DATA MINING DENGAN ALGORITMA APRIORI PADA RDBMS ORACLE

Dana Sulistiyo Kusumo [1], Moch. Arief Bijaksana [2], Dhinta Darmantoro[3]

Jurusan Teknik Informatika Sekolah Tinggi Teknologi Telkom
[1]dana@stttelkom.ac.id, [2]arifb@telkom.co.id, [3]dhinta@stttelkom.ac.id

Abstrak

Data mining **merupakan proses analisis data menggunakan perangkat lunak untuk menemukan pola dan aturan (*rules*) dalam himpunan data. *Data mining* dapat menganalisis data yang besar untuk menemukan pengetahuan guna mendukung pengambilan keputusan. Dalam penelitian ini akan dibahas *Association Rule* sebagai salah satu fungsi *data mining* yang diimplementasikan menggunakan Algoritma *Apriori*. Akan dianalisis pula dua teknik penghitungan *support* di *candidate generation* pada Algoritma *Apriori,* yakni : *K-way dan 2 Group-By* pada tiga sampel dataset dengan atribut transaksi *id* dan item. Pada penelitian ini terlihat bahwa permasalahan penghitungan *support* di *candidate generation* merupakan *bottleneck* dari Algoritma Apriori dimana perbaikan Algoritma Apriori ditekankan pada *candidate generation* dan efektivitas dari Algoritma *Apriori*. Penelitian ini dilakukan pada RDBMS Oracle dengan memanfaatkan *tools* TKPROF untuk mengukur performansi query berdasarkan operasi I/O pada penghitungan support di *candidate generation*. Hasil penelitian membuktikan bahwa metode *support counting K-way* lebih baik daripada *Two Group-by*.**

Kata Kunci : ***Data Mining, Association Rule,*** **Algoritma** ***Apriori, candidate generation, K-way, 2 Group-By***

Abstract

Data Mining is a data process analysis using software for finding pattern and rule in dataset. Data Mining could analyze large dataset to obtain knowledge as a pattern having meaning for making decision in management level for business organization. This paper discusses Association Rule as one function of data mining implemented using Apriori Algorithm. Two counting techniques of support in candidate generation in Apriori algorithm, i.e. K-way and 2 group-By on three dataset samples with transaction attribute id and item, would also be discussed. This research reveals that the problem of support counting in candidate generation is a bottleneck of Apriori algorithm, where Apriori Algorithm fitting is emphasised on candidate generation and effectiveness of Apriori Algorithm. This research was done on RDBMSS Oracle utilizing TKPROF tools to measure query performance based on I/O operation on support counting in candidate generation. The result shows that K-way support counting method is better than Two Group-by method.

Keywords : Data Mining, Association Rule, Apriori algorithm, support, candidate generation, K-way, 2 Group-By and RDBMS.

1. Pendahuluan

Data mining merupakan suatu proses pendukung pengambil keputusan dimana kita mencari pola informasi dalam data. Pencarian ini dapat dilakukan oleh pengguna, misalnya dengan menggunakan query (dalam kasus ini sangat sulit dilakukan) atau dapat dibantu dengan suatu aplikasi yang secara otomatis mencari pola informasi pada basis data. Pencarian ini disebut *discovery*. *Discovery* adalah proses pencarian dalam basis data untuk menemukan pola yang tersembunyi tanpa ide yang didapatkan sebelumnya atau hipotesa tentang pola yang ada. Dengan kata lain aplikasi mengambil inisiatif untuk menemukan pola dalam data tanpa pengguna berpikir mengenai pertanyaan yang relevan terlebih dulu.

Salah satu bentuk pola yang dapat dihasilkan *data mining* adalah *association rule*. *Association Rule* dapat digunakan untuk menemukan: hubungan atau sebab akibat.

