Jurnal Ilmiah KURSOR
Menuju Solusi Teknologi Informasi
Vol. 5, No. 4, Juli 2010 ISSN 0216 - 0544

PENGGALIAN *FREQUENT CLOSED ITEMSETS* DENGAN *MULTIPLE MINIMUM SUPPORT* PADA BASISDATA RETAIL

Endah Purwanti
Program Studi Sistem Informasi, Fakultas Sains dan Teknologi, Universitas Airlangga
Kampus C Unair, Jl. Mulyosari, Surabaya, 60115
E-Mail: endah_purwanti@unair.ac.id

Abstrak

Penggalian *frequent pattern* memegang peranan penting pada proses *Association Rule Mining*. Namun penggalian *frequent pattern*,seringkali menghasilkan sejumlah besar *frequent itemsets* dan *rule*, sehingga mengurangi efisiensi dan keefektifan dari proses *mining* karena *user* harus menyaring sejumlah besar *rule* hasil penggalian untuk menemukan *rule* yang penting. Masalah ini dapat ditangani dengan melakukan proses penggalian *rule* hanya pada *frequent closed itemsets*. Dalam dunia *retail*, pihak manajemen dapat memanfaatkan pengetahuan hasil analisis dari basisdata penjualan untuk memahami pola kebutuhan pelanggan untuk membantu membuat keputusan bisnis. Analisis terhadap basisdata *retail* dalam jumlah yang besar bukanlah suatu pekerjaan yang mudah. Penggunaan *minsup* sama (tunggal) untuk semua *item* secara implisit mengasumsikan bahwa semua *item* pada basisdata memiliki sifat dan frekuensi yang sama. Padahal pada kenyataannya, *item* yang berbeda memiliki kriteria yang berbeda untuk mempertimbangkan kepentingannya. *Multiple minimum support* digunakan untuk menggambarkan sifat dasar dan frekuensi dari *item* yang ada. Penelitian ini menerapkan struktur *MIS-tree* untuk menggali *frequent closed itemsets* dengan menggunakan *multiple minimum support*. *MIS-tree* adalah struktur pohon yang dikembangkan serupa dengan struktur *FP-tree*. Hasil uji coba menunjukkan bahwa himpunan *frequent closed itemsets* dengan *multiple minimum support* mampu mereduksi jumlah *redundant rule* yang dihasilkan pada proses *rule mining*.

Kata kunci: *Data Mining, Association Rule Mining, Frequent Closed Itemsets, Multiple Minimum Support.*

Abstract

Excavating of frequent pