ASSOCIATION RULE

Rachmat Selamet

Sekolah Tinggi Manajemen Informatika dan Komputer LIKMI
Jl. Ir. H. Juanda 96 Bandung 40132

E-mail : if25005@students.itb.ac.id

Abstrak

Data mining adalah mencari informasi yang tersembunyi di dalam database dan dapat dilihat sebagai langkah dari proses pencarian knowledge. Fungsi-fungsi yang terdapat dalam data mining adalah clustering, classification, prediction dan association. Association rule mengidentifikasi hubungan dari sekumpulan item yang ada di database. Untuk dapat memperoleh association rule, ada beberapa algoritma yang dapat digunakan, antara lain Algoritma Single Dimensional (terdiri dari Algoritma sekuensial, Algoritma paralel) dan Paralel lain.

Kata-kata kunci: Algoritma, paralel

1. PENDAHULUAN

Association rule, ditemukan pertama pada tahun 1993, yang mengidentifikasi hubungan dari sekumpulan item yang ada di database. Hubungan ini tidak berdasarkan properti dari data tersebut, tetapi didasarkan dari peristiwa dari item data. Contoh berikut menggambarkan association rule dan pemakaiannya.

Contoh : Sebuah toko mencoba mencari hubungan suatu barang yang dibeli dengan barang lainnya, contohnya selai kacang. Mereka menemukan selai kacang 30% dibeli bersama roti tawar dan 40% dibeli bersama agar-agar. Berdasarkan fakta ini, maka toko tersebut menempatkan selai kacang berdekatan dengan roti dan agar-agar. Dengan adanya penempatan ini, maka akan meningkatkan penjualan barang yang dibeli bersamaan.

Pada contoh di atas terdapat 2 association rule untuk barang selai kacang, yaitu roti yang dibeli 30% dan agar-agar yang dibeli 40%. Association rule biasa digunakan untuk menganalisa transaksi yang terjadi di pasar. Untuk dapat memperoleh association

rule, ada beberapa algoritma yang dapat digunakan. Algoritma ini akan dibahas pada bagian selanjutnya.

Dari algoritma yang akan dibahas, akan dilihat bagaimana performance dari algoritma tersebut. Untuk ukuran standar biasa digunakan algoritma apriori sebagai standar.

2. ALGORITMA ASSOCIATION RULE

Algoritma association rule digunakan untuk menghasilkan association rule. Berdasarkan dimensinya, algoritma yang digunakan dapat digunakan untuk single dimensional (1 dimensi) dan multi dimensional (2 atau lebih).

2.1.1. ALGORITMA SINGLE DIMENSIONAL

Dilihat dari prosesnya, algoritma untuk association rule ini bisa dijalankan sekuensial atau paralel. Pada bagian berikut akan dibahas 1 algoritma sekuensial dan 1 algoritma paralel.

2.1.1 ALGORITMA SEKUENSIAL

Ada banyak algoritma sekuensial yang sudah dikembangkan, yaitu : AIS, SETM, Apriori, Apriori-TID, Apriori-Hybrid, Off-line Candidate Determination (OCD), Partitioning, Sampling, Dynamic Itemset Counting (DIC), Continuous Association Rule Mining Algorithm (CARMA), FP-Growth, Charm, MagnumOpus, dan lain-lain. Yang akan dibahas adalah CARMA.

