Available online at www.sciencedirect.com

SciVerse ScienceDirect

Physics Procedia 25 (2012) 2066 – 2071

Physics Procedia

2012 International Conference on Solid State Devices and Materials Science

Simulation System of Car Crash Test in C-NCAP Analysis Based on an Improved Apriori Algorithm*

LI Xiang

School of Computer Engineering, Huaiyin Institute of Technology, Huai'an 223003, China

Abstract

In order to analysis car crash test in C-NCAP, an improved algorithm is given based on Apriori algorithm in this paper. The new algorithm is implemented with vertical data layout, breadth first searching, and intersecting. It takes advantage of the efficiency of vertical data layout and intersecting, and prunes candidate frequent item sets like Apriori. Finally, the new algorithm is applied in simulation of car crash test analysis system. The result shows that the relations will affect the C-NCAP test results, and it can provide a reference for the automotive design.

Keywords: Apriori algorithm; car crash; association rule

1. Introduction

Apriori algorithm[1-2] is data mining association rules algorithm, put forward by Imielinski, Agrawal and Swami. Apriori algorithm is a kind of data mining algorithm which is based on the horizontal data representation and breadth first search .This paper goes into Apriori algorithm through the analysis of car crash data, puts forward a new algorithm that adopts intersecting count to implement breadth first searching on database of vertical data layout. The new algorithm takes advantage of the efficiency of vertical data layout and intersecting, and prunes candidate frequent item sets like Apriori. It is applied in emulation analysis of variety of cars crash data, the result can show the relations that affect crash test among related cars, and it can provide a reference for the automotive design.

2. Procedure for Paper Submission

* This work was supported by the Natural Science Foundation of Huai'an Technology under Grant HAG2010068 .

 doi:10.1016/j.phpro.2012.03.351

2.1 Association Rules[3-4]

Definition 1: Let $I=\{ i_1 , i_2 ,\cdots, i_m \}$ be a collection. D is a transactional database. D is the collection of transaction T, and T can be seen as the collection of item, and $T \subseteq I$; TD is the only identifier of transaction T in transactional database. A is a collection, if $A \subseteq T$, and then we say T includes A. An association rule is a formula as $A \Rightarrow B$, $A \subseteq I$, $B \subseteq I$, and $A \cap B = \Phi$.

Conditions fill the association rule $A \Rightarrow B$:

a) With Support $S\%$. $S\%$=support $(A \Rightarrow B) = P(A \cup B)$, namely the percentage of transactions in D that contain $A \cup B$ at least is $S\%$.

b) With Confidence $C\%$. $C\%$ = confidence $(A \Rightarrow B) = P(B|A) = P(A \cup B) / P(A)$, namely the percentage of transactions in D contain A that also contain B at least is $C\%$.

Support is the percentage of transactions in database. Confidence is the intensity of the rule. Itemsets that satisfy minimum support (min_sup) is called frequent itemsets. Rules that satisfy both minimum support (min_sup) and minimum confidence (min_conf) are called strong association rules, usually noted $A \Rightarrow B$ $(S\%, C\%)$.

2.2 Apriori Algorithm

Apriori Algorithm is a basic algorithm that mining generates Boolean association rules needs frequent itemsets. Apriori algorithm mainly uses a circular system to rake though one gradation in turn to complete frequent itemsets mining. The main idea of this circular system is to generate $(k+1)$- itemsets from the k- the frequent itemsets. Detailed procedure: First, find frequent 1- itemsets, called L_1; then, use L_1 to mine L_2, namely frequent 2- itemsets; the circular system terminates when it can't find more frequent k- itemsets. Mine one gradation L_k needs traversing the whole database once.

Apriori Algorithm can be described as follows:

Input: data set D, minimum support min_sup

Output: frequent itemsets L in D

Steps:

L_1 = find_frequent_1 - itemsets (D) ;
For ($k = 2$; $L_{k-1} != \Phi$; $k++$)
{C_k = apriori_gen (L_{k-1}) ;
For each transaction $t \in D${
$C_t = subset(C_k , t)$
For each candidates $c \in C_t$
c. count + + ; }
$L_k = \{ c \in C \mid c.count \geq min\_sup \}$}
$L = L \cup L_k$;

Function apriori_gen
Input: L_{k-1} frequent$(k - 1)$_itemsets
Output: candidate itemsets C_k
Steps:
for each itemset $l_1 \in L_{k-1}$
for each itemset $l_2 \in L_{k-1}$
if $(l_1[1] = l_2[1]) \wedge l_1[2] = l_2[2]) \wedge \cdots \wedge (l_1[k-1] \neq l_2[k-1])$ {
$c = l_1 \infty l_2$ // generate candidate itemsets
if has infrequent subset (c, L_{k-1}) then

delete c // delete un-frequent itemsets
else add c to C_k }
return C_k

3. Improved Apriori Algorithm

Through research we can discover that Apriori algorithm has important features as follows:

a) It needn't to judge all k *(k-1)* dimensional subsets when judging if the *(k-1)*- dimensional subsets of *k*-dimensional candidate itemsets R are included in L_{k-1}. It just needs to judge the other k-2 subsets because the two subsets, $\{ R[l] \mid 0 \leqslant l \leqslant k-2 \}$ and $\{ R[l] \mid 0 \leqslant l \leqslant k-1, l \neq k-2 \}$, which connect into candidate itemsets R are included in L_{k-1}.

b) If every transaction T and itemsets in transaction are sequenced in dictionary order, once two *(k-1)*-dimensional frequent itemsets I_x and I_y can't be connected, then all the itemsets after I_x and I_y not satisfy connection conditions, do not have to test the connection to determine.

