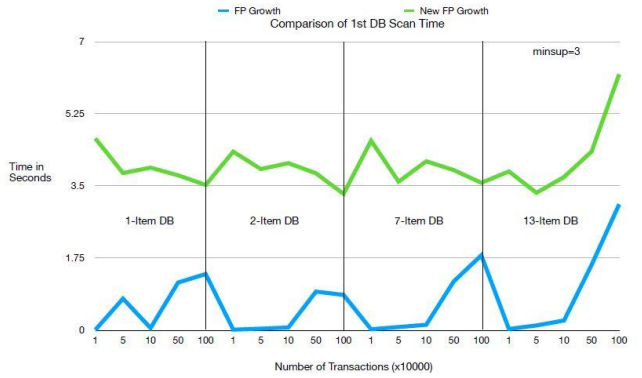
Chart 6. Time Comparison for minsup = 10%

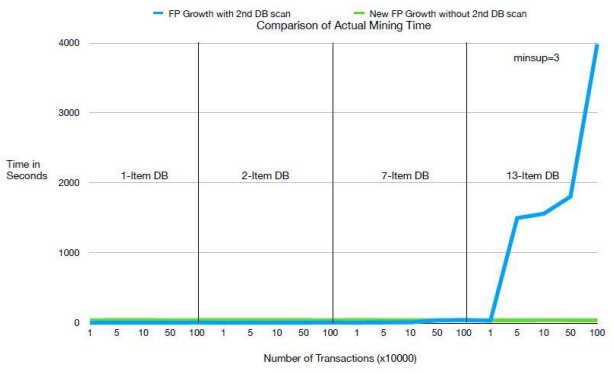


In terms of the practicality of our proposed changes to FP-Growth, these results are promising. Since most interesting mining will be done using large datasets with many unique itemsets, our new algorithm could pose for more efficient frequent pattern mining in certain scenarios.

Chart 7. Time Comparison for minsup = 3







V. Conclusions

While the results we have found from mining this particular dataset are nothing groundbreaking, there are a few useful conclusions we can draw.

First, the population most at risk of death due to complications related to COVID-19 is consistently those 80 years of age and older. This result might seem obvious but drawing this conclusion through our data-mining process lends credible evidence to what might be seen as common sense.

Second, young people, particularly males between the ages of 20-29 are the most frequent demographic to contract the virus through a close contact or travel. In addition to being the most frequently infected group due to these two specific infection vectors, they also present the most cases per any individual demographic. According to Statistics Canada [6], Canadians aged 20-29 make up 13.4% of total population. Comparing to the 30-39 demographic (13.9%), 40-49 demographic (12.8%), and <20 demographic (21.3%), we can see that young adults 20-29 are not the largest demographic in the country, and therefore show a higher-than-average incident of COVID-19 infection.

The fact that this age group and gender appears to consistently be infected at a greater rate than the general population could imply a few different things. One such option is that existing messaging to this demographic is ineffective and additional education and restriction might be required to lower infection rates. Without additional data on each COVID-19 case, it's hard to draw strong conclusions and is a potential area for future work.

With regards to our improvement to the FP-Growth algorithm, we can clearly see the benefit of a single database pass on algorithm runtime. As mentioned above, this time improvement comes with a space penalty. While not every data mining situation is appropriate for use of our modified algorithm, the detailed analysis we provide of results provides a good idea of applications where it may be appropriate.

VI. Limitations

The primary limitation in our work to determine useful information on various COVID-19 demographics is a result of limited patient information. Even the information included in the database is largely incomplete. For example, our dataset includes an attribute to describe exposure type. A large portion of patients do not have an exposure type reported and as a result, leaves our data incomplete. For us to determine truly novel insights, more attributes per patient are required. For example, including the profession and employment status of each person, their housing situation (i.e., house, apartment, rooming house, personal care home, etc.), pre-existing medical conditions, etc. would provide a much more comprehensive look at the personal situations that surround each COVID-19 case.

Another limitation is the limited quantity of patient data available from a central source. As cases continue to rise throughout the winter at a near exponential rate throughout Canada, more data will become available and will provide either additional insights into COVID-19 infection demographics or additional evidence to support existing points.

VII. Future Work

Future work will primarily involve the use of more data and additional patient details. Accessing more accurate data from around the country that is regularly updated will provide more insights into how COVID-19 affects demographics differently and can help to educate our response to it. This response could come in the form of generating a tiered vaccination plan for Canadians who are most at risk from Covid-19.

Acknowledgment

This work was supported by the compiled COVID-19 case details (Canada) dataset, provided by Esri Canada [11]. Furthermore, the frequent pattern mining was support in part by enaeseth and their FP-Growth python implementation [10]. This repository was used to mine frequent patterns using the standard FP-Growth algorithm, and further modified to support our proof of concept for an improved FP-Growth algorithm.

References

[1] Lippi, G., Mattiuzzi, C., Sanchis-Gomar, F. and Henry, B.M. (2020), Clinical and demographic characteristics of patients dying from COVID-19 in Italy vs China. J Med Virol, 92: 1759-1760. https://doi.org/10.1002/jmv.25860

[2] Simona Bignami-Van Assche & Ari Van Assche, 2020. "Demographic Profile of COVID-19 Cases, Fatalities, Hospitalizations and Recoveries Across Canadian Provinces," CIRANO Working Papers2020s-31, CIRANO.

[3] Drefahl, S., Wallace, M., Mussino, E. *et al.* A population-based cohort study of socio-demographic risk factors for COVID-19 deaths in Sweden. *Nat Commun* 11, 5097 (2020). https://doi.org/10.1038/s41467-020-18926-3

[4] J. Heaton, "Comparing dataset characteristics that favor the Apriori, Eclat or FP-Growth frequent itemset mining algorithms," *SoutheastCon 2016*, Norfolk, VA, 2016, pp. 1-7, doi: 10.1109/SECON.2016.7506659.

[5] Jiang, Y., Zhao, M., Hu, C. *et al.* A parallel FP-growth algorithm on World Ocean Atlas data with multi-core CPU. *J Supercomput* 75, 732–745 (2019). https://doi-org.uml.idm.oclc.org/10.1007/s11227-018-2297-6

[6] Statistics Canada. Table 17-10-0005-01 Population estimates on July 1st, by age and sex. https://doi.org/10.25318/1710000501-eng

[7] Jia K., Liu H. (2017) An Improved FP-Growth Algorithm Based on SOM Partition. In: Zou B., Li M., Wang H., Song X., Xie W., Lu Z. (eds) Data Science. ICPCSEE 2017. Communications in Computer and Information Science, vol 727. Springer, Singapore. https://doi-org.uml.idm.oclc.org/10.1007/978-981-10-6385-5_15

[8] M. Chen, X. Gao and H. Li, "An efficient parallel FP-Growth algorithm," *2009 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery*, Zhangijajie, 2009, pp. 283-286, doi: 10.1109/CYBERC.2009.5342148.

[9] Meera Narvekar, Shafaque Fatma Syed, "An Optimized Algorithm for Association Rule Mining Using FP Tree", Procedia Computer Science, Volume 45, 2015, Pages 101-110, ISSN 1877-0509. https://doi.org/10.1016/j.procs.2015.03.097

[10] enaeseth (2016) python-fp-growth [Source code]. https://github.com/enaeseth/python-fp-growth

[11] *Compiled COVID-19 Case Details (Canada)*, esri Canada, 2020. [Dataset]. Available: https://resources-covid19canada.hub.arcgis.com/datasets/compiled-covid-19-case-details-canada?page=5