CARMA (Continuous Association Rule Mining Algorithm) membawa perhitungan sejumlah besar itemset online. online, artinya CARMA mengizinkan pemakai untuk mengubah parameter, support minimum dan confidence minimum pada beberapa transaksi selama scan pertama dari database. Jadi prosesnya memerlukan paling banyak 2 scan database. Mirip dengan DIC, CARMA menghasilkan itemset pada scan pertama dan selesai menghitung semua itemset pada scan kedua. Perbedaan dengan DIC adalah CARMA menghasilkan itemset langsung dari transaksi. Setelah membaca setiap transaksi, pertama dihitung jumlah dari itemset yang merupakan subset dari transaksi. Kemudian menghasilkan itemset baru dari transaksi, jika semua subset dari itemset langsung sudah mendekati besar dengan melihat nilai dari support minimum dan bagian

dari database yang dibaca. Selanjutnya akurasi prediksi dari apakah sebuah itemset kemungkinan besar, dengan menghitung upperbound untuk jumlah dari itemset, yang menjumlahkan jumlah sekarang dan nilai perkiraan dari sejumlah yang muncul sebelum itemset dihasilkan. Perkiraan dari sejumlah yang muncul (disebut juga maximum misses) dihitung ketika itemset pertama kali dihasilkan. Beberapa hal yang dapat dihasilkan oleh CARMA :

Umpan balik terus menerus : CARMA terus menerus menghasilkan association rule, ketika daftar penjualan discan. Untuk setiap rule dijaga dengan diciutkan, penetapan interval untuk support dan confidencenya.

Pengaturan oleh pemakai : Selama scan pertama, pemakai bebas dapat mengubah nilai dari support dan confidence langsung. Dengan kombinasi dari umpan balik terus menerus, pemakai dapat menetapkan nilai yang benar interaktif.

Penetapan dan akurasi dari hasil : CARMA menjamin akan memproduksi semua association rule setelah paling banyak 2 scan dan untuk setiap rule akan menghasilkan nilai support dan confidence yang benar.

Berikut ini algoritma dari CARMA yang dibagi menjadi 3 fungsi, yaitu Carma, PhaseI dan PhaseII dengan fungsi utamanya Carma.

```
Function Carma   (   transaction sequence T,
                     support sequence σ)
                 :   support lattice;
support lattice V;
begin
 V := PhaseI( T, σ );
 V := PhaseII( V, T, σ );
 return V;
end;
```

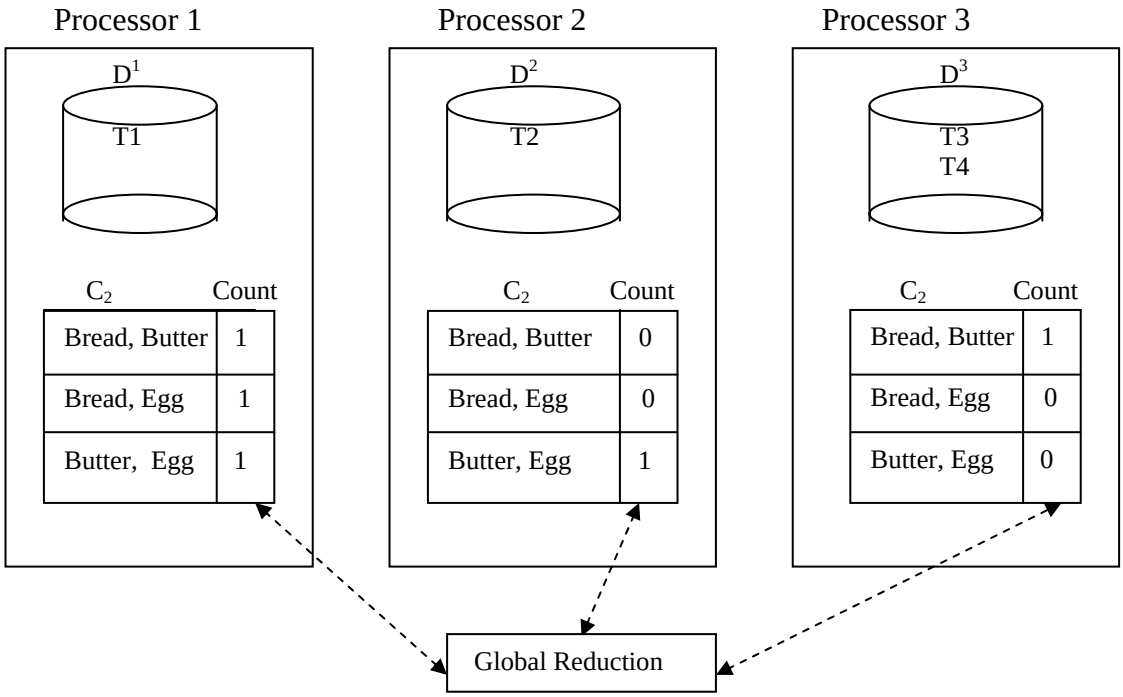
```
Function PhaseI(   transaction sequence (t_1, ..., t_n),
                   support sequence σ = (σ_1, ..., σ_n)
                   ) : support lattice;
support lattice V;
begin
 V := {∅};
 maxMissed(v) := 0, firstTrans(v) := 0
count(v) := 0;
 for i from 1 to n do
  // 1) Increment
  for all v ∈ V with v ⊆ t_i do count(v) + +; od;
  // 2) Insert
  for all v ⊆ t_i with v ∉ V do
   if ∀w ⊂ v : w ∈ V and maxSupport(w) ≥ σ_i then
    V := V ∪ {v};
    firstTrans(v) := i;
    count(v) := 1;
    maxMissed(v) :=
       min{   ⌊(i − 1)avg_{i−1}(⌈σ⌉_{i−1})⌋ + |v| − 1,
              maxMissed(w) + count(w) − 1 | w ⊂ v };
    if |v| == 1 then maxMissed(v) := 0; fi;
   fi;
  od;
  // 3) Prune
  if ( i % max{⌈1/σ_i⌉, 500} ) == 0 then
   V := {v ∈ V | maxSupport(v) ≥ σ_i or |v| == 1 };
  fi;
 od;
 return V;
end;
```

```
Function PhaseII   (   support lattice V,
                       transaction sequence (t_1, ..., t_n),
                       support sequence σ )
                   :   support lattice;
integer ft, i = 0;
begin
 V := V\{v ∈ V | maxSupport(v) < σ_n };
 while ∃v ∈ V : i < firstTrans(v) do
  i++;
  for all v ∈ V do
   ft := firstTrans(v);
   if v ⊆ t_i and ft < i then
    count(v)++, maxMissed(v)−;
   fi;
   if ft == i then
    maxMissed(v) := 0;
    for all w ∈ V:
       v ⊂ w and maxSupport(w) > maxSupport(v)
do
      maxMissed(w) := count(v) − count(w);
    od;
   fi;
   if maxSupport(v) < σ_n then V := V\{v}; fi;
 od; od;
 return V;
end;
```

2.1.2. ALGORITMA PARALEL

Untuk algoritma paralel dan distribusi berbasis pada algoritma apriori. Pada dasarnya, algoritma paralel dapat dibagi dengan 2 cara, yaitu paralel data dan paralel tugas. Perbedaannya ada pada candidate set yang didistribusikan menyeberang processor atau tidak. Pada paralel data, setiap node menghitung sekumpulan candidate yang sama. Sedangkan pada paralel tugas, candidate dipartisi dan didistribusi menyeberang processor, sehingga setiap node menghitung sekumpulan candidate yang berbeda. Selain dari 2 paralel tersebut ada juga algoritma paralel lain yang tidak dikelompokkan. Algoritma tersebut dimasukkan ke kelompok paralel lain.

2.1.2.1.PARALEL DATA



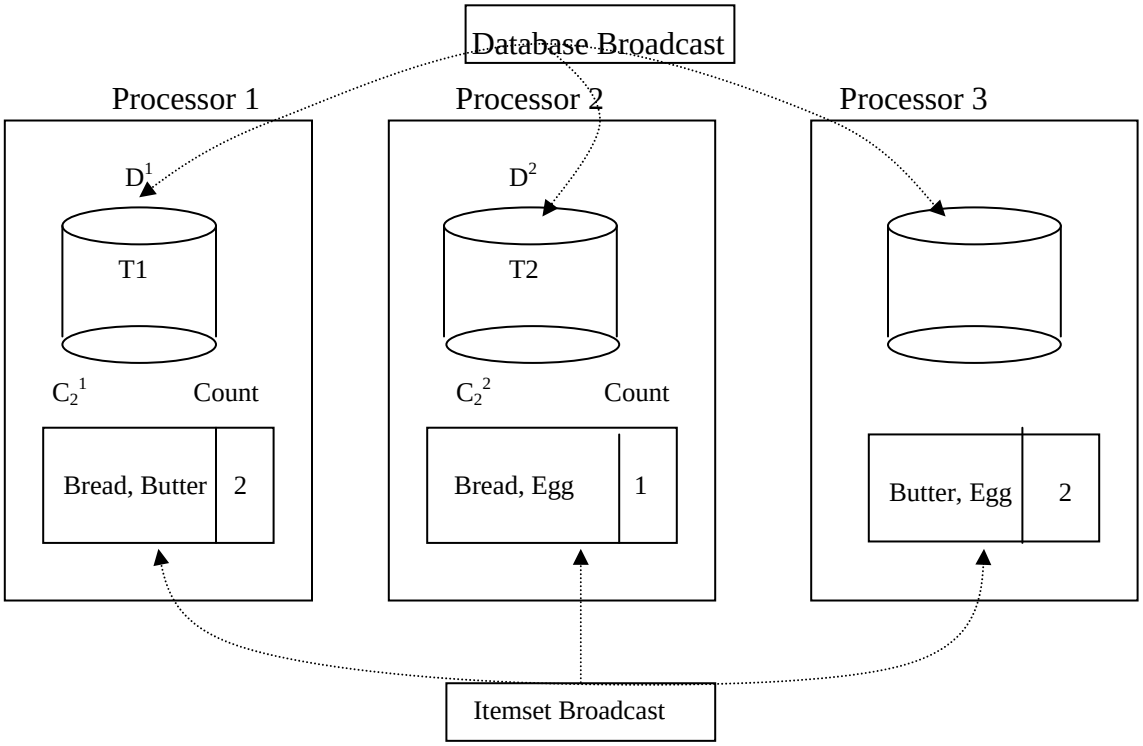
Paralel data

Algoritma yang dapat digunakan untuk paralel data adalah Count Distribution (CD), Parallel Data Mining (PDM), Distributed Mining Algorithm (DMA) dan Common Candidate Partitioned Database (CCPD). Yang akan dibahas adalah Count Distribution(CD).

Algorithm CD
Input:
I, s, D^1, D^2, …, D^p
Output:
L
Algorithm:
1) C_1=I;
2) **for** k=1;$C_k \neq \varnothing$;k++ **do begin**
//step one: counting to get the local counts
3) count(C_k, D^j); //local processor is *i*
//step two: exchanging the local counts with other processors
//to obtain the global counts in the whole database.
4) **forall** itemset $X \in C_k$ **do begin**
5) X.count=$\sum_{j=1}^{p}\{X^j.count\}$;
6) **end**
//step three: identifying the large itemsets and
//generating the candidates of size k+1
7) $L_k=\{c \in C_k \mid c.count \geq s \times \mid D^1 \cup D^2 \cup \ldots \cup D^p \mid\}$;

8) C_{k+1}=apriori_gen(L_k);
9) **end**
10) return L=L_1 Y L_2 Y … Y L_k;

2.1.2.2.PARALEL TUGAS



Paralel tugas

Algoritma yang dapat digunakan untuk paralel tugas adalah Data Distribution (DD), Intelligent Data Distribution (IDD), Hash based Parallel mining for Association rules (HPA) dan Parallel Association Rule (PAR). Algoritma yang akan dibahas adalah DD.

