데이터 구조

2차 프로젝트

학　　과: 컴퓨터정보공학부

담당교수: 이기훈교수님

학　　번: 2017202087

성　　명: 홍 세 정

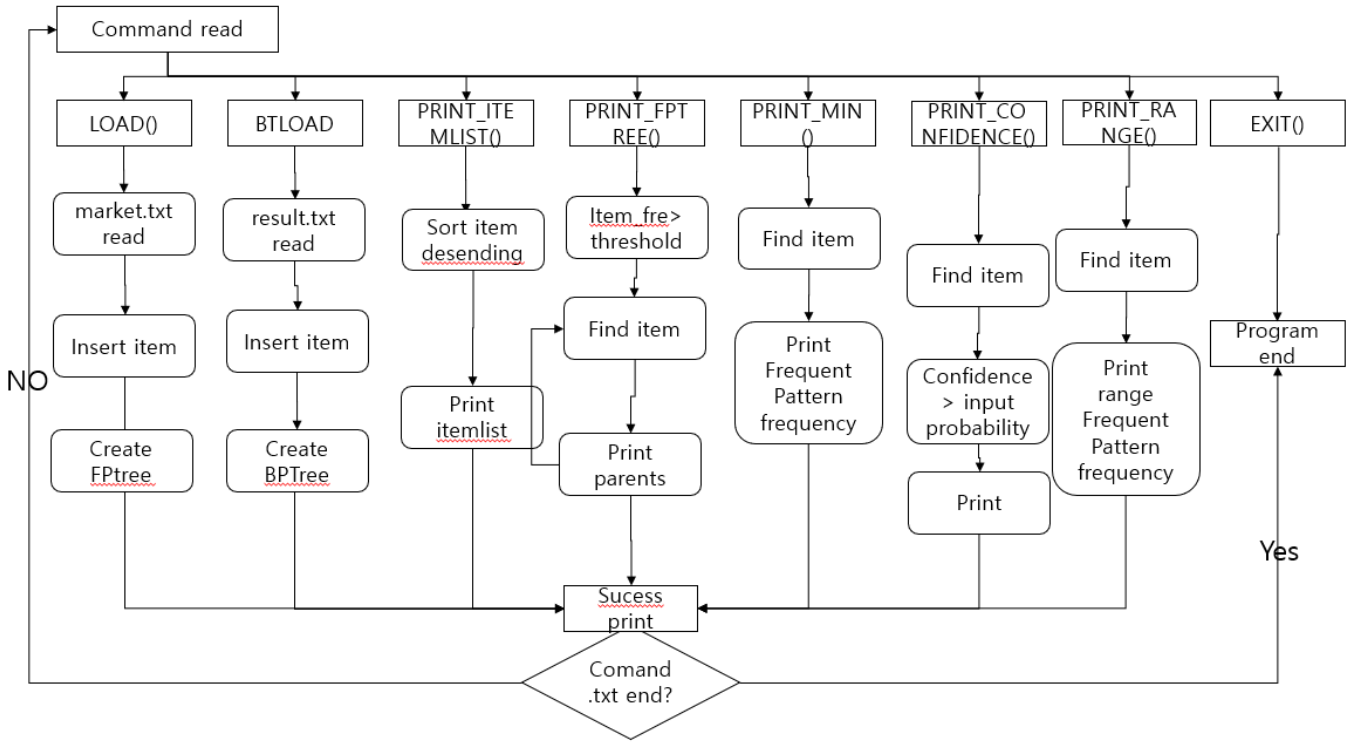
1. Introduction

이번 2차 프로젝트에서는 FP-Growth와 B+-Tree를 이용하여 상품 추천 프로그램을 구현하는 것이다.

Market.txt에 저장된 데이터를 이용하여 FP-Growth를 구현할 수 있다.

Result.txt에 저장된 데이터를 이용하여 B+-Tree를 구현할 수 있다.

2. Flowchart



이번 프로젝트의 전체적인 flowchart는 다음과 같다.