Association rule memiliki bentuk LHS $\Rightarrow$ RHS dengan interpretasi bahwa jika setiap item dalam LHS (*Left Hand Side*) dibeli, maka item dalam RHS (*Right Hand Side*) juga dibeli. *Association rule* dapat dihasilkan dengan Algoritma Apriori. Salah satu penggunaan *Association rule* adalah mendukung pengambilan keputusan dalam bidang pemasaran, misalnya untuk mengetahui pola pembelian pelanggan, penentuan tata letak barang dan lain-lain.

Salah satu obyek untuk *data mining* adalah RDBMS. Umumnya *data mining* dilakukan pada penyimpanan data berukuran besar. Dalam penelitian ini akan dilakukan pengujian terhadap proses penghitungan *support* dalam implementasi *data mining* pada RDBMS.

Analisis terhadap teknik penghitungan *support* perlu dilakukan karena faktor :

1. Banyaknya kandidat *frequent-itemset* yang dihasilkan sebagai input terhadap penghitungan *support*,
2. Scan/pembacaan record pada RDBMS.

Kedua faktor diatas dapat mempengaruhi jalannya pemrosesan query pada RDBMS.

Penelitian ini dilakukan dengan tujuan untuk menerapkan Algoritma Apriori dengan teknik *support counting* K-way dan 2 Group-By, mencari *Association Rule* pada RDBMS Oracle, dan mengimplemantasikannya dalam bentuk perangkat lunak. Analisis Algoritma Apriori didasarkan pada hasil pengukuran performansi proses query pada teknik *support counting* K-way dan 2 Group-By menggunakan tools TKPROF pada RDBMS Oracle.

Batasan masalah untuk penelitian ini adalah sebagai berikut :

1. Data yang digunakan merupakan tabel relasional
2. Hanya digunakan tiga dataset sebagai bahan pengukuran untuk analisis pemrosesan query

2. *Data Mining*

2.1 Definisi *Data Mining*

Pengertian *Data Mining* atau *Knowledge Discovery in Database* menurut William J. Frawley, Gregory Piatetsky-Shapiro dan Christopher J. Matheus [10]:

Data Mining atau Knowledge Discovery in Database(KDD) adalah penyaringan data secara implisit dimana sebelumnya tidak diketahui terdapatnya informasi yang potensial.

2.2 Fungsi dan Tugas Data Mining

Data mining menganalisis data menggunakan tool untuk menemukan pola dan aturan dalam himpunan data. Perangkat lunak bertugas untuk menemukan pola dengan mengidentifikasi aturan dan fitur pada data. Tool Data mining diharapkan mampu mengenal pola ini dalam data dengan input minimal dari user.

Dalam penelitian ini pembahasan Data Mining diklasifikasikan dalam fungsi Association.

2.3 Association Rule

Tipe pola yang penting yang dapat ditemukan dari basis data adalah sebuah aturan. *Association rule* mempunyai bentuk LHS $\Rightarrow$ RHS dengan interpretasi jika setiap item dalam LHS dibeli maka sepertinya item dalam RHS juga dibeli. Dua pengukuran penting untuk sebuah rule adalah *support* dan *confidence*. Kita dapat menghitung semua association rule dengan ambang support dan confidence masukkan pengguna dengan post-processing frequent-itemset.

Secara umum Association Rule mempunyai bentuk : LHS $\Rightarrow$ RHS, dimana LHS dan RHS adalah himpunan item; jika setiap item-item dalam LHS terdapat dalam transaksi maka item-item di RHS juga terdapat dalam transaksi.

Ada dua aturan pengukuran untuk ‘association rule’ :

1. **Support**

Support untuk himpunan item adalah prosentase transaksi yang berisi semua item-item ini. Support untuk aturan LHS $\Rightarrow$ RHS di-support untuk himpunan item-item LHS $\cup$ RHS.

2. **Confidence**

Pertimbangkan transaksi yang berisi semua item dalam LHS. Confidence untuk rule : LHS $\Rightarrow$ RHS adalah prosentasi transaksi yang juga terdiri semua item-item dalam RHS.

Lebih tepatnya, misalkan sup (LHS) adalah prosentase transaksi yang berisi LHS dan sup (LHS $\cup$ RHS) adalah prosentase transaksi yang berisi LHS dan RHS, maka confidence rule: LHS $\Rightarrow$ RHS adalah sup(LHS $\cup$ RHS)/sup (LHS).

Permasalahan Association Rule dapat dikomposisikan menjadi dua sub masalah, yaitu:

1. Penemuan semua kombinasi item-item, yang disebut *frequent-itemset*, yang support-nya lebih besar daripada *minimum support*.
2. Gunakan *frequent-itemset* untuk membangkitkan aturan yang diinginkan. Idenya adalah, katakan, ABCD dan AB sering muncul dalam transaksi, maka aturan AB $\Rightarrow$ CD akan dipenuhi jika perbandingan antara *support*(ABCD) terhadap *support*(AB) minimum sebesar *minimum confidence*. Semua rule akan mempunyai *minimum support* karena ABCD sering muncul dalam transaksi.

2.4 Algoritma Apriori

Langkah yang membutuhkan pemrosesan lebih adalah penemuan *frequent-itemset.* Algoritma untuk menemukan *frequent-itemset* berdasar pada sifat *frequent-itemset*:

Sifat Apriori : Setiap subset *frequent-itemset* harus menjadi *frequent-itemset* [12].

Algoritma Apriori untuk menemukan *frequent-itemset* merupakan iterasi pada data. Pada iterasi ke-k ditemukan semua himpunan item-item yang mempunyai k item yang disebut k-*itemset.* Setiap iterasi terdiri dari dua tahap.. Pertama, adalah tahap pembangkitan kandidat (*candidate generation*) dimana himpunan semua *frequent*$(k-1)$-*itemset* F_{k-1} yang ditemukan pada pass ke-$(k-1)$ digunakan untuk membangkitkan kandidat *itemset* C_k. Prosedur pembangkitan kandidat menjamin bahwa C_k adalah superset dari himpunan semua *frequent k-itemset.* Kemudian data di-scan dalam tahap Penghitungan *Support* (*Support Counting*). Pada akhir pass C_k diperiksa untuk menentukan kandidat mana yang sering muncul, menghasilkan F_k. Penghitungan *support* berakhir ketika F_k atau C_{k+1} kosong.

Untuk membangkitkan *rule* akan dibangkitkan lebih dahulu *candidate rule. Candidate rule* berisi semua kemungkinan *rule* yang memiliki *support* > *minimum support* karena input *candidate rule* adalah *frequent-itemset*. Kemudian *candidate rule* akan di-

join dengan tabel *F* untuk menemukan *support antecedent*. *Confidence rule* dihitung dengan cara membandingkan *support rule* dengan *support antecedent rule*. Hanya *rule* yang mempunyai *confidence > minimum confidence* yang disimpan dalam tabel *rule* (tabel *R*).

3. Pengukuran data penelitian

Untuk menganalisis teknik penghitungan support *K-way* dan *2 Group-By* pada Algoritma *Apriori* digunakan 3 sampel dataset dengan atribut *transaction identifier (id)* dan *item identifier (item),* dimana untuk setiap *id* terdapat beberapa item. Deskripsi data yang digunakan dalam penelitian ini diperlihatkan pada Tabel 1 di bawah ini.

Tabel 1. Data Penelitian

I	II	III	IV	V	VI
A	102	27	9	5	3,78
B	27701	5840	16	9	4,74
C	97084	11680	32	18	8,31

Keterangan judul kolom :

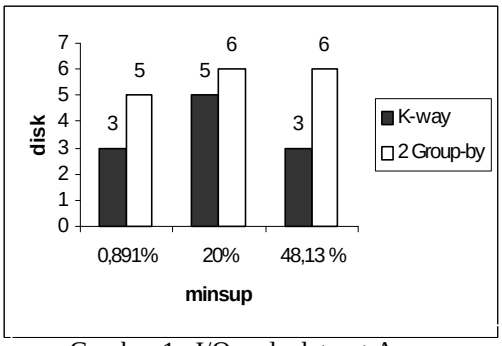
I : Dataset
II : ∑Record
III : ∑Group
IV : ∑Transaksi
V : ∑Item
VI : Max. Itemset
VII : ∑Rata-rata item per transaksi