Algorithm DD
Input:
I,s,D^1, D^2, …, D^p
Output:
L
Algorithm:
1) $C_1^i \subseteq I$;
2) **for** (k=1;$C_k^i \neq \varnothing$;k++) **do begin**
//step one: counting to get the local counts

3) count(C_k^i , D^i); //local processor is *i*
//step two: broadcast the local database partition to others,
// receive the remote database partitions from others,
// scan D^j($1 \le j \le p$, $j \ne i$) to get the global counts.
4) broadcast(D^i);
5) **for** ($j=1$; ($j \le p$ and $j \ne i$);j++) **do begin**
6) receive(D^j) from processor j;
7) count(C_k^i , D^j);
8) **end**
//step three: identify the large itemsets in C^i_k,
// exchange with other processors to get all large itemsets C_k,
// generate the candidates of size k+1,
// partition the candidates and distribute over all processors.
9) $L_k^i = \{c | c \in C^i_k, c.count \ge s * |D^1 \cup D^2 \cup \ldots \cup D^p|\}$;
10) $L_k = Y_{i=1}^{p}(L_k^i)$;
11) $C_{k+1} = apriori\_gen(L_k)$;
12) $C_{k+1}^i \subseteq C_{k+1}$; //partition the candidate itemsets across the processors
13) **end**
14) return $L = L_1 Y L_2 Y \ldots Y L_k$;

2.1.2.3.PARALEL LAIN

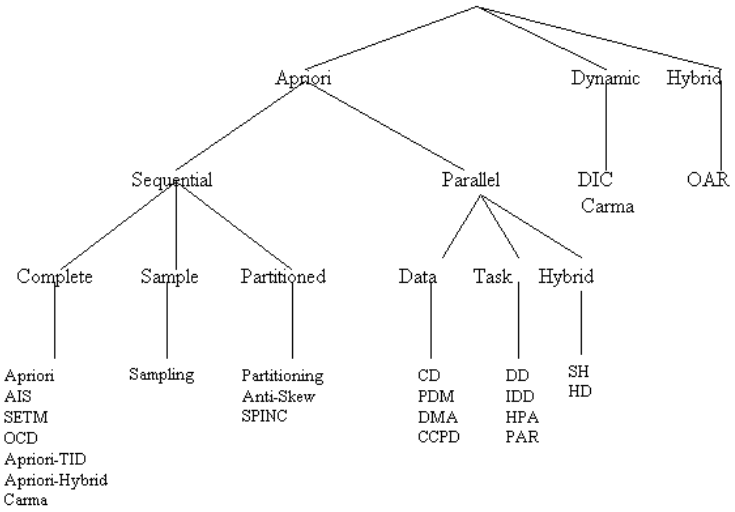
Ada juga algoritma lain yang tidak masuk dalam paralel data dan paralel tugas, karena perbedaan fitur. Yang masuk dalam kelompok ini adalah Candidate Distribution, SH(Skew Handling) dan HD(Hybrid Distribution).

2.2 ALGORITMA MULTI DIMENSIONAL

Selain algoritma yang digunakan untuk 1 dimensi, ada juga algoritma yang digunakan untuk multi dimensi. Algoritma untuk multi dimensi umumnya merupakan pengembangan dari algoritma apriori. Ada beberapa cara yang umum digunakan, yaitu dengan algoritma top-down, algoritma bottom-up dan menggunakan rumus statistik. Yang akan dibahas adalah algoritma top-down.