- LOAD()

Market.txt파일을 읽어와서 FPtree를 생성한다. 이때 fp tree가 이미 있거나 불러올 txt가 없다면 flase를 반환하여 error를 출력하게 된다. Txt 한줄에는 같이 산 물품들이 저장되어있다. FPTable에 상품과 빈도수를 저장한다. 생성된 Table을 바탕으로 FP-tree를 구현할 수 있다. 입력받은 threshold보다 빈도수가 큰 item들로만 리스트를 만들어 fp-tree에 insert할 수 있다. 이때 tree에서 같은 item들끼리는 list로 연결해주어야 한다.

- BTLOAD()

  다음 명령어는 result.txt 파일을 입력 받아 B+tree를 구현한다. Result.txt에는 빈도수와 연관된 상품들이 저장되어 있다. set으로 연관된 item들을 저장한다. B+tree는 차수를 3을 하고 빈도수에 관련하여 나눌 수 있다.

- PRINT_ITEMLIST()

  다음 명령어는 FP-growth의 header table에 저장된 상품들을 내림차순으로 출력한다. Header table이 없을 때는 에러 메시지를 출력하고, 그렇지 않으면 모든 item들을 출력한다.

- PRINT_FPTREE()

  다음 명령어는 생성한 FPTREE를 출력하는 명령어이다. FP-TREE가 생성되지 않았다면 에러 메시지를 출력하고, 가장 낮은 빈도수부터 순차적으로 출력한다. 출력할 때 item의 부모 노드가 NULL일 때까지 출력하고, item에 연결된 모든 list를 출력할 때까지 반복한다.

- PRINT_MIN()

  다음 명령어는 item과 최소 빈도수를 입력 받아 standard-item과 frequentpattern frequency를 출력하게 된다. 최소 빈도수 이상인 list는 모두 출력한다.

- PRINT_CONFIDENCE()

  입력 받은 item, 연관률에서 item이 있는 list중에서 연관률이 입력 받은 연관률에 보다 크다면 모두 출력한다. 받아 standard-item과 frequent pattern frequency, confidence를 출력한다.

- PRINT_RANGE()

  범위를 지정하여 standard-item과 frequen tpattern frequency를 출력하게 된다. 입력 받은 범위 안에 있는 빈도수만 출력하게 된다.

- SAVE()

  Fp-growth tree를 연관된 상품들이 빈도수와 함께 저장하는 명령어이다. 다음 명령어 명 구현하였을 때 가산점을 받을 수 있다.

- EXIT()

  다음 명령어를 입력 받았을 때 프로그램을 종료한다.

3. Algorithm

이번 프로젝트에서는 다음과 같은 알고리즘이 사용되었다.

- Fp-growth 알고리즘

연관규칙 알고리즘이다. Fp-growth는 fp-tree라는 구조를 이용하여 apriori를 효과적으로 구현하는 것이다. Fp-tree의 구조는 tree, array, linked-list를 모두 합쳐 놓은 구조라고 볼 수 있다.

Market-Basket transactions

TID	Items
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

순차적으로 알고리즘을 진행할 수 있다.

TID	Items Bought
100	a, b, c, d, e, f, g, h
200	a, f, g
300	b, d, e, f, j
400	a, b, d, i, k
500	a, b, e, g

1. transaction 리스트를 포함하고 있는 아이템마다 support를 계산한다.

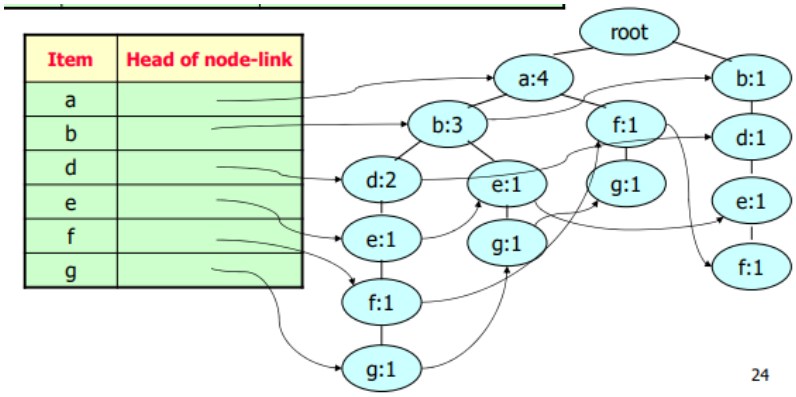
2. 구해진 support를 토대로 support가 threshold 이상인 아이템들만 골라낸다.

