**Aim: Demonstration of FP growth algorithm on supermarket data.**

**Theory:**

The two primary drawbacks of the Apriori Algorithm are:

1. At each step, candidate sets have to be built.
2. To build the candidate sets, the algorithm has to repeatedly scan the database.

These two properties inevitably make the algorithm slower. To overcome these redundant steps, a new association-rule mining algorithm was developed named Frequent Pattern Growth Algorithm. It overcomes the disadvantages of the Apriori algorithm by storing all the transactions in a Trie Data Structure.

| | FP Growth | Apriori |
|---|---|---|
| Speed | Faster, runtime increases linearly with increase in number of itemsets | Slower, runtime increases exponentially with increase in number of itemsets |
| Memory | Small, storing the compact version of database | Large, all the candidates from self-joining are stored in the memory |
| Candidates | No candidate generation | Use self-joining for candidate generation |
| Frequent patterns | Pattern growth achieved by mining conditional FP trees. | Patterns selected from the candidates whose support is higher than minSup. |
| Scans | Only require two scans | Scan the database over and over again. |

**Code:**

```
setwd("G:\\CU_Work\\Sem 6\\Data mining FDP\\exp3")

library("arules")

data("Mushroom")

fprules <- fim4r(Mushroom, method = "fpgrowth", target = "rules", supp = 70, conf = 60)

fprules

inspect(fprules[1:5])

x <- as(fprules,"data.frame")

write.csv(x, file="mushroomrules.csv")
```