3. PERBANDINGAN ALGORITMA

Untuk membandingkan algoritma dapat juga dibagi berdasarkan prosesnya seperti gambar berikut ini :

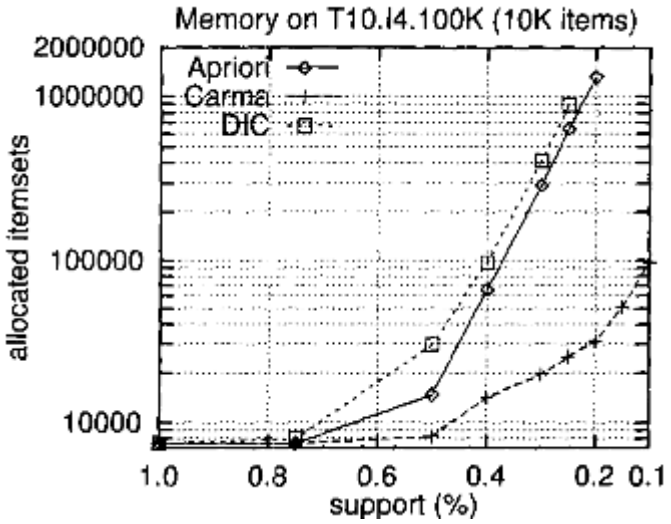


Berikut ini adalah table perbandingan algortima dengan melihat jumlah maksimal scan, struktur data yang digunakan dan komentar.

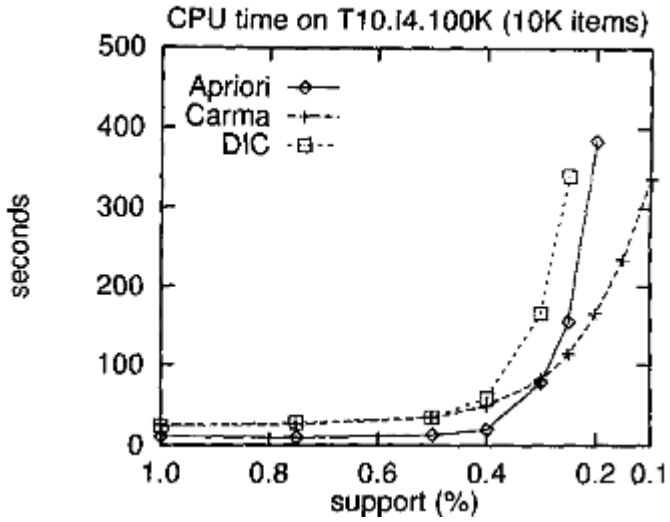
Algorithm	**Scan**	**Data structure**	**Comments**
AIS	m+1	Not Specified	Suitable for low cardinality sparse transaction database; Single consequent
SETM	m+1	Not Specified	SQL compatible
Apriori	m+1	L_{k-1} : Hash table C_k: Hash tree	Transaction database with moderate cardinality; Outperforms both AIS and SETM; Base algorithm for parallel algorithms
Apriori-TID	m+1	L_{k-1} : Hash table C_k: array indexed by TID $\overline{C_k}$: Sequential structure ID: bitmap	Very slow with larger number of $\overline{C_k}$; Outperforms Apriori with smaller number of $\overline{C_k}$;
Apriori-Hybrid	m+1	L_{k-1} : Hash table 1st Phase: C_k: Hash tree 2^{nd} phase: C_k: array indexed by IDs $\overline{C_k}$: Sequential structure ID: bitmap	Better than Apriori. However, switching from Apriori to Apriori-TID is expensive; Very crucial to figure out the transition point.
OCD	2	Not specified	Applicable in large DB with lower support threshold.
Partition	2	Hash Table	Suitable for large DB with high cardinality of data; Favors homogenous data distribution
Sampling	2	Not Specified	Applicable in very large DB with lower support.

DIC	Depe nds on interv al size	Trie	Database viewed as intervals of transactions; Candidates of increased size are generated at the end of an interval
CARMA	2	Hash Table	Applicable where transaction sequences are read from a Network; Online, users get continuous feedback and change support and/or confidence any time during process.
CD	m+1	Hash table and tree	Data Parallelism.
PDM	m+1	Hash table and tree	Data Parallelism; with early candidate pruning
DMA	m+1	Hash table and tree	Data Parallelism; with candidate pruning
CCPD	m+1	Hash table and tree	Data Parallelism; on shared-memory machine
DD	m+1	Hash table and tree	Task Parallelism; round- robin partition
IDD	m+1	Hash table and tree	Task Parallelism; partition by the first items
HPA	m+1	Hash table and tree	Task Parallelism; partition by hash function
SH	m+1	Hash table and tree	Data Parallelism; candidates generated independently by each processor.
HD	m+1	Hash table and tree	Hybrid data and task parallelism; grid parallel architecture