By comprehensive application of these two important features, we can propose improved Apriori algorithm implemented with vertical data sets to perform breadth first searching and intersecting. Store itemset and item according to the dictionary order. That is, first dig out all the frequent k-itemset and record them, then (k-1)-same as the previous item of itemset connect to a candidate (k+1)-itemsets, when connecting use the important property 2 to reduce the number of judgments; use the important property 1 to pruning judge on candidate itemsets, if the candidate itemset does not meet the properties of Apriori ,then cut; if meets, then directly count support with intersecting.

```
// use breadth first searching
for ( k = 2; L_{k-1} ≠ Φ; k + + ) {
for all X_i ∈L_{k-1} {
   for all X_j ∈L_{k-1} (j > i) {
   if (Link Item (X_i , X_j, k - 1) ) {
R =X_i ∪X_j ;
   if ( not Trimed (R, L_{k-1} ) ) {
Tidset (R) = Tidset (X_i ) ∩ Tidset (X_j ) ;
if ( Tidset (R) ⩾minsupp) {
L =L ∪R;
L_k =L_k ∪R; }
                    }
                } else
                  break;
          }
   }
}
```

//use Link Item ( ) function to determine if X_i and X_j can be connected to a candidate (k+1)-dimensional itemset. As long as their forward (k-1)- items are all the same, k-th-item of the former is less than the latter, then X_i and X_j can be connected to a (k+1)-dimensional candidate itemset. Specific function as follows:

```
bool Link Item (X_i , X_j , k) {
for (i= 0; l < k - 1; i+ + )
   if (X_i [l] ≠X_j[ l] )
         return false;
if (X_i [k - 1]< X_j [k - 1])
```

```
            return true;
    else
       return false;
    }
```

//use Trimed ( ) function to determine if *k-(k-1)*-dimensional subsets of k-dimensional itemset R are all in L_{k-1}. If one *(k-1)*-dimensional subset is not in L_{k-1}, the candidate itemset R can't be frequent, then return true. Reduce number of counts k to *(k-2)*, specific function as follows:

```
bool Trimed (R, L_{k-1} ) {
 for each itemset r_1 ∈ R
 for each itemset l_1 ∈ L_{k-1} {
    if (r_1 ∉ L_{k-1} )
                return true;
 }
 return false;
 }
```

Experiment comparison of Apriori algorithm and improved Apriori algorithm. Hardware environment of experiment is Intel Core2 Duo T7350 , 3G DDR3 memory, software environment is Visual C++6.0, the experiment data acquisition uses T10 I4D100k, from Frequent ItemsetMining Implementations Repository (http://fimi. cs. helsinki. fidata). Time of the two algorithms are showed in Fig.1.

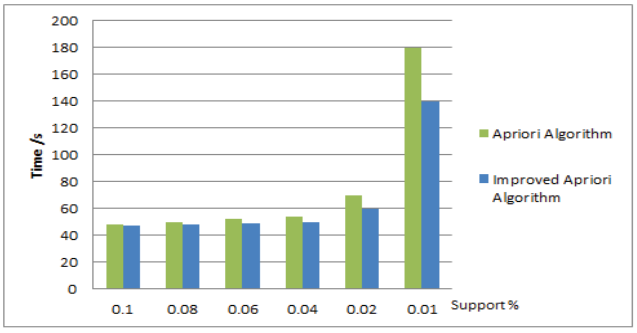

Fig.1. Time of the two algorithms

As we can see from the Figure, improved algorithm operating time is less than the original algorithm, and with the reduction of minimum support and the need for the number of frequent itemsets mining increased advantage of the improved Apriori algorithm is very obvious.

4. Simulation

C-NCAP (China-New Car Assessment Program) is vehicle collision safety performance test according to European NCAP standard, evaluation results divided by stars and publicly announced. C-NCAP requires three kinds of collision tests on one model, vehicle speed 50km/h and the rigid fixed barrier 100% overlapping rate frontal collision, vehicle speed 56km / h on the deformable barrier 40% overlap offset frontal collision, moving deformable barrier speed of 50km/h with the vehicle's side collision. Calculate scores in every test and total score according to the test data, then determine stars by total score. Scoring rules are very careful and rigid .The highest score is 51 points, the lowest star is one star, the highest is five stars.