TID	Items Bought	(Ordered) Frequent Items
100	a, b, c, d, e, f, g, h	a, b, d, e, f, g
200	a, f, g	a, f, g
300	b, d, e, f, j	b, d, e, f
400	a, b, d, i, k	a, b, d
500	a, b, e, g	a, b, e, g

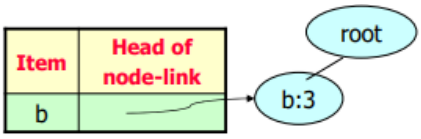
Item	Frequency
a	4
b	4
c	1
d	3
e	3
f	3
g	3
h	1
i	1
j	1
k	1

3. 골라낸 아이템들을 support 내림차순으로 정렬을 하여 header table을 구현한다.

4. transaction마다 root부터 삼입하고, 새로운 item들이면 새로운 node를 가리키도록 한다. 새로운 item이 아닌 원래 있던 item들이면 빈도수를 추가한다. 이렇게 해서 모든 list를 삽입한 tree를 global 트리라고 한다.



5. table에서 바텀업 방식으로 아이템을 하나씩 prefix로 잡고 그 아이템을 포함하는 transaction의 다른 item들의 support값을 계산하고, thresold보다 크거나 같은 경우 위와 같은 서브트리를 구현한다.



6. 이러한 방식으로 tree를 구현하고 support값을 구하면서 패턴이 threshold 이상의 패턴을 뽑아낸다.

참고

- https://process-mining.tistory.com/92

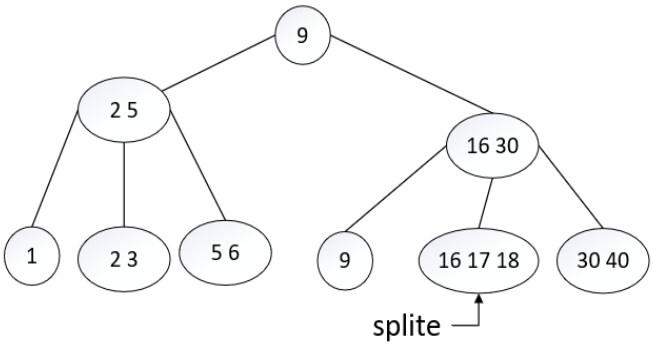
- 2020년 데이터구조설계 07.FP-Growth-Algorithm

- B+tree 알고리즘

B tree는 이진트리를 확장하여 더 많은 자식을 가질 수 있도록 하였다. B+트리는 index node와 data node로 구성되어 있다. index노드는 leaf node를 제외한 나머지 node들이고 data node는 leaf node라고 한다. 이때 data node들은 doubly link로 연결되어 있다.

이번 프로젝트에서는 order가 3이며 B+tree에 삽입만 존재한다.

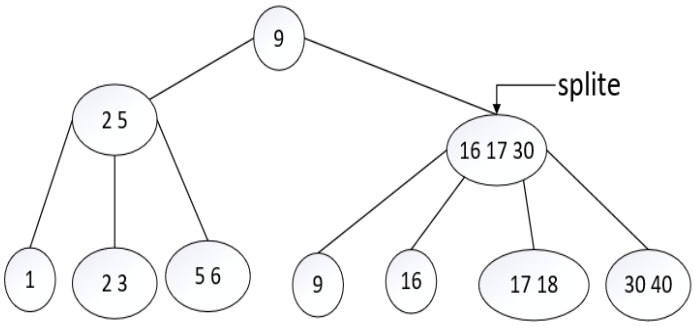
삽입은 다음과 같은 순서대로 진행된다.



17을 삽입할 경우이다.

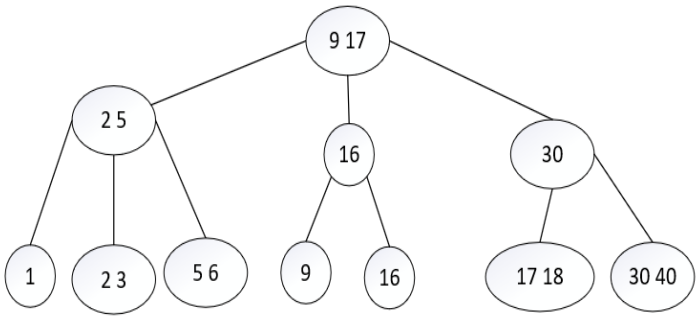
Order가 3이다. 최대 2개까지만 값을 가질 수 있는데 총 3개가 되었으므로 split을 해주어야 한다.

Splite할 때 가장 작은 수는 중간수의 왼쪽 자식 가장 큰 수는 중간수의 오른 쪽 자식으로 가는 형태로 만들어 주고 합칠 수 있다.



합쳐진 형태가 다음과 같다.

합쳐진 tree도 값을 총 3개 가지고 있으므로 split해줄 수 있다.



16, 17, 30 노드 마찬가지로 split해주어 tree에 합치면 다음과 같은 tree 형태로 바꿔줄 수 있다.

다음과 같은 방식으로 B+ tree 삽입을 할 수 있다.

4. Result Screen

- LOAD

```
========LOAD========
Success
========================
========LOAD========
======== ERROR 100 ========
========================
```

처음 명령어를 입력하였을 때 LOAD success를 출력하였다.

LOAD를 두 번 입력하였을 때 이미 fp-growth tree가 존재하기 때문에 error 메시지를 출력한다.

- PRINT_ITEMLIST

```
========PRINT_ITEMLIST========
soup 12
spaghetti 9
green tea 9
mineral water 7
milk 5
french fries 5
eggs 5
chocolate 5
ground beef 4
burgers 4
white wine 3
protein bar 3
honey 3
energy bar 3
chicken 3
body spray 3
avocado 3
whole wheat rice 2
turkey 2
shrimp 2
salmon 2
pasta 2
pancakes 2
hot dogs 2
grated cheese 2
frozen vegetables 2
frozen smoothie 2
fresh tuna 2
escalope 2
brownies 2
black tea 2
almonds 2
whole wheat pasta 1
toothpaste 1
tomatoes 1
soda 1
shampoo 1
shallot 1
red wine 1
pet food 1
pepper 1
parmesan cheese 1
meatballs 1
ham 1
gums 1
fresh bread 1
extra dark chocolate 1
energy drink 1
cottage cheese 1
cookies 1
carrots 1
bug spray 1
```

PRINT_ITEMLIST를 출력한다. Market.txt 파일을 read하고 빈도수를 내림차순으로 정렬한 Table을 출력하였다.