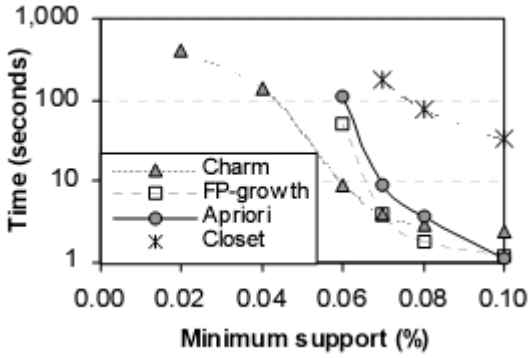
Berikut ini adalah grafik perbandingan beberapa algoritma dilihat dari performancenya (waktu atau resource).

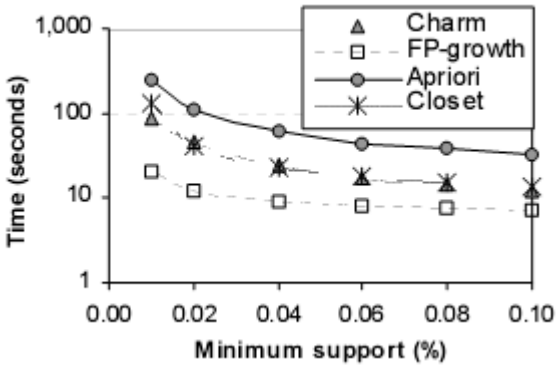


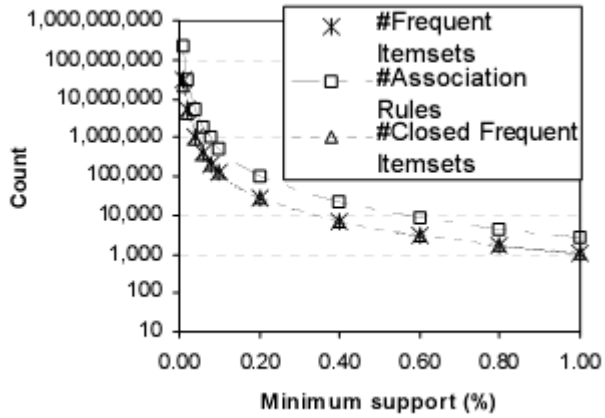
Grafik perbandingan algoritma carma pemakaian memory (y adalah jumlah scanning itemset, x adalah threshold)

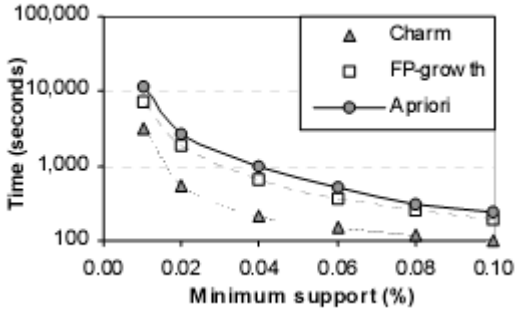


Grafik perbandingan algoritma carma waktu pemakaian (y adalah waktu yang digunakan, x adalah threshold)









	High Min-Support	**Low Min-Support**
IBM-Artificial	Ap > FP > Ch > Cl	FP > Ch > Cl > Ap
BMS-POS	Ap > Cl > FP > Ch	Ch > FP > Ap > Cl
BMS-WebView-1	Ap > FP > Cl > Ch	Ch > FP > Ap > Cl
BMS-WebView-2	Ap > FP > Ch > Cl	Ch > FP > Ap > Cl