Untuk menganalisis kedua teknik penghitungan *support* di atas digunakan tools TKPROF untuk mengetahui besarnya operasi I/O (disk) pada eksekusi Algoritma *Apriori.* Pengukuran terhadap operasi I/O (disk) dijadikan suatu parameter karena operasi I/O (disk) mendominasi waktu total eksekusi query pada RDBMS. Pada Gambar 1, 2, dan 3 ditampilkan hasil pengukuran operasi I/O (disk) pada tiga sampel dataset.

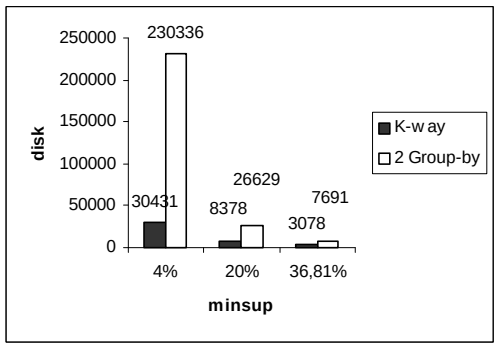
Performansi kedua metode *support counting* dipengaruhi oleh besarnya *minimum support* dan data yang digunakan. Jika *minimum support* kecil maka semakin banyak kebutuhan I/O dibandingkan dengan jika *minimum support* besar. Hal ini disebabkan karena jumlah *record* yang diproses lebih banyak. Data yang di-*mining* juga memberi pengaruh terhadap jumlah *record* yang diproses. Besarnya data yang di-*mining* berbanding lurus terhadap jumlah *record* yang diproses. Sehingga berdasar hasil pengukuran, K-way secara umum lebih baik daripada Two Group-by, K-way karena lebih sedikit dalam melakukan operasi I/O dibandingkan dengan Two Group-by.

4. Analisis Algoritma *Apriori*

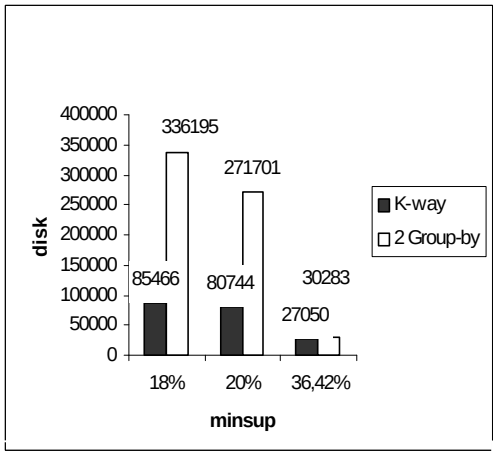
Berdasarkan teori pemrosesan query, Two Group-by berbentuk *nested* query dengan *inner* query dan *outer* query-nya berbentuk *group by*. Pada pemrosesan query Two Group-by, *Inner group-by* melakukan *sorting* menggunakan banyak relasi antara. Hasil query dari *inner* query kemudian menjadi masukan *outer* query. Pada Two Group-by agregasi dilakukan dua kali.



Gambar 1. I/O pada dataset A



Gambar 2. I/O pada dataset B



Gambar 3. I/O pada Dataset C

Pemrosesan query pada K-way lebih sedikit menggunakan relasi antara yang berupa *sorting* dibandingkan Two Group-by. Pada K-way agregasi hanya dilakukan sekali.

Berdasar pemrosesan query diatas, K-way lebih sedikit dalam melakukan *sorting* dan operasi agregasi dalam mengeksekusi query dibandingkan Two Group-by.

Algoritma Apriori untuk menemukan *frequent-itemset* merupakan iterasi pada data. Algoritma Apriori dimulai dengan pembangkitan kandidat *itemset*. Pembangkitan *itemset* menjamin C_k adalah superset dari himpunan semua *frequent k-itemset.*

Kemudian data di-*scan* dalam tahap Penghitungan *Support* (*Support Counting*).

insert into F_k select $item_1$, …, $item_k$, count(*) from (select $item_1$, …, $item_k$, count(*) From Xact, C_k where item = $C_k.item_1$ or….. item = $C_k.item_k$ group by $item_1$, …, $item_k$, tid having count(*) = k) group by $item_1$, …, $item_k$ having count(*) > : *minsup*

Gambar 4. Penghitungan support *2 Group-by*

insert into F_k select $item_1$, … , $item_k$, count(*) from C_k, Xact t_1, …, Xact t_k where $t_1.item = C_k.item_1$ and …$t_k.item = C_k.item_k$ and $t_1.id = t_2.id$ and …$t_{k-1}.id = t_k.id$ group by $item_1$, … , $item_k$ having count(*) > :*minsup*

Gambar 5. Penghitungan support *K-way*

Penghitungan *support* menggunakan data dengan skala menengah ke atas dengan nilai *minimum support* yang kecil dapat ditangani menggunakan K-way pada bagian *support counting*. Penggunaan Two Group-by pada proses *support counting* pada data berskala menengah ke atas tidak dianjurkan karena untuk *minimum support* dengan nilai yang kecil akan menyebabkan pemrosesan query dengan banyak *sorting* (misalnya : pada *dataset* B dengan *minimum support* = 0,001, K-way masih dapat berjalan dengan baik tetapi pada 2 Group-by, algoritma Apriori berhenti tidak normal). Dengan banyaknya *sorting* menyebabkan *temporary segments* yang digunakan sebagai area kerja operasi *sorting* pada *tablespace* yang ada tidak dapat dikembangkan lagi karena telah mencapai batas maksimal. Sehingga dapat menyebabkan iterasi dalam pengisian tabel F tidak selesai yang dapat mengakibatkan berhentinya algoritma Apriori secara tidak normal.

Pada akhir iterasi, C_k diperiksa untuk menentukan kandidat mana yang sering muncul, menghasilkan F_k. Penghitungan *support* berakhir ketika F_k atau C_{k+1} kosong.

Untuk menghasilkan *rule* digunakan tabel RC yang berisi semua kemungkinan *rule* dari *frequent-itemset* dengan nilai *sup > minimum support.* Jika nilai *minimum support* yang diberikan sangat kecil menyebabkan pengisian tabel RC terdiri atas banyak operasi *sorting*. Hal ini menyebabkan *temporary segments* yang digunakan sebagai area kerja operasi *sorting* pada *tablespace* yang ada tidak dapat dikembangkan lagi karena telah mencapai batas maksimal. Sehingga dapat menyebabkan iterasi dalam pengisian tabel RC tidak selesai yang dapat mengakibatkan berhentinya algoritma Apriori secara tidak normal.

Pembangkitan *rule* dimulai dengan cara melakukan join antara *antecedent* dari kandidat *rule* dengan *frequent-itemset* untuk mendapatkan *support antecedent*. *Confidence* dari *rule* didapatkan dengan membagi *support* dari *rule* dengan *support antecedent*-nya. Algoritma berakhir setelah pembangkitan *rule* menghasilkan *rule-rule* dengan nilai *support > minimum support* dan *confidence > minimum confidence.*

5. Kesimpulan

Berdasar analisis pengukuran, metode *support counting* K-way lebih baik daripada 2 Group-by karena operasi *sorting* dan agregasi pada K-way lebih sedikit dibandingkan Two Group-by sehingga lebih sedikit mengkonsumsi I/O (disk). Besarnya *minimum support* berbanding terbalik dengan banyak baris *record* yang diproses. Sedangkan besarnya baris *record* yang diproses berbanding lurus dengan konsumsi operasi I/O, akses *buffer* dan *elapsed time.* K-way mampu dengan baik menangani data berskala menengah ke atas dengan nilai *minimum support* yang kecil, sedangkan penggunaan 2 Group-by dapat mengakibatkan berhentinya Algoritma Apriori secara tidak normal. Teknik *candidate frequent k-itemset (candidate generation)* merupakan *bottleneck* dari Algoritma *Apriori* karena:

a. banyaknya kandidat *frequent-itenset* yang dihasilkan sebagai input terhadap penghitungan *support* sangat besar.
b. scan/pembacaan berulang record pada RDBMS