- PRINT_FPTREE

```
========PRINT_FPTREE========
{almonds,2}
(almonds,1) (burgers,1) (french fries,1) (eggs,1) (green tea,4) (soup,12)
(almonds,1) (turkey,1) (ground beef,1) (burgers,1) (eggs,1) (chocolate,1) (soup,12)
{black tea,2}
(black tea,1) (fresh tuna,1) (turkey,1) (chicken,1) (eggs,1) (mineral water,3) (spaghetti,5)
(black tea,1) (energy bar,1) (ground beef,1) (milk,1) (mineral water,3) (spaghetti,5)
{brownies,2}
(brownies,1) (hot dogs,1) (body spray,1) (avocado,1) (french fries,1) (green tea,4) (soup,12)
(brownies,1) (white wine,1) (chocolate,1) (green tea,2) (spaghetti,5)
{escalope,2}
(escalope,1) (frozen smoothie,1) (salmon,1) (black tea,1) (energy bar,1) (ground beef,1) (milk,1) (mineral water,3) (spaghetti,5)
(escalope,1) (frozen smoothie,1) (whole wheat rice,1) (frozen vegetables,1) (fresh tuna,1) (honey,1) (mineral water,1) (spaghetti,4) (soup,12)
{fresh tuna,2}
(fresh tuna,1) (turkey,1) (chicken,1) (eggs,1) (mineral water,3) (spaghetti,5)
(fresh tuna,1) (honey,1) (mineral water,1) (spaghetti,4) (soup,12)
{frozen smoothie,2}
(frozen smoothie,1) (salmon,1) (black tea,1) (energy bar,1) (ground beef,1) (milk,1) (mineral water,3) (spaghetti,5)
(frozen smoothie,1) (whole wheat rice,1) (frozen vegetables,1) (fresh tuna,1) (honey,1) (mineral water,1) (spaghetti,4) (soup,12)
{frozen vegetables,2}
(frozen vegetables,1) (fresh tuna,1) (honey,1) (mineral water,1) (spaghetti,4) (soup,12)
(frozen vegetables,1) (ground beef,1) (chocolate,1) (green tea,2) (spaghetti,4) (soup,12)
{grated cheese,2}
(grated cheese,1) (white wine,1) (honey,1) (avocado,1) (burgers,1)
(grated cheese,1) (white wine,1) (ground beef,1) (mineral water,3) (spaghetti,5)
{hot dogs,2}
(hot dogs,1) (body spray,1) (avocado,1) (french fries,1) (green tea,4) (soup,12)
(hot dogs,1) (almonds,1) (turkey,1) (ground beef,1) (burgers,1) (eggs,1) (chocolate,1) (soup,12)
{pancakes,2}
(pancakes,1) (body spray,1) (mineral water,1) (green tea,2) (spaghetti,5)
(pancakes,1) (brownies,1) (hot dogs,1) (body spray,1) (avocado,1) (french fries,1) (green tea,4) (soup,12)
{pasta,2}
(pasta,1) (shrimp,1) (chocolate,1) (eggs,1) (soup,12)
(pasta,1) (shrimp,1) (grated cheese,1) (white wine,1) (honey,1) (avocado,1) (burgers,1)
{salmon,2}
(salmon,1) (black tea,1) (fresh tuna,1) (turkey,1) (chicken,1) (eggs,1) (mineral water,3) (spaghetti,5)
(salmon,1) (black tea,1) (energy bar,1) (ground beef,1) (milk,1) (mineral water,3) (spaghetti,5)
{shrimp,2}
(shrimp,1) (chocolate,1) (eggs,1) (soup,12)
(shrimp,1) (grated cheese,1) (white wine,1) (honey,1) (avocado,1) (burgers,1)
{turkey,2}
(turkey,1) (chicken,1) (eggs,1) (mineral water,3) (spaghetti,5)
(turkey,1) (ground beef,1) (burgers,1) (eggs,1) (chocolate,1) (soup,12)
{whole wheat rice,2}
(whole wheat rice,1) (energy bar,1) (milk,1) (mineral water,1) (green tea,1)
(whole wheat rice,1) (frozen vegetables,1) (fresh tuna,1) (honey,1) (mineral water,1) (spaghetti,4) (soup,12)
{avocado,3}
(avocado,1) (burgers,1)
(avocado,1) (milk,1) (spaghetti,4) (soup,12)
(avocado,1) (french fries,1) (green tea,4) (soup,12)
{body spray,3}
(body spray,1) (mineral water,1) (green tea,2) (spaghetti,5)
(body spray,1) (avocado,1) (french fries,1) (green tea,4) (soup,12)
(body spray,1) (chicken,1) (green tea,4) (soup,12)
{chicken,3}
(chicken,1) (eggs,1) (mineral water,3) (spaghetti,5)
(chicken,1) (french fries,1) (chocolate,1) (eggs,1) (mineral water,1) (soup,12)
(chicken,1) (green tea,4) (soup,12)
{energy bar,3}
(energy bar,1) (milk,1) (mineral water,1) (green tea,1)
(energy bar,1) (ground beef,1) (milk,1) (mineral water,3) (spaghetti,5)
(energy bar,1) (soup,12)
{honey,3}
(honey,1) (french fries,1) (milk,1)
(honey,1) (avocado,1) (burgers,1)
(honey,1) (mineral water,1) (spaghetti,4) (soup,12)
{protein bar,3}
(protein bar,1) (honey,1) (french fries,1) (milk,1)
(protein bar,1) (green tea,4) (soup,12)
(protein bar,1) (energy bar,1) (soup,12)
```

