PEMBUATAN PERANGKAT LUNAK DATA MINING UNTUK PENGGALIAN KAIDAH ASOSIASI MENGGUNAKAN METODE APRIORI

Leo Willyanto Santoso
Fakultas Teknologi Industri, Jurusan Teknik Informatika, Universitas Kristen Petra
e-mail : leow@peter.petra. ac.id

ABSTRAK: Banyak teori dan pendekatan yang dikembangkan untuk memperoleh hasil penemuan kaidah asosiasi dan pola. Salah satu metode yang dikembangkan yaitu dengan menggunakan metode apriori. Beberapa dari metode sebelumnya melakukan pencarian itemset dengan pendekatan graf asosiasi yang memiliki kelemahan pada penggunaan memori yang besar. Keterbatasan memori jelas akan mempengaruhi banyaknya item yang bisa diproses. Lebih jauh lagi, sebagian besar pendekatan menggunakan struktur data internal sangat rumit yang tidak bersifat lokal dan membutuhkan tambahan sumber daya dan banyak komputasi.

Pada riset ini, metode apriori digunakan untuk memperoleh kaidah asosiasi yang menggambarkan hubungan antar item pada database transaksional. Database yang digunakan ada tiga buah yang masing-masing memiliki jumlah transaksi yang berbeda.

Dari hasil pengujian empiris dapat ditarik kesimpulan bahwa waktu komputasi untuk menghasilkan kaidah asosiasi dipengaruhi oleh jumlah transaksi dan Penggunaan struktur data "tidlist" pada algoritma apriori menyebabkan waktu komputasi yang dibutuhkan relatif berkurang karena hanya memerlukan pembacaan basis data sekali saja.

Kata kunci: Data Mining, Algoritma Apriori.

ABSTRACT: *There are many theories and approaches that have been developed to find pattern and association rule. One of the methods that have been developed is apriori method. Any method previously finds itemset using graph approaches that have weakness on huge memory usage. The lack of memory will affect many item that able to process. Further, many approaches use internal data structure that very complex and need resource addition to do this computation.*

In this research, apriori method was used to get association rule that describe relation between items in transactional database. The database that used is three kinds that have different in total transactional.

Based on the empirical test, can be concluded that computational time to get asociation rule is influenced by the number of transaction and using "tidlist" data structure in apriori methods can reduce time because only read database one time.

Keywords: *Data Mining, Apriori Algorithm.*

1. PENDAHULUAN

Ketersediaan data sudah bukan hal yang sulit diperoleh lagi dewasa ini apalagi ditunjang dengan banyaknya kegiatan yang sudah dilakukan secara komputerisasi. Namun data ini seringkali diperlakukan hanya sebagai rekaman tanpa pengolahan lebih lanjut sehingga tidak mempunyai nilai guna lebih untuk keperluan masa mendatang. Analisa dari tiap koleksi data tersebut akan menghasilkan pengetahuan atau informasi, misalnya berupa pola dan kaidah asosiasi yang terjadi pada data. Pola dan kaidah asosiasi bisa terjadi pada berbagai jenis data baik data ekonomi, keuangan, kesehatan dan lain-lain.

Penggalian kaidah asosiasi mempunyai peranan penting dalam proses pengambilan keputusan. Tahapan besar dari proses *Data Mining* adalah mengidentifikasikan *frequent* itemset dan membentuk kaidah asosiasi dari itemset tersebut. Kaidah asosiasi digunakan untuk menggambarkan hubungan antar item pada tabel data transaksional.

Tapi semakin berkembangnya teknologi komputer di dunia industri, semakin pesat pula perkembangan ukuran data tabel transaksional yang dihasilkan. Dan pada data tabel transaksional yang besar (*VLDB,*

Very Large Database) tersebut, proses pencarian *frequent* itemset sangatlah sulit. Dari kondisi tersebut, sudah banyak algoritma yang dibentuk untuk mencari kaidah asosiasi. Tetapi keterbatasan tetap saja ada. Keterbatasan yang paling mencolok adalah diperlukannya pembacaan basis data secara berulang yang mengurangi kinerja algoritma tersebut. Sehingga diperlukan suatu algoritma yang sangat efisien yang bisa meminimalisasi pembacaan basis data, sehingga bisa mengoptimasi waktu yang dibutuhkan.

