Solution to COMP5121 Assignment 1

QING Pei, 11500811G

October 7, 2011

Contents

1 Part A

"Maintenance" should be EXCLUDED from the analysis.

1.1 The Isolated "Mantenance"

There are 10 transactions containing "maintenance". None of them involve any other items. If the sample of transactional data is representative and unbiased, it is natural to assume that customers coming for maintenance would not bother purchasing any items in his store. Thus the support for an item set of more than just "Maintenance" would be pretty low. Actually, in this specific case, that support is always 0%. The mining of potential association rules involving "maintenance" is therefore filtered out for lack of support.

1.2 Impact of the Removal of "Maintenance"

Removing data entries of "Maintenance" has both positive and negative consequences.
The good news is that we only have to process 80 percent of the original dataset. Even if the algorithm is O(N), this saves a lot of time. (E.g. Scanning through the table for support of a given item set in Part B.) Not to mention if the algorithm is more complex.
The bad news, however, is that the support and confidence values will change slightly from the **real** values. This is not a primary concern, so we just accept that little sacrifice in accuracy.

2 Part B

The Minimum Support is 14%, i.e. 40*14% = 5.6 appearances.
The Minimum Confidence is 85%.

2.1 Simpler Notation

Table 1 shows the aliases for the items for faster manual work. The items will appear in tables as C,D,G,M and S.

Table 1: Item Aliases

Item	Alias
Display Card	G
Speaker	S
Desktop	D
Case	C
Mouse	M

2.2 The Sample Data For Calculation

Group the entries in the original sample data by TransID as the following, using aliases and removing "maintenance" transactions, as Table 2.

2.3 Unit of Support

As the support of an item is always $appearances/40$, we may just count the appearances and use that as a NOMINAL SUPPORT, with a unit of 1/40.

Table 2: Data with Aliases

TID	Items	TID cont.	Item cont.
1	G,S	26	C,G,S
2	D,G	27	C,D,G
3	C,S	28	C,D,G,M,S
4	S	29	D
5	G,M	30	G
7	M,D	31	G
8	C	33	C
9	D	35	C
10	M	36	G
12	C	37	C,D,M
13	C,D,G,M,S	38	D,C
14	G	40	D
16	M	41	C
18	G	42	D
19	C,D,G,M,S	44	C
21	C	45	G
22	C,G	46	D,G,M
23	D,G,M,S	47	C
24	D,G	49	C,G,S
25	C,D,G,M,S	50	C,D,G,M,S

2.4 Support of Single Items

Scan Table 2 to count the support for each item as Table 3.

Table 3: Support of 1-item sets, with supp>5.6.

Item	Support
G	21
S	11
D	17
C	20
M	12

As all the supports are greater than 5.6, Table 3 is kept for the next step with no modification.

2.5 2-Item Sets

Scan Table 2 again counting 2-item pairs as Table 4.

2.6 3-Item Sets

Scan Table 2 and count to get Table 5 for 3-item sets.
Only keep rows with supp>5.6 and we have Table 6.

Table 4: Support of 2-item sets, with supp>5.6.

Item	Support
{G,S}	9
{G,D}	10
{G,C}	9
{G,M}	8
{S,D}	6
{S,C}	8
{S,M}	6
{D,C}	7
{D,M}	9
{C,M}	6

Table 5: Support of 3-item sets.

Item	Support
{G,S,C}	7
{G,D,C}	6
{G,D,M}	7
{G,S,D}	6
{G,S,M}	6
{G,C,M}	5
{S,D,C}	5
{S,D,M}	6
{S,C,M}	5
{C,D,M}	6

Table 6: Support of 3-item sets, with supp>5.6.

Item	Support
{G,S,C}	7
{G,D,C}	6
{G,D,M}	7
{G,S,D}	6
{G,S,M}	6
{S,D,M}	6
{C,D,M}	6

2.7 4-Item Sets

Get Table 7 for 4-item sets the same way.
Remove rows with supp<5.6 to get Table 8.
No more possible pair of item sets with support over 5.6.

Table 7: Support of 4-item sets.

Item	Support
{G,S,C,M}	5
{G,S,C,D}	5
{G,D,C,M}	5
{G,D,M,S}	6
{S,D,M,C}	5

Table 8: Support of 4-item sets, with supp>5.6.