{white wine,3}
(white wine,1) (honey,1) (avocado,1) (burgers,1)
(white wine,1) (chocolate,1) (green tea,2) (spaghetti,5)
(white wine,1) (ground beef,1) (mineral water,3) (spaghetti,5)
{burgers,4}
(burgers,1)
(burgers,1) (french fries,1) (milk,1) (green tea,2) (spaghetti,4) (soup,12)
(burgers,1) (french fries,1) (eggs,1) (green tea,4) (soup,12)
(burgers,1) (eggs,1) (chocolate,1) (soup,12)
{ground beef,4}
(ground beef,1) (milk,1) (mineral water,3) (spaghetti,5)
(ground beef,1) (mineral water,3) (spaghetti,5)
(ground beef,1) (chocolate,1) (green tea,2) (spaghetti,4) (soup,12)
(ground beef,1) (burgers,1) (eggs,1) (chocolate,1) (soup,12)
{chocolate,5}
(chocolate,1) (eggs,1) (soup,12)
(chocolate,1) (eggs,1) (mineral water,1) (soup,12)
(chocolate,1) (green tea,2) (spaghetti,5)
(chocolate,1) (green tea,2) (spaghetti,4) (soup,12)
(chocolate,1) (soup,12)
{eggs,5}
(eggs,1) (mineral water,3) (spaghetti,5)
(eggs,1) (soup,12)
(eggs,1) (mineral water,1) (soup,12)
(eggs,1) (green tea,4) (soup,12)
(eggs,1) (chocolate,1) (soup,12)
{french fries,5}
(french fries,1) (milk,1)
(french fries,1) (chocolate,1) (eggs,1) (mineral water,1) (soup,12)
(french fries,1) (green tea,4) (soup,12)
(french fries,1) (milk,1) (green tea,2) (spaghetti,4) (soup,12)
(french fries,1) (eggs,1) (green tea,4) (soup,12)
{milk,5}
(milk,1) (mineral water,1) (green tea,1)
(milk,1)
(milk,1) (spaghetti,4) (soup,12)
(milk,1) (mineral water,3) (spaghetti,5)
(milk,1) (green tea,2) (spaghetti,4) (soup,12)
{mineral water,7}
(mineral water,1) (green tea,1)
(mineral water,3) (spaghetti,5)
(mineral water,1) (green tea,2) (spaghetti,5)
(mineral water,1) (soup,12)
(mineral water,1) (spaghetti,4) (soup,12)
{green tea,9}
(green tea,1)
(green tea,2) (spaghetti,5)
(green tea,4) (soup,12)
(green tea,2) (spaghetti,4) (soup,12)
{spaghetti,9}
(spaghetti,5)
(spaghetti,4) (soup,12)
{soup,12}
(soup,12)

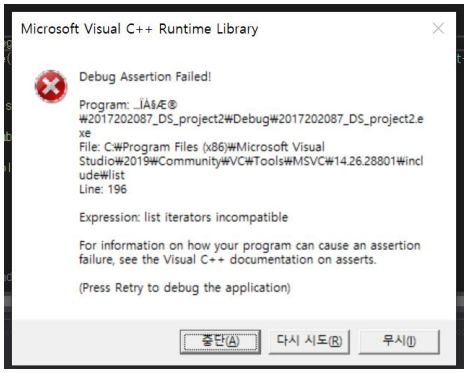
FPTREE를 출력한다. 출력은 빈도수가 낮은 순서대로 출력을 하고 list 순서대로 parent가 null일 때까지 출력된다.

●

5. Consideration

이번과제는 STL pair, list, map을 사용하여 fp-growth를 구현할 수 있었다. STL에 익숙하지 않아서 많은 시행착오를 겪었다.

또한 iterater 반복자를 처음 사용해보았다. List나 map을 출력하기 위해서 반복자를 사용하게 되는데 이 반복자는 list나 map의 begin으로 설정하고 end까지 반복문으로 출력할 수 있었다.



하지만 다음과 같은 에러가 계속 발생하고 프로그램이 진행되지 못하였다. Iterater가 list의 시작주소를 찾지 못하였다.

오류를 구글링한 결과 iterator가 begin과 end 두번 초기화되고 다른 값을 가리키고 있어 begin에서 end까지 반복하여 출력하지 못하였다.

그래서 list나 map을 먼저 선언해준 후 반복자를 사용하여 list를 출력할 수 있었다.

B+ tree를 구현하는데 어려움을 겪었다. 결국 오류를 해결하지 못해 tree를 구현하지 못하였다. 시간이 좀 더 있었으면 구현할 수 있었을 것 같다는 아쉬움이 있다.