Perangkat lunak yang dibuat ini menggunakan suatu algoritma yang menyimpan data tabel transaksional di memory pada pembacaan awal. Sehingga untuk proses selanjutnya pembacaan basis data dapat dikurangi. Sifat ini tentu saja menguntungkan algoritma tersebut.

2. DATA MINING

Akhir–akhir ini, kemampuan sistem komputer dalam menghasilkan dan mengumpulkan data meningkat dengan pesat. Terlihat dari semakin banyaknya komputerisasi pada setiap transaksi bisnis dan pemerintahan, dan tersedianya perangkat keras penyimpan basis data yang dapat menyimpan data yang sangat besar sekali. Berjuta–juta basis data dihasilkan pada manajemen bisnis, administrasi pemerintahan, dan pada banyak aplikasi lainnya. Pesatnya perkembangan ukuran basis data dapat disebabkan karena kemampuan dari sistem basis datanya. Kondisi ini menimbulkan kebutuhan baru yang penting, yaitu : teknik baru yang melakukan proses transformasi dari basis data transaksional yang besar tersebut untuk mendapatkan informasi penting yang dibutuhkan. Sehingga *Data Mining* menjadi bahan riset yang penting sekarang ini. Teknik baru yang akan diteliti pada penelitian ini, yang akan digunakan untuk menggali kaidah asosiasi adalah metode Apriori.

Metode Apriori yang akan digunakan ini mempunyai tujuan utama adalah untuk mencari maksimal *frequent* itemset (didapatkan juga *frequent* itemset yang tidak maksimal). Metode Apriori yang akan digunakan pada penelitian ini, mempunyai beberapa kelebihan:

a. Menggunakan format data *vertical tid-list* yang mengasosiasikan itemset dengan transaksi yang terjadi.
b. Menggunakan pendekatan apriori untuk mencari maksimal *frequent* itemset.
c. Membutuhkan hanya sedikit pembacaan database, dan meminimalkan penggunaan I/O.

Untuk beberapa simbol, ada beberapa yang harus diketahui [AGR-1993] : *I* adalah himpunan item, *D* adalah data transaksi, dimana setiap transaksi mempunyai ID unik (*tid*) dan terdiri dari beberapa item. Himpunan item disebut i*temset.* Sebuah itemset yang terdiri dari k item disebut *k-itemset*. *Support* dari itemset X, dinotasikan sebagai $\sigma(X)$, adalah jumlah transaksi dimana X berada sebagai subset. Sebuah subset dari itemset yang mempunyai panjang k disebut *k-subset*. Itemset disebut *maksimal* bila bukan merupakan subset dari itemset lainnya. Itemset disebut *frequent* bila support-nya sama atau lebih besar dari *support* minimum *(minsup)* seperti yang telah ditentukan sebelumnya. Sebuah *frequent* k- itemset dinotasikan sebagai F_k. Kaidah asosiasi merupakan ekspresi $A \rightarrow B$, dengan A dan B adalah itemset.

$$Support = \sigma(A \cup B) \quad (1)$$

$$Confidence = \sigma(A \cup B)/\sigma(A) \quad (2)$$

Confidence adalah nilai probabilitas adanya itemset A pada suatu transaksi, maka juga ada itemset B pada transaksi tersebut. Kaidah disebut *confidence* bila nilai *confidence*-nya sama atau lebih besar dari *comfidence* minimum *(minconf)* seperti yang telah ditentukan sebelumnya.

Tugas dari *Data Mining* adalah untuk menghasilkan semua kaidah asosiasi pada suatu tabel transaksional, yang mempunyai nilai *support* lebih dari minsup. Kaidah tersebut juga harus mempunyai *confidence* yang lebih besar dari minconf. Tugas tersebut dapat diselesaikan dengan melakukan 2 tahap penting [AGR-1996], yaitu:

- Mencari semua *frequent* itemset, dan
- Mencari aturan asosiasi yang confidence.