Fig.2. Schematic Diagram Of C-NCAP Crash Test Program

Frontal 100% collision perfect score is 16 points, frontal 40% offset collison perfect score is 16 points, side collision perfect score is 16 points.

This simulation experiment primary data gathering in Sina Net automobile channel C-NCAP collision database http://auto.sina.com.cn/z/cncapsinahz/index.shtml

Randomly select following several models for data analysis: Passat, Buick LaCrosse, Peugeot 307, Nissan Tida, Suzuki Swift, BYD F3, Volkswagen Sagitar, Chery A5, Geely Merrie, Great Wall Hover, crash test data as Table I.

TABLE I C-NCAP vehicle crash analysis BASE data

Car Models	Frontal 100% Collision	Frontal 40% Offset Collision	Side Collision
Passat	12.32	14.12	12.82
Buick LaCrosse	13.55	14.92	10.72
Peugeot 307	14.4	16	12.61
Nissan Tiida	13.45	13.89	13.5
Suzuki Swift	11.98	14.19	15.09
BYD F3	10.02	13.44	10.52
Volkswagen Sagitar	12.59	15.03	16
Chery A5	13.05	12.84	10.67
Geely Merrie	9.51	11.64	7.49
Great Wall Hover	10.07	12.29	14.77
…	…	…	…

Experiments carried out association analysis of car crash score when mining required a logical type data. Therefore turn the data of Table I into Boolean. Due to finding good relations between several collisions, so the filed of monomial points up to 12 points is set “TRUE”, it shows this item is in the transaction. The others set “FALSE”, it shows this item isn’t in the transaction. Turn Table I into the style which is easy to be handled with association rules, in Table 2.

TABLE 2 C-NCAP vehicle crash analysis logic data

Car Models	Frontal 100% Collision	Frontal 40% Offset Collision	Side Collision
Passat	TRUE	TRUE	TRUE
Buick LaCrosse	TRUE	TRUE	FALSE
Peugeot 307	TRUE	TRUE	TRUE
Nissan Tiida	TRUE	TRUE	TRUE
Suzuki Swift	FALSE	TRUE	TRUE
BYD F3	FALSE	TRUE	FALSE
Volkswagen Sagitar	TRUE	TRUE	TRUE
Chery A5	TRUE	TRUE	FALSE
Geely Merrie	FALSE	FALSE	FALSE
Great Wall Hover	FALSE	TRUE	TRUE
…	…	…	…

To mine association rules in good courses in Table II, this experiment in Table II set the minimum support is 20%, the confidence is 50%. The 72 records in Table II as Apriori mining association rules data, using the improved Apriori algorithm, obtain results in Table III.

Table III Frequent item set table

Frequent Itemsets	Support Count
{Frontal 100%, Frontal 40%}	65
{Frontal 40%, Side}	32
{Frontal100%}	76
{Frontal40%}	93
{Side}	43

Count the confidence of non-empty subset in final frequent itemset and delete the records which are less than the minimum confidence. Eventually produce association rules as follows:

a) Frontal 100% >=frontal 40%, confidence=65/76 ≈ 85.52%;

b) Frontal 40% >=frontal 100%, confidence=65/93 ≈ 69.89%;

c) Side>=frontal 40%, confidence=32/43 ≈ 74.41%.

From the above run results can obtain the following potential association:

a) When frontal 100% overlap the rigid barrier crash test is good, frontal 40% overlap the deformable barrier crash test has 85.52% good possibilities.

b) When frontal 40% overlap the deformable barrier crash test is good, moving deformable barrier side crash test has 69.89% good possibilities.

c) When moving deformable barrier side crash test is good, frontal 40% overlap the deformable barrier crash test has 74.41% good possibilities.

5. Conclusions

This paper aims for improving the efficiency of mining frequent itemsets when using mining association rules algorithm, puts forward an improved Apriori algorithm. And the algorithm is applied to the analysis of car crash tests; simulation results show the affective relationships among of car crash tests, to provide reference for automotive manufacture's design and consumers to buy cars.

References

[1] Agrawa1 R , Imielinski T , SwamiA . M ining association rules between sets of item s in large databases [C] / /Proc ofACM SIGMOD Conference on Management of Data, ACM New York, NY, USA, 1993: 2072216.

[2] Agrawal R, Srikant R. Fast algorithm form ining association rules in large databases [C] / /Proceedings of the 20 th International Conference onVery Large Data Bases, Morgan Kaufmann Publishers Inc San Francisco, CA, USA, 1994: 4872499.

[3] M J Zaki, S Parthasarathy, M Ogihara andW Li. New algorithms for fast discovery of association rules[C]. In Proc. of the 3 rd Intpl Conf. on KDD and DataMining (KDD'97) , NewportBeach, Cali2 fornia, August 1997.

[4] Han J W, Micheline K. Data Mining Concepts and Techniques [M]. Fan Ming, Meng Xiaofeng,Translated. Beijing: Machinery Industry Press, 2002.