Item	Support
{G,D,M,S}	6

2.8 Calculate Confidences

The item-sets with sufficient support are found in Table 4, Table 6 and Table 8 now. Go on to calculate the confidence for all combinations of items as implication rules, shown in Table 9. Here the confidence is calculated as:

$$Conf(A \rightarrow B) = \frac{Support(A\&B)}{Support(A)} \tag{1}$$

Table 9: Calculating Confidences

LHS	**RHS**	**Supp(LHS∧RHS)**	**Supp(LHS)**	**Conf**	**Conf≥0.85**
C	D	7	20	0.35	
C	D,G	6	20	0.3	
C	D,M	6	20	0.3	
C	G	9	20	0.45	
C	G,S	7	20	0.35	
C	S	8	20	0.4	
C,D	G	6	7	0.85714286	Yes
C,D	M	6	7	0.85714286	Yes
C,G	D	6	9	0.66666667	
C,G	S	7	9	0.77777778	
C,M	D	6	6	1	Yes
C,S	G	7	8	0.875	Yes
D	C	7	17	0.41176471	
D	C,G	6	17	0.35294118	
D	C,M	6	17	0.35294118	
D	G	10	17	0.58823529	
D	G,M	7	17	0.41176471	
D	G,M,S	6	17	0.35294118	
D	G,S	6	17	0.35294118	
D	M	9	17	0.52941176	

Continued on next page

Table 9 – continued from previous page

LHS	RHS	Supp(LHS∧RHS)	Supp(LHS)	Conf	Conf≥0.85
D	M,S	6	17	0.35294118	
D,G	C	6	10	0.6	
D,G	G,S	6	10	0.6	
D,G	M	7	10	0.7	
D,G	M,S	6	10	0.6	
D,G	S	6	10	0.6	
D,G,M	S	6	7	0.85714286	Yes
D,G,S	M	6	6	1	Yes
D,M	C	6	9	0.66666667	
D,M	G	7	9	0.77777778	
D,M	S	6	9	0.66666667	
D,M,S	G	6	6	1	Yes
D,S	G	6	6	1	Yes
D,S	G,M	6	6	1	Yes
D,S	M	6	6	1	Yes
G	C	9	21	0.42857143	
G	C,D	6	21	0.28571429	
G	C,S	7	21	0.33333333	
G	D	10	21	0.47619048	
G	D,M	7	21	0.33333333	
G	D,M,S	6	21	0.28571429	
G	D,S	6	21	0.28571429	
G	M	8	21	0.38095238	
G	M,S	6	21	0.28571429	
G	S	9	21	0.42857143	
G,M	D	7	8	0.875	Yes
G,M	D,S	6	8	0.75	
G,M	S	6	8	0.75	
G,M,S	D	6	6	1	Yes
G,S	C	7	9	0.77777778	
G,S	D	6	9	0.66666667	
G,S	D,M	6	9	0.66666667	
G,S	M	6	9	0.66666667	
M	C,D	6	12	0.5	
M	D	9	12	0.75	
M	D,G	7	12	0.58333333	
M	D,G,S	6	12	0.5	
M	D,S	6	12	0.5	
M	G	8	12	0.66666667	
M	G,S	6	12	0.5	
M,S	D	6	6	1	Yes
M,S	D,G	6	6	1	Yes
M,S	G	6	6	1	Yes
S	C	8	11	0.72727273	
S	C,G	7	11	0.63636364	
S	D,G	6	11	0.54545455	
S	D,G,M	6	11	0.54545455	

Continued on next page

Table 9 – continued from previous page

LHS	RHS	Supp(LHS∧RHS)	Supp(LHS)	Conf	Conf≥0.85
S	D,M	6	11	0.54545455	
S	G	9	11	0.81818182	
S	G,M	6	11	0.54545455	

2.9 Interesting Rules

With Minimum Confidence set to 85%, the interesting rules are shown in Table 10.

Table 10: List of interesting rules found.

LHS	RHS	Support	Confidence
C,D	G	0.15	0.85714286
C,D	M	0.15	0.85714286
C,M	D	0.15	1
C,S	G	0.175	0.875
D,G,M	S	0.15	0.85714286
D,G,S	M	0.15	1
D,M,S	G	0.15	1
D,S	G	0.15	1
D,S	G,M	0.15	1
D,S	M	0.15	1
G,M	D	0.175	0.875
G,M,S	D	0.15	1
M,S	D	0.15	1
M,S	D,G	0.15	1
M,S	G	0.15	1

3 Part C

For the interesting rules found in Part B, their lift ratios are calculated as

$$Lift(A \rightarrow B) = \frac{Rule\ Confidence}{Expected\ Confidence\ of\ Consequent(RHS)} = \frac{Rule\ Confidence}{Support(RHS)} \quad (2)$$

The results are shown in Table 11.
Lift ratios of 11 rules are greater than the minimum lift ratio of 2. 6 rules found in Part B are considered no longer interesting.