4. KESIMPULAN

Data mining adalah mencari informasi yang tersembunyi di dalam database dan dapat dilihat sebagai langkah dari proses pencarian knowledge. Fungsi-fungsi yang terdapat dalam data mining adalah clustering, classification, prediction dan association. Salah satu yang terpenting adalah Association rule. Association rule, ditemukan pertama pada tahun 1993, yang mengidentifikasi hubungan dari sekumpulan item yang ada di database. Hubungan ini tidak berdasarkan properti dari data tersebut, tetapi didasarkan

dari peristiwa dari item data. Dalam kehidupan sehari-hari, association rule biasa digunakan untuk menganalisis data.

Algoritma association rule digunakan untuk menghasilkan association rule. Berdasarkan dimensinya, algoritma yang digunakan dapat digunakan untuk single dimensional (1 dimensi) dan multi dimensional (2 atau lebih).

Untuk 1 dimensi terdapat 2 macam algoritma, yaitu sekuensial dan paralel. Contoh algoritma sekuensial adalah AIS, SETM, Apriori, Apriori-TID, Apriori-Hybrid, Off-line Candidate Determination (OCD), Partitioning, Sampling, Dynamic Itemset Counting (DIC), Continuous Association Rule Mining Algorithm (CARMA), FP-Growth, Charm, MagnumOpus, dan lain-lain.

Untuk algoritma paralel terdapat 3 macam, yaitu : Paralel data, paralel tugas dan paralel lain. Algoritma yang dapat digunakan untuk paralel data adalah Count Distribution (CD), Parallel Data Mining (PDM), Distributed Mining Algorithm (DMA) dan Common Candidate Partitioned Database (CCPD). Algoritma yang dapat digunakan untuk paralel tugas adalah Data Distribution (DD), Intelligent Data Distribution (IDD), Hash based Parallel mining for Association rules (HPA) dan Parallel Association Rule (PAR). Yang masuk dalam algoritma lain adalah Candidate Distribution, SH(Skew Handling) dan HD(Hybrid Distribution).

Ada beberapa cara yang umum digunakan untuk algoritma multi dimensional, yaitu dengan algoritma top-down, algoritma bottom-up dan menggunakan rumus statistik.

Untuk perbandingan antar algoritma, banyak algoritma baru yang performancenya cukup baik tetapi tidak dapat digunakan untuk kondisi tertentu. Sedangkan untuk standar (karena dapat digunakan untuk semua kondisi), digunakan algoritma apriori. Algoritma apriori ini yang banyak dikembangkan untuk algoritma lain seperti untuk kebutuhan multi dimensional, untuk paralel dan sebagainya.

5. DAFTAR PUSTAKA

1. Christian Hidber, *"Online Association Rule Mining"*, SIGMOD 1999, Proceedings ACM SIGMOD International Conference on Management of Data, June 1-3, 1999, Philadephia, Pennsylvania, pp.145-156.
2. M. Zaki and C. Hsiao. CHARM: *"An Efficient Algorithm for Closed Itemset Mining"*, 2nd SIAM International Conference on Data Mining, Arlington, April 2002.

3. bit.csc.lsu.edu/~xiaoz/ar.pdf
4. citeseer.ist.psu.edu/322705.html
5. control.cs.berkeley.edu/carma.html
6. engr.smu.edu/~mhd/pubs/00/ar.doc
7. pandora.compsci.ualr.edu/milanova/7399-11/week10/ar.doc
8. robotics.stanford.edu/users/ronnyk/realWorldAssoc.pdf
9. www.almaden.ibm.com/software/quest/Publications/papers/vldb94.pdf
10. www.csse.monash.edu.au/~webb/Files/WebbZhang01.pdf
11. www.iiit.ac.in/~vikram/publications/vikram_armor.pdf
12. www.cs.fiu.edu/~cyang01/cop5992/project3-2.htm