Contoh yang bisa digunakan adalah basis data penjualan buku. Ada 5 buku berbeda, yang dianggap sebagai 5 item. Masing-masing item dinotasikan dengan sebuah pengenal yang berbeda. Misalnya buku Jane Austin diberi pengenal huruf 'A', seperti terlihat pada tabel 1. Kemudian semua pengenal tersebut dimasukkan ke himpunan I = {A,C,D,T,W}. Data tabel transaksional *D* terdiri dari 6 transaksi pembelian dimana tiap transaksi pembelian terdiri dari beberapa buku (beberapa item) yang ada di himpunan I seperti yang terlihat pada tabel 2.

Tabel 1. Himpunan I

Nama Buku	Kode
Abdul Kadir	A
Jogiyanto HM	C
Daniel Wirajaya	D
Inge Martiana	T
Andy Kurniawan	W

Tabel 2. Data Tabel Transaksional *D*

Transaksi	Item yang dibeli
1	ACTW
2	CDW
3	ACTW
4	ACDW
5	ACDTW
6	CDT

Pada tabel 3 terlihat semua *frequent* itemset, yaitu yang memiliki nilai *support* sama atau lebih besar dari 3 (asumsi : *min_sup* = 50%). Itemset CDW dan ACTW yang tercetak tebal adalah maksimal *frequent* itemset. Pada gambar 1 terlihat semua kaidah asosiasi yang memiliki nilai *confidence* 100%.

Tabel 3 Tabel *Frequent* Itemset (*min_sup*=50%)

Support	Itemset
100% (^)	C
83% (5)	W, CW
67% (4)	A, D, T, AC, AW, CD, CT, ACW
50% (3)	AT, DW, TW, ACT, ATW, CDW, CTW, ACTW

ASSOCIATION RULES (min_comf=100%)

A → C (4\4)	AC → W (4\4)	TW → C (3\3)
A → W (4\4)	AT → C (3\3)	AT → CW (3\3)
A → CW (4\4)	AT → W (3\3)	TW → AC (3\3)
D → C (4\4)	AW → C (4\4)	ACT → W (3\3)
T → C (4\4)	DW → C (3\3)	ATW → C (3\3)
W → C (5\5)	TW → A (3\3)	CTW → A (3\3)

Gambar 1. Data Kaidah Asosiasi

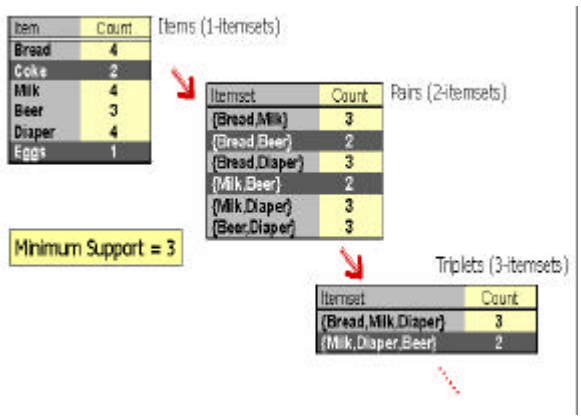
Selama *frequent* itemset yang lain adalah subset dari salah satu maksimal *frequent* itemset, maka proses pencarian itemset dapat dikurangi dengan hanya mencari maksimal *frequent* itemset saja.

Untuk membentuk semua kaidah asosiasi, diperlukan nilai *support* dari semua *frequent* itemset. Proses ini dapat dengan mudah dilakukan selama maksimal *frequent* itemset sudah ditemukan.

3. METODE APRIORI

Prinsip dari Algoritma Apriori adalah:
- Kumpulkan jumlah item tunggal, dapatkan item besar.
- Dapatkan candidate pairs, hitung => large pairs dari item-item
- Dapatkan candidate triplets, hitung => large triplets dari item-item dan seterusnya.
- Sebagai petunjuk: Setiap subset dari sebuah frequent itemset harus menjadi frequent

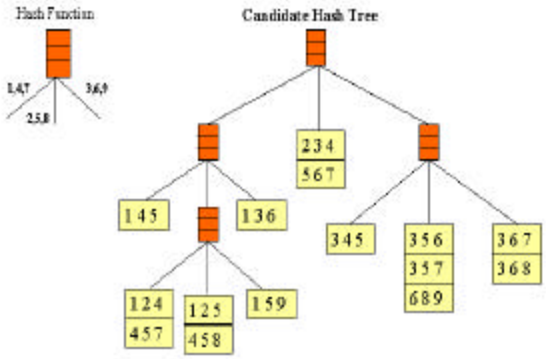
Ilustrasi dari algoritma apriori adalah sebagai berikut:



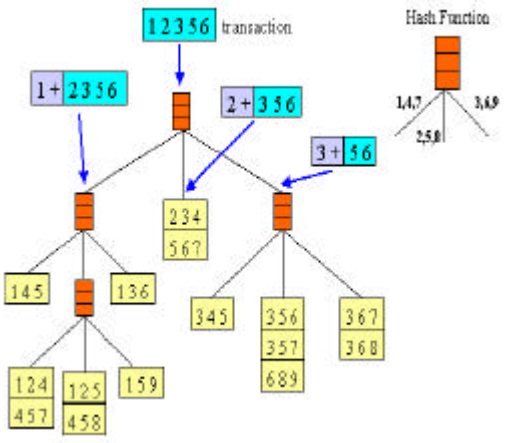
Gambar 2. Ilustrasi Algortima Apriori

Apabila dituliskan dalam bentuk pseudo-code, algoritma apriori adalah sebagai berikut:

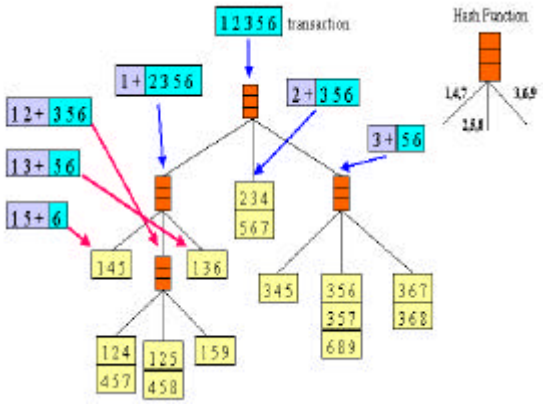
```
F1 = {Frequent1 – Item sets};
K = 2;
While (Fk-1 tidak kosong)
{
    Ck = Apriori_generate(Fk-1);
    Untuk semua transaksi dalam T
          {
            Subset(Ck, t);
          }
          Fk = { C in Ck s.t. c.count >= min_sprt};
}
Answer = Union dari semua set Fk;
Apriori_generate (F (k-1))
{
    join Fk-1 dengan sehingga Fk-1,
          c1 = (i1, i2, ..., Ik-1) dan c2 = (J1, J2, ...,
          Jk-1)  join bersama-sama jika Ip = Jp
          untuk 1<=p<=k-1,
          dan kemudian candidate baru, c, punya
          bentuk c=(i1, i2, ..., Ik-1, Jk-1).
          c kemudian ditambahkan ke struktur
          hash tree.
}
```