4 Part D

As the datafile contains some typos and NULL values, I would show the steps for data cleanup before finding the rules.

Table 11: Interesting Rules

LHS	RHS	Exp Conf * 40	Conf	Lift Ratio	Lift≥2?
C,S	G	21	7/8	1.6666667	
G,M	D	17	7/8	2.0588235	Yes
C,D	G	21	0.85714286	1.6326531	
C,D	M	12	0.85714286	2.8571429	Yes
C,M	D	17	1	2.3529412	Yes
C,S	G	21	0.875	1.6666667	
D,G,M	S	11	0.85714286	3.1168831	Yes
D,G,S	M	12	1	3.3333333	Yes
D,M,S	G	21	1	1.9047619	
D,S	G	21	1	1.9047619	
D,S	G,M	8	1	5.	Yes
D,S	M	12	1	3.3333333	Yes
G,M	D	17	0.875	2.0588235	Yes
G,M,S	D	17	1	2.3529412	Yes
M,S	D	17	1	2.3529412	Yes
M,S	D,G	10	1	4.	Yes
M,S	G	21	1	1.9047619	

4.1 Data Cleanup

At the very beginning, we load the csv file to a data frame named *df*.

```
> df <- read.csv("data-q1e.csv", header = TRUE, sep = ",", na.strings = "")
```

Then we want to examine whether the data is ready for applying the Apriori algorithm. We may have a look at the **Product** column.

```
> summary(df$Product, maxsum = 20)
```

```
        CASE  CRT-Monitor         Case         Cese      Desjtop      Desktop
        1013         1430         1557            3          794         1229
Diaplay Card Display Card     Keyboard  LCD-Monitor        Mouse     Notebook
        1002         1660         1990         1760         1980         2163
     Printer      SPeaker   Sound Card      Speaker         case      desktop
        1317           94          935         1457            1            1
     printer         NA's
           1          590
```

As we can see in the result, there are some ”noisy” cells in the table:

- Inconsistent cases. E.g. ”CASE” and ”Case”.
- Typos. E.g. Desjtop
- Null values. 590 NA’s reported.

For typos and differences in letter cases, map the original value(s) to a new aggregate value. For example, ”case”, ”CASE” and ”Cese” are all mapped to ”Case” for future processing. Here is how all of the typos and case differences are mapped.

```
> df$Product[df$Product == "case" | df$Product == "CASE" | df$Product ==
+     "Cese"] <- "Case"
> df$Product[df$Product == "desktop" | df$Product == "Desjtop"] <- "Desktop"
> df$Product[df$Product == "Diaplay Card"] <- "Display Card"
> df$Product[df$Product == "printer"] <- "Printer"
> df$Product[df$Product == "SPeaker"] <- "Speaker"
```

Examine the data again, we should have a "cleaner" data table so far. The following command shows the current distribution of the **Product** column. Cells previously counted as "CASE", "case" and "Cese" no longer exist. They are all "Case"'s instead.

```
> summary(df$Product, maxsum = 20)
```

```
        CASE  CRT-Monitor         Case         Cese      Desjtop      Desktop
           0         1430         2574            0            0         2024
Diaplay Card Display Card     Keyboard  LCD-Monitor        Mouse     Notebook
           0         2662         1990         1760         1980         2163
     Printer      SPeaker   Sound Card      Speaker         case      desktop
        1318            0          935         1551            0            0
     printer         NA's
           0          590
```

NA's should also be excluded from the association rules. Since each row of the table is a TID-Product pair, we may just remove all the rows with NA values and keep the rest for rule mining.
The steps are:

- Count how many valid cells are in each row.
- Mark rows with 2 valid cells as "good rows".
- Select good rows in the original data frame into a new data frame called *df.NAexcluded*.

```
> numbers.present <- rowSums(df != "")
> good.rows <- which(numbers.present == 2)
> df.NAexcluded <- df[good.rows, ]
```

Examine the **Product** column of df.NAexcluded again and it is shown that NA's are gone.

```
> summary(df.NAexcluded$Product, maxsum = 12)
```

```
Display Card         Case     Notebook      Desktop     Keyboard        Mouse
        2662         2574         2163         2024         1990         1980
 LCD-Monitor      Speaker  CRT-Monitor      Printer   Sound Card      (Other)
        1760         1551         1430         1318          935            0
```

Finally, we export the clean data to "data-q1e-clean.csv" for next steps.

```
> names(df.NAexcluded) <- NULL
> write.table(df.NAexcluded, "data-q1e-clean.csv", sep = ",", col.names = F,
+     row.names = F, quote = F)
```

4.2 Rule Mining and Suggestions

4.2.1 Apriori with R

With the clean data, we may start to find interesting rules. This is done with the **arules** package for GNU R.
First, load the package:

```
> library(arules)
```

Then load the transaction data from "data-q1e-clean.csv" we prepared in 4.1.

```
> trans <- read.transactions(file = "data-q1e-clean.csv", format = "single",
+     sep = ",", cols = c(1, 2), rm.duplicates = T)
```

The following command shows the number of transactions loaded.

```
> length(trans)

[1] 6611
```

Use Apriori algorithm to find association rules:

```
> rules <- apriori(trans, parameter = list(supp = 0.14, conf = 0.87))

parameter specification:
 confidence minval smax arem  aval originalSupport support minlen maxlen target
       0.87    0.1    1 none FALSE            TRUE    0.14      1     10  rules
   ext
 FALSE

algorithmic control:
 filter tree heap memopt load sort verbose
    0.1 TRUE TRUE  FALSE TRUE    2    TRUE

apriori - find association rules with the apriori algorithm
version 4.21 (2004.05.09)        (c) 1996-2004   Christian Borgelt
set item appearances ...[0 item(s)] done [0.00s].
set transactions ...[11 item(s), 6611 transaction(s)] done [0.00s].
sorting and recoding items ... [11 item(s)] done [0.00s].
creating transaction tree ... done [0.00s].
checking subsets of size 1 2 3 4 done [0.00s].
writing ... [4 rule(s)] done [0.00s].
creating S4 object  ... done [0.00s].
```

View the rules found:

```
> inspect(rules)

  lhs               rhs              support confidence     lift
1 {Case,
   Speaker}      => {Display Card} 0.1480865  0.8741071 2.174905
2 {Display Card,
   Speaker}      => {Case}         0.1480865  0.9081633 2.337045
```

```
3 {Display Card,
   Mouse}        => {Case}         0.1582211  0.8797309 2.263877
4 {Case,
   Desktop}      => {Display Card} 0.1495992  0.8830357 2.197120
```

4.2.2 Suggestions to Improve Sales

- Display cards and cases are the most popular products sold. Therefore, it might help to adapt the stock combination to have have more display cards and cases in stock to avoid out-of-stock disappointments.
- Sound cards are the least bought one. Keep less sound cards in stock to reduce storage costs.
- People who by case and speaker are 2 times more likely to also buy a display card. Bundling case and speaker with a discount price to attract customer to buy both the items to boost the sales of display cards.
- It may also work to bundle display card and speaker to increase sales of cases.
- According to the other two rules found, bundling display card and mouse or bundling case and desktop could also make a difference.
- In general, it might help to let customer by 2 products as a bundle with a discount price. When they actually purchased the bundle, they are 2 times more likely to buy a third product.
- But it seems sound cards do not appear in the rules. I suggest bundling sound cards with the most popular products, i.e. display cards and cases with more discount to see whether this leads to a surge in sound card sales.

4.3 Discover the min_{supp} and min_{conf} Parameters

The finding of an optimal min_{conf}, min_{sup} parameter is basically a trial-and-error process. The following is always true about changes to the parameters:

- Increasing one of the parameter with the other fixed will lead to less (or the same number of) rules found.
- The rule sets found with lower(higher) support and higher(lower) confidence may be close in size, but with different characteristics of the data.

4.3.1 The Starting Point

With the previous work in 2.9, it turns out (14%,85%) is not too bad for the sample data. If the sample is well selected, the same parameter would also work for the complete dataset. Then we modify the parameters and see how the interestingness of the result rules change accordingly. By repeating this, we will finally get to the optimal parameter (or one of the optima).

4.3.2 Interestingness Measurement

Also, we have to define what is "interestingness" of a rule. Here I used the conviction measurement introduced by Sergey Brin, Rajeev Motwani, Jeffrey D. Ullman, and Shalom Tsur. Dynamic itemset counting and implication rules for market basket data. In SIGMOD 1997, Proceedings ACM SIGMOD International Conference on Management of Data, pages 255-264, Tucson, Arizona, USA, May 1997.

$$conviction(X \to Y) = \frac{1 - supp(Y)}{1 - conf(X \to Y)} = \frac{P(X)P(\bar{Y})}{P(X \wedge \bar{Y})} \tag{3}$$

The range of conviction is 0.5 to infinity. 1 indicates unrelated items. If we mine rules with $min_{sup} = 0.07, min_{conf} = 0.99$, the results will have antecedents and consequences which are perfectly related to each other. In other words, such rules could be called common senses. We are mining rules that are not obvious, therefore the target is to find a pair of $(min_{conf}, minsupp)$ which results in a set of rules with the mean of conviction as close to 1 as possible. The least size of the rule set is 4.

4.3.3 Trial and Error Example

Let's try with $min_{sup} = 0.14, min_{conf} = 0.85$:

```
> rules1 <- apriori(trans, parameter = list(supp = 0.14, conf = 0.85))
> quality(rules1) <- cbind(quality(rules1), conviction = interestMeasure(rules1,
+     method = "conviction", trans))

> summary(rules1)

set of 6 rules

rule length distribution (lhs + rhs):sizes
3
6

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
      3       3       3       3       3       3

summary of quality measures:
    support         confidence          lift         conviction
 Min.   :0.1413   Min.   :0.8567   Min.   :2.132   Min.   :4.173
 1st Qu.:0.1481   1st Qu.:0.8683   1st Qu.:2.161   1st Qu.:4.546
 Median :0.1488   Median :0.8769   Median :2.186   Median :4.917
 Mean   :0.1506   Mean   :0.8780   Mean   :2.210   Mean   :5.043
 3rd Qu.:0.1561   3rd Qu.:0.8822   3rd Qu.:2.247   3rd Qu.:5.106
 Max.   :0.1582   Max.   :0.9082   Max.   :2.337   Max.   :6.658

mining info:
  data ntransactions support confidence
 trans          6611    0.14       0.85
```

6 rules are found, the mean of conviction is 5.043.
Try again with $min_{sup} = 0.12, min_{conf} = 0.9$:

```
> rules2 <- apriori(trans, parameter = list(supp = 0.12, conf = 0.9))
> quality(rules2) <- cbind(quality(rules2), conviction = interestMeasure(rules2,
+     method = "conviction", trans))

> summary(rules2)

set of 10 rules

rule length distribution (lhs + rhs):sizes
3 4
5 5

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
    3.0     3.0     3.5     3.5     4.0     4.0

summary of quality measures:
    support         confidence          lift         conviction
 Min.   :0.1212   Min.   :0.9011   Min.   :2.251   Min.   : 6.184
 1st Qu.:0.1215   1st Qu.:0.9095   1st Qu.:2.279   1st Qu.: 6.721
 Median :0.1249   Median :0.9166   Median :2.328   Median : 7.169
 Mean   :0.1271   Mean   :0.9173   Mean   :2.321   Mean   : 7.528
 3rd Qu.:0.1292   3rd Qu.:0.9183   3rd Qu.:2.361   3rd Qu.: 7.485
 Max.   :0.1481   Max.   :0.9479   Max.   :2.389   Max.   :11.486

mining info:
  data ntransactions support confidence
 trans          6611    0.12        0.9
```

We get a set of 10 rules with mean conviction of 7.528, which is worse according to our interest (for mean conviction close to 1).
So we try in another direction with $min_{sup} = 0.15, min_{conf} = 0.8$:

```
> rules3 <- apriori(trans, parameter = list(supp = 0.15, conf = 0.8))
> quality(rules3) <- cbind(quality(rules3), conviction = interestMeasure(rules3,
+     method = "conviction", trans))

> summary(rules3)

set of 4 rules

rule length distribution (lhs + rhs):sizes
3
4

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
      3       3       3       3       3       3

summary of quality measures:
    support         confidence          lift         conviction
 Min.   :0.1582   Min.   :0.8252   Min.   :2.089   Min.   :3.497
 1st Qu.:0.1582   1st Qu.:0.8361   1st Qu.:2.115   1st Qu.:3.673
```

```
 Median :0.1619   Median :0.8482   Median :2.128   Median :3.952
 Mean   :0.1619   Mean   :0.8503   Mean   :2.152   Mean   :4.121
 3rd Qu.:0.1656   3rd Qu.:0.8624   3rd Qu.:2.165   3rd Qu.:4.401
 Max.   :0.1656   Max.   :0.8797   Max.   :2.264   Max.   :5.084

mining info:
  data ntransactions support confidence
 trans          6611    0.15        0.8
```

This time the mean conviction is 4.121, better than the first try.