Diperlukan konfigurasi perangkat keras yang mendukung implementasi *data mining.* Selain itu, pada sisi RDBMS perlu dilakukan *tuning* pada implementasi *data mining*, karena sifat data yang di-*mining* biasanya merupakan data berukuran besar. Perlu dilakukan perbaikan terhadap teknik *candidate frequent k-itemset (candidate generation)* yang merupakan *bottleneck* algoritma *Apriori.* Dapat dilakukan pengembangan lanjut implementasi *Association Rule*, yaitu meliputi:

a. dataware housing: OLAP mining
b. data mining yang lain: spatial data, multimedia data, time series data dsb.

Daftar Pustaka

[1] Agrawal R., T. Imielinski, and A.Swami. 1993. *Special Issue on Learning and Discovery in Knowledge Based Databases. Database Mining : A Performance Perspective*. IEEE Transactions on Knowledge and Data Engineering, 914-925.

[2] Agrawal R. and R. Srikant. 1994. *Fast Algorithms for Mining Association Rules in Large Databases.* Research Report RJ 9839. San Jose, CA: IBM Almaden Research Center

[3] Ambarwaty, Retno. 1997. *Sistem Pendukung Keputusan Pemasaran Jasa Telepon Dengan Segmentasi Pelanggan Psikografis. Studi Kasus di Kandatel Jakarta Barat.* Bandung: Sekolah Tinggi Teknologi Telkom.

[4] Direct Marketing Association. 1992. *Managing Database Marketing Technology for Sucess*.
[5] Date, C.J. 1994. *An Introduction to Database System.* New York: Addison Wesley.
[6] Fayyad, U., G.Piatetsky-Shapiro, and P. Smyth. *1996. From data mining to knowledge discovery : An overview. In Advances in Knowledge Discovery and Data Mining.* Cambridge, MASS: AAAI/MIT Press,
[7] *Han, Jiawei and M. Kamber. 2000. Data Mining: Concepts and Techniques. Morgan Kaufmann.*
[8] *Han, Jiawei. Towards On-line Analytical Mining : An Integration of OLAP and Data Mining. Intelligent Systems*. Canada: Database Research Lab. DBMiner Technology Inc. and School of Computing Science Simon Fraser University, British Columbia. http://www.dbsummit.com/articles/Han/
[9] Information Discovery, Inc. 1996. *Datamines for Dataware Houses*.
[10] The Parallel Computer Center. 1995. *Data Mining : An Introduction*. Student Note. The Queen's University of Belfast. http://www.pcc.qub.ac.uk
[11] Piatetsky-Shapiro, G., Fayyad, U., and P. Smyth. 1996. *The KDD Process for Extracting Useful Knowledge from Volumes of Data.* Communications of The ACM, November 1996/Vol. 36, No. 11. ACM
[12] Ramakrishnan, Raghu. 2000. *Database Management System*. MacGraw Hill.
[13] Sarawagi, S., Thomas S. and R. Agrawal. 1998. *Integrating association rules mining with relational database system : Alternative and implications. Research Report RJ 10107 (91923)*. IBM Almaden Research Center.
[14] Silberschatz , Abraham, Henry F. Korth, and S. Sudarshan. 1997. *Database System Concepts*. The McGraw-Hill Companies, Inc.
[15] Stonebraker, M. 1993 *The DBMS research at crossroads.* Dublin: *Proc. of the VLDB Conference.*
[16] Turban, Efrain. 1995. *Decision Support System and Expert System*, Prentice Hall, Inc.
[17] Yourdon, Edward. 1989. *Modern Structured Analysis.* Prentice Hall, Inc.