Gambar 3. Ilustrasi Hash Tree 1



Gambar 4. Ilustrasi Hash Tree 2



Gambar 5 Ilustrasi Hash Tree 3

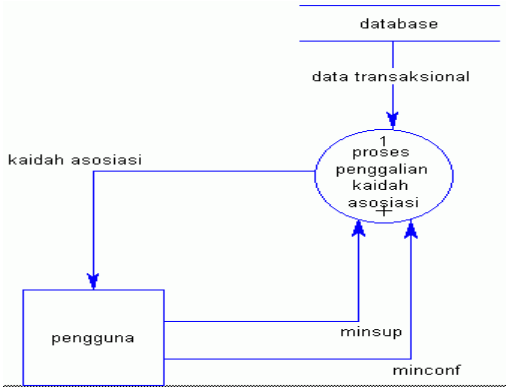
4. PERANCANGAN SISTEM

Perancangan proses dari perangkat lunak ini menggunakan pendekatan fungsional yang direpresentasikan menggunakan *Data Flow Diagram (DFD)* atau Diagram Aliran Data (DAD). DAD ini digunakan untuk menunjukkan secara fisik alur proses dan data pada perangkat lunak yang akan dibuat. Pembuatan DAD tersebut menggunakan perangkat lunak Power Designer versi 6.1. Diagram aliran data ini menjelaskan alur proses mulai dari level 0 sampai dengan level 2.

Diagram untuk penggalian kaidah asosiasi pada level 0 merupakan DAD yang paling sederhana, seperti yang terlihat pada Gambar 6. Pada DAD dapat terlihat bahwa proses penggalian kaidah asosiasi membutuhkan 3 data masukan, yaitu data transaksional, minsup (*support* minimum), dan minconf (*confidence* minimum). Data transaksional didapatkan dari basis data, sedangkan data minsup dan minconf dimasukkan oleh pengguna. Dan data keluaran dari proses tersebut adalah data kaidah asosiasi, yang diberikan ke pengguna.

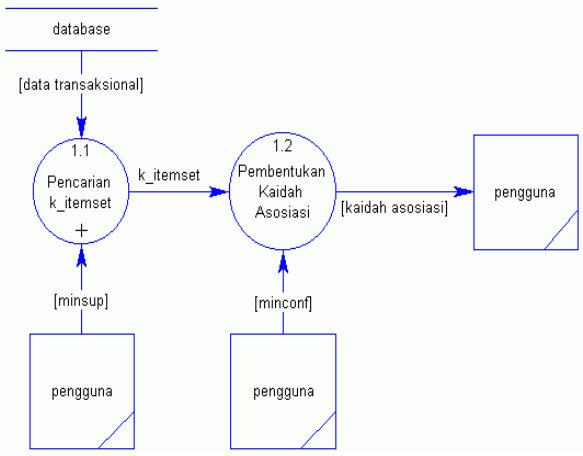
Pada DAD tingkat 1, seperti yang terlihat pada gambar 6, proses penggalian kaidah asosiasi dijabarkan menjadi 2 subproses, yaitu:

a. Proses pencarian *frequent* k-itemset, dan
b. Proses pembentukan kaidah asosiasi.



Gambar 6. DAD tingkat 0 (Context Diagram)

Seperti yang terlihat pada gambar 7, proses pencarian *frequent* k-itemset membutuhkan 2 data masukan, yaitu data transaksional dan minsup. Sedangkan data keluaran dari proses tersebut adalah semua *frequent* k-itemset. Data ini nantinya akan digunakan sebagai data masukan untuk proses selanjutnya, yaitu proses pembentukan kaidah asosiasi. Jadi, proses pembentukan kaidah asosiasi membutuhkan 2 data masukan, yaitu k-itemset yang *frequent* (data keluaran dari proses pemcarian k-itemset), dan data minconf. Dan data keluaran dari proses pembentukan kaidah asosiasi adalah semua data kaidah asosiasi yang memenuhi minconf.

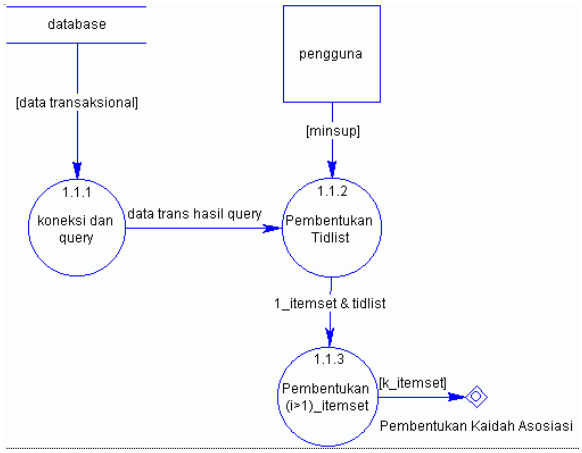


Gambar 7. DAD level 1

Seperti terlihat pada gambar 8, proses pencarian *frequent* k-itemset dipecah lagi menjadi 3 subproses, yaitu:

a. Koneksi ke database dan query
b. Pembentukan tidlist
c. Pembentukan (i>1)-itemset

Pada proses koneksi dan query, dilakukan koneksi ke suatu basis data dimana data transaksional yang dibutuhkan berada. Dari hasil koneksi tersebut, pengguna dapat memilih tabel transaksional yang akan digunakan sebagai data masukan. Kemudian data tabel transaksional ini diquery berdasarkan atom dan tid. Jadi, data keluaran dari proses ini adalah data transaksional yang sudah diquery. Data ini akan menjadi data masukan pada proses selanjutnya.



Gambar 8. DAD level 1.2

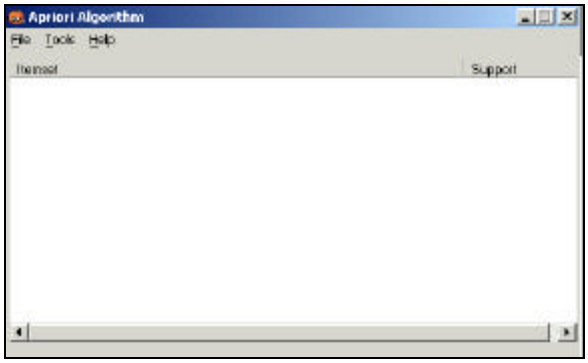
Proses selanjutnya adalah proses pembentukan tidlist. Data yang dibutuhkan pada proses ini ada 2, yaitu data transaksional yang sudah diquery, dan data minsup yang dimasukkan pengguna. Data keluaran dari proses ini adalah *frequent* 1-itemset beserta nilai tidlistnya. Data ini nantinya akan digunakan untuk proses terakhir.

Proses terakhir adalah proses pembentukan (i>1)-itemset. Data yang dibutuhkan adalah *frequent* 1-itemset beserta tidlistnya. Proses ini dilakukan berdasarkan nilai minsup, yang telah dimasukkan sebelumnya pada proses pembentukan tidlist. Data keluaran yang dihasilkan adalah semua *frequent* k- itemset, yang akan digunakan untuk proses pembentukan kaidah asosiasi.

5. IMPLEMENTASI SISTEM

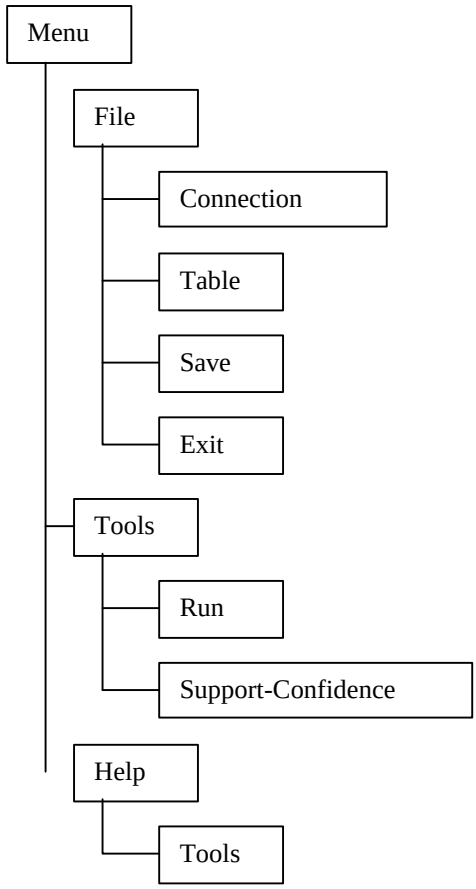
Bahasa pemrograman yang digunakan untuk melakukan implementasi adalah Microsoft Visual C++ 6.0. Basis data yang

digunakan untuk mendukung pembuatan perangkat lunak ini adalah Microsoft Access 2000. Tampilan awal dapat dilihat pada gambar 9.



Gambar 9. Menu Utama Program

Pada gambar 9. terlihat bahwa menu utama yang digunakan ada 3, yaitu File, Tools dan Help. Untuk menunjukkan bentuk susunan tree secara jelas, dapat dilihat pada gambar 10.



Gambar 10. Susunan Menu Pada Perangkat Lunak

1. Menu File
   Pada menu ini terdapat beberapa submenu yaitu:
   - Connection: digunakan untuk melakukan koneksi ke suatu basis data.
   - Table: digunakan untuk memilih table yang ada pada basis data sesuai dengan hasil koneksi.
   - Save: digunakan untuk menyimpan data keluaran proses penggalian kaidah asosiasi.
   - Exit: digunakan untuk keluar dari program.
2. Menu Tools
   Pada menu ini terdapat beberapa submenu yaitu:
   - Run: digunakan untuk menjalankan proses penggalian kaidah asosiasi.
   - Support-Confidence: digunakan untuk memberikan data masukan berupa data minimum support dan data minimum confidence.
3. Menu Help
   Pada menu ini hanya terdapat satu submenu saja, yaitu submenu About. Submenu About ini berisi data pembuat perangkat lunak.

6. UJI COBA DAN ANALISA

Perangkat keras yang dipergunakan pada uji coba ini adalah komputer dengan prosesor Intel Pentium III 500 MHz dengan memori sebesar 128 MB. Sedangkan sistem operasi yang dipergunakan adalah Windows 2000 Professional. Basis data yang digunakan adalah Microsoft Access 2000.

Pada uji coba yang akan dilakukan, digunakan 3 data tabel transaksional yang berbeda. Spesifikasi ketiga data tabel transaksional yang akan digunakan adalah seperti yang terlihat pada tabel 4.

Tabel 4. Spesifikasi data untuk Uji Coba

	Data 1	**Data 2**	**Data 3**
Nama Tabel	Trans1	Trans2	Trans3
Jml Record	24.674	36.982	49.370
Jml Transaksi	5.000	7.500	10.000
Jml Item	30	30	30

Pada uji coba ini, yang akan dilakukan adalah menjalankan perangkat lunak dengan

parameter yang sama pada tabel yang berbeda. Tabel yang akan digunakan ada tiga 3, yaitu: Trans1, Trans2, dan Trans3. Sedangkan parameter yang digunakan adalah minsup dan minconf. Nilai minsup yang akan dimasukkan adalah 1%, 0.75%, 0.5%, dan 0.25%. Sedangkan nilai minconf yang dimasukkan adalah 20%. Hasil uji coba untuk masing-masing tabel dapat dilihat pada tabel 5, tabel 6, dan tabel 7.

Tabel 5. Hasil Uji Coba 1 menggunakan Data1

***Support* Minimum(%)**	**0.25**	**0.5**	**0.75**	**1**
Waktu Proses	458.5	373	322.5	275
Jml k-itemset	3526	1147	396	365
Waktu Pembuatan tidlist	14.5	14.5	14	15.5
Jml kaidah asosiasi	1509	1178	133	56

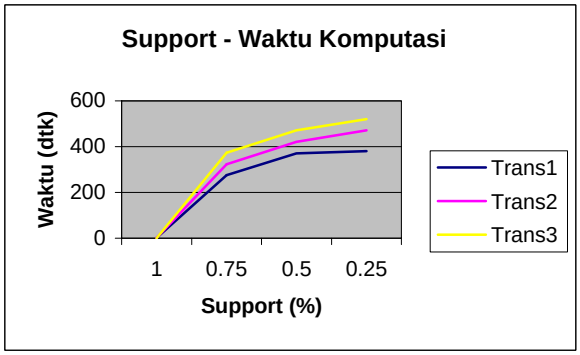
Tabel 6. Hasil Uji Coba 1 menggunakan Data 2

***Support* Minimum(%)**	**0.25**	**0.5**	**0.75**	**1**
Waktu Proses	584.5	472	421.5	371
Jml k-itemset	3526	1148	374	365
Waktu Pembuatan tidlist	10.5	9.5	10.5	11
Jml kaidah asosiasi	1018	888	50	23

Tabel 7. Hasil Uji Coba 1 menggunakan Data 3

***Support* Minimum(%)**	**0.25**	**0.5**	**0.75**	**1**
Waktu Proses	653	520.5	472	380
Jml k-itemset	4527	1357	473	465
Waktu Pembuatan tidlist	13.5	13.5	13.5	14
Jml kaidah asosiasi	989	858	41	18

Dari ketiga hasil coba di atas, dibuat suatu grafik berdasarkan nilai support dan waktu yang dibutuhkan untuk mencari *frequent* k-itemset (*Process Time*). Grafik tersebut dapat dilihat pada gambar 11.



Gambar 11. Grafik Uji Coba Support - Waktu Komputasi

7. PENUTUP

Dari hasil uji coba perangkat lunak yang telah dilakukan, maka dapat diambil beberapa kesimpulan sebagai berikut:

1. Berdasarkan uji coba Nilai Support dengan nilai support bervariasi antara 1%, 0.75%, 0.5% dan 0.25%, dapat diambil kesimpulan bahwa semakin kecil nilai support maka waktu komputasi semakin lama, jumlah itemset yang dihasilkan semakin banyak, dan jumlah kaidah asosiasi yang dibentuk semakin banyak pula.
2. Waktu komputasi untuk menghasilkan kaidah asosiasi dipengaruhi oleh jumlah transaksi - yaitu semakin besar jumlah transaksi maka waktu komputasi yang dibutuhkan akan semakin lama.
3. Penggunaan struktur data "tidlist" pada algoritma apriori menyebabkan waktu komputasi yang dibutuhkan relatif berkurang karena hanya memerlukan pembacaan basis data sekali saja.

Algoritma apriori dapat dikembangkan lebih lanjut untuk semakin meningkatkan performa dan kinerja dalam memperoleh kaidah asosiasi dengan melakukan:

1. *Transaction Reduction*, dimana sebuah transaksi yang tidak berisi itemset frequent-k tidak perlu dimasukkan dalam subsequent-scans.
2. *Partitioning*, dimana sejumlah itemset yang tingkat frekuensinya tinggi dalam basis data, paling tidak harus didalam satu partisi dari basis data.

3. *Dynamic Itemset Counting*, dimana penambahan sebuah candidate itemset hanya dilakukan apabila semua subsetnya mempunyai tingkat frekuensi yang tinggi.

DAFTAR PUSTAKA

1. Mohammed J. Zaki, member, IEEE. *Scalable Algorithm for Association Mining*. IEEE Transactions on Knowledge and Data Engineering, Volume 12, No. 3, May/June 2000.

2. S. J. Yen, and A. L. P. Chen. *An Efficient Approach to Discovery Knowledge from Large Databases.* In Proceedings of the 4th International Conference on Parallel and Distributed Information Systems, 1996.

3. R. Agrawal and R. Srikant. *Fast Algorithm for Mining Association Rules.* In Proceedings of the International Conference on Very Large Data Bases, 1994.

4. R. Agrawal and et al. Mining Association Rules Between Sets of Items in Large Databases. In Proceedings of the ACM SIGMOD, 1993.

5. R. Agrawal, H. Mannila , R. Srikant, H. Toivonen, and A. Inkeri Verkamo, "Fast Discovery of Association Rules," *Advances in Knowledge Discovery and Data Mining,* U. Fayyad and et al., eds., pp. 307-328, Menlo Park, Calif.: AAAI Press, 1996.