4.3.4 The Optimal Parameter

After several tries, we may get to $min_{sup} = 0.2, min_{conf} = 0.4$:

```
> rules4 <- apriori(trans, parameter = list(supp = 0.2, conf = 0.4))
> quality(rules4) <- cbind(quality(rules4), conviction = interestMeasure(rules4,
+     method = "conviction", trans))

> summary(rules4)

set of 5 rules

rule length distribution (lhs + rhs):sizes
1 2
1 4

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
    1.0     2.0     2.0     1.8     2.0     2.0

summary of quality measures:
    support         confidence          lift         conviction
 Min.   :0.2007   Min.   :0.4019   Min.   :1.000   Min.   :1.000
 1st Qu.:0.2007   1st Qu.:0.4994   1st Qu.:1.438   1st Qu.:1.346
 Median :0.2246   Median :0.5589   Median :1.438   Median :1.386
 Mean   :0.2505   Mean   :0.5307   Mean   :1.388   Mean   :1.341
 3rd Qu.:0.2246   3rd Qu.:0.5780   3rd Qu.:1.531   3rd Qu.:1.417
 Max.   :0.4019   Max.   :0.6152   Max.   :1.531   Max.   :1.554

mining info:
  data ntransactions support confidence
 trans          6611     0.2        0.4
```

The rules are:

```
> inspect(rules4)

  lhs               rhs              support confidence     lift conviction
1 {}             => {Display Card} 0.4019059  0.4019059 1.000000   1.000000
2 {Notebook}     => {Display Card} 0.2007261  0.6152063 1.530722   1.554324
3 {Display Card} => {Notebook}     0.2007261  0.4994355 1.530722   1.345932
4 {Case}         => {Display Card} 0.2246256  0.5780459 1.438262   1.417439
5 {Display Card} => {Case}         0.2246256  0.5589010 1.438262   1.386095
```

4.3.5 Problems with the Measurement

If we have chosen another measurement for interestingness of association rules, e.g. lift, this would be a totally different story. The optimal parameter would change as well as the rules found.
Suppose we really chose lift as the criteria, we may start from $min_{sup} = 0.04, min_{conf} = 0.3$ to have a look at the highest lifts we may get from a large pool of rules:

```
> rules <- apriori(trans, parameter = list(supp = 0.04, conf = 0.3))

> inspect(sort(rules, by = "lift")[1:5])

  lhs                rhs          support confidence     lift
1 {CRT-Monitor,
   Display Card,
   Mouse,
   Sound Card}    => {Speaker} 0.04159734  1.0000000 4.273432
2 {CRT-Monitor,
   Display Card,
   Keyboard,
   Sound Card}    => {Speaker} 0.04159734  1.0000000 4.273432
3 {CRT-Monitor,
   Display Card,
   Sound Card}    => {Speaker} 0.04492512  0.9642857 4.120810
4 {CRT-Monitor,
   Case,
   Display Card,
   Sound Card}    => {Speaker} 0.04326123  0.9629630 4.115157
5 {CRT-Monitor,
   Mouse,
   Sound Card}    => {Speaker} 0.04159734  0.9615385 4.109070
```

The lift are over 4, but people may think a support of 4 percent is too weak, or a confidence of 40 percent is far from sufficient. So we try $min_{sup} = 0.12, min_{conf} = 0.85$:

```
> rules <- apriori(trans, parameter = list(supp = 0.12, conf = 0.85))

> inspect(sort(rules, by = "lift")[1:5])

  lhs                rhs                 support confidence     lift
1 {Keyboard,
   Speaker}       => {Case}            0.1250945  0.9281706 2.388531
2 {Mouse,
   Speaker}       => {Case}            0.1279685  0.9185668 2.363817
3 {Display Card,
   Mouse,
   Notebook}      => {Case}            0.1296324  0.9175589 2.361223
4 {Case,
   Desktop,
   Notebook}      => {Display Card}    0.1211617  0.9479290 2.358584
5 {Display Card,
   Speaker}       => {Case}            0.1480865  0.9081633 2.337045
```

We get higher support and confidence, but the lift is lowered accordingly. Now we have question: is there any interestingness measurement of association rules that is neither monotonically decreasing nor monotonically increasing with our change in parameters?

4.3.6 AltI: An Alternative Measurement of Interesingness

The following basic rules should apply to our interestingness measurement:

- Rules with higher support are more interesting.
- Rules with higher confidence are more interesting.
- Rules with higher lift are more interesting.

We define the following as the interestingness of a rule:

$$AltI = min_{supp} * min_{conf} * lift \tag{4}$$

Define a function in R to calculate the interestingness:

```
> AltI <- function(x, transactions) interestMeasure(x, method = "support",
+     transactions) * interestMeasure(x, method = "confidence",
+     transactions) * interestMeasure(x, method = "lift", transactions)
```

Then we may apply the same trial-and-error approach to find the set of rules with the maximum "interestingness".
Tryout different parameters:

Table 12: AltI is not monotonic.

min_{conf}	min_{supp}	interestingness
0.05	0.4	0.167
0.07	0.4	0.195
0.15	0.4	0.171
0.05	0.6	0.173
0.12	0.6	0.231
0.15	0.6	0.225

From the first 3 rows in Table 12, there must be an extreme point of AltI when changing min_{conf}. From the last 3 rows, AltI also has an extreme point when changing min_{supp}. And thus there must be an optimal pair of (min_{conf}, min_{supp}) that provides maximum interestingness.

4.3.7 Optimal Parameter to Maximize AltI

With the same trial-and-error approach, we may get the optimal parameter to maximize AltI as $min_{sup} = 0.14, min_{conf} = 0.87$
The rules found are:

```
> rules <- apriori(trans, parameter = list(supp = 0.14, conf = 0.87))
> quality(rules) <- cbind(quality(rules), interestingness = AltI(rules,
+     trans))
> summary(rules)
```

```
> inspect(rules)

  lhs               rhs              support confidence     lift interestingness
1 {Case,
   Speaker}      => {Display Card} 0.1480865  0.8741071 2.174905       0.2815273
2 {Display Card,
   Speaker}      => {Case}         0.1480865  0.9081633 2.337045       0.3143015
3 {Display Card,
   Mouse}        => {Case}         0.1582211  0.8797309 2.263877       0.3151137
4 {Case,
   Desktop}      => {Display Card} 0.1495992  0.8830357 2.197120       0.2902427
```

5 Part E

5.1 Level 2 Rules

In the previous "data-q1e-clean.csv", map all the products to a higher level of abstraction:

- Map "Desktop" and "Notebook" to "Computers".
- "LCD-Monitor" and "CRT-Monitor" to "Monitors".
- The same to the rest products.

Save this as "data-q1e-l2.csv". And we have the data at level 2 abstraction.
Find the association rules with:

```
> transl2 <- read.transactions(file = "data-q1e-l2.csv", format = "single",
+     sep = ",", cols = c(1, 2), rm.duplicates = T)
> rulesl2 <- apriori(transl2, parameter = list(supp = 0.2, conf = 0.85))

> inspect(rulesl2)

  lhs                 rhs                support confidence     lift
1 {IntPeripherals,
   Monitor}        => {Computers}      0.2023900  0.8699610 1.854664
2 {IntPeripherals,
   Monitor}        => {ExtPeripherals} 0.2143397  0.9213264 1.348735
3 {Computers,
   Monitor}        => {ExtPeripherals} 0.2288610  0.9191981 1.345620
4 {Computers,
   IntPeripherals} => {ExtPeripherals} 0.2456512  0.9012209 1.319303
```

Considering the lift ratios, the 1st rule is interesting. $\{IntPeripherals, Monitor\} \rightarrow \{Computers\}$

5.2 Level 3 Rules

More abstraction of "External Peripherals" and "Internal Peripherals" will give the following results:

```
> transl3 <- read.transactions(file = "data-q1e-l3.csv", format = "single",
+     sep = ",", cols = c(1, 2), rm.duplicates = T)
> rulesl3 <- apriori(transl3, parameter = list(supp = 0.2, conf = 0.8))
```

```
> inspect(rules13)

  lhs              rhs             support confidence      lift
1 {}            => {Peripherals} 0.8271063  0.8271063 1.0000000
2 {Monitor}     => {Peripherals} 0.2955680  0.8172313 0.9880607
3 {Computers,
   Monitor}     => {Peripherals} 0.2405082  0.9659781 1.1679008
4 {Monitor,
   Peripherals} => {Computers}   0.2405082  0.8137155 1.7347542
```

At this level, the 1st rule tells us that peripheral is so frequent that 82 percent of the transactions include some peripheral. And the 2nd and 3rd rules are two examples of redundant rules, in which the LHSs are all descendant of LHS of the 1st rule which is implicitly "Products".
The 4th rule is interesting as the lift ratio is somewhat far from 1.

5.3 Cross Level Rules

So far we have found out rules at different level of abstraction in **??**, 5.1 and 5.2.
But some of them are so general that the lift ratios are very close to 1. Others might have a high lift ratio but with rather low support.
By mining cross-level rules, we might be able to find something more interesting.

```
> transcross <- read.transactions(file = "data-q1e-cross.csv",
+     format = "single", sep = ",", cols = c(1, 2), rm.duplicates = T)
> rulescross <- apriori(transcross, parameter = list(supp = 0.2,
+     conf = 0.8))

> inspect(sort(rulescross, decreasing = FALSE, by = "confidence")[1:10])

   lhs                       rhs                       support confidence     lift
1  {Computers,
    Monitor}              => {Internal Peripherals} 0.2023900  0.8128797 1.799715
2  {External Peripherals,
    Internal Peripherals} => {Computers}            0.2406595  0.8129790 1.733184
3  {External Peripherals,
    Internal Peripherals} => {Case}                 0.2427772  0.8201329 2.110509
4  {Case,
    Internal Peripherals} => {Computers}            0.2022387  0.8330218 1.775913
5  {Case,
    External Peripherals,
    Internal Peripherals} => {Computers}            0.2022387  0.8330218 1.775913
6  {Display Card,
    External Peripherals} => {Case}                 0.2246256  0.8389831 2.159018
7  {Display Card,
    External Peripherals,
    Internal Peripherals} => {Case}                 0.2246256  0.8389831 2.159018
8  {External Peripherals,
    Monitor}              => {Computers}            0.2271971  0.8395752 1.789884
9  {Computers,
    External Peripherals,
```

```
    Internal Peripherals} => {Case}                 0.2022387  0.8403520 2.162541
10 {Case,
    Computers}            => {Internal Peripherals} 0.2022387  0.8467384 1.874678
```

There should be some filtering on the results. The confidence of rule 3, 6, 7 and 9 may be largely influenced by the inheritance existing in LHS and RHS.
Looking at rule 1 and 10, we may want to know if "Monitor" or "Case" contributes to the $Computers \rightarrow InternalPeripherals$ association. With further calculation, we find out that for the more general rule, the confidence is below 60%. Therefore rule 1 and 10 are not unnecessary.
Rule 4 is redundant since there is rule 2.
Rule 5 is unnecessary because of rule 2.
Rule 1, 2, 8 and 10 are still interesting after filtering. And they are helpful to remove unnecessary or redundant rules found previously at the lowest level. E.g. $\{Case, Desktop\} \rightarrow \{DisplayCard\}$ is redundant because of the rule $\{Case, Computers\} \rightarrow \{InternalPeripherals\}$.

6 Paper Review

With product taxonomy, products can be grouped into a superset for data mining at a higher level. This is desirable in different ways.

- The product set may contain too many items for machine with limited speed and storage to deal with.
- People may want to know the association of one group of products with another.
- Trivial differences in products are known not interesting and therefore should be ignored during the mining process.
- By abstraction, it is possible to remove some redundant rules.

In real-world applications, a miner may use such abstraction to filter the input data with a sneak peak and then do further investigation only on part of the data which is more important/interesting/relevant to the requirements.

6.1 Reducing Calculation

In Apriori, the calculation of support for itemsets is heavy. With a product taxonomy, the calculation is carried out from a top-down view of the tree. If a high level node in the tree has insufficient support, none of its descendants would ever have a chance to have sufficient support. Taking this into consideration, there would be much less calculation for the frequent itemsets table. And thus speed up the Apriori process.

6.2 Mining General Rules

People may have the need to know whether there is an association between "Computers" and "Monitors". The raw data may only be available with specific items sold, either a desktop or a notebook. Product taxonomy can help add an abstract level "Computers" to facilitate this kind of rule mining.

6.3 Mining Cross Level Rules

Rather than rules at a single level, people may also be interested in association of one category of products with another specific product. For instance, association between flat panel televisions and Nintendo Wii. We may not care whether the television is a Sony or a Panasonic, as long as it is a flat panel.

6.4 Removing Redundant Rules

By processing the raw data at the lowest level, we may find rules like $\{Desktop, Notebook\} \rightarrow Mouse$. If we take advantage of the product taxonomy, we may also find a rule $Computers \rightarrow Mouse$. If both rules are found, it is natural to regard the one of them as redundant and has a motivation to remove such redundant rules from the results.
Normally, we would keep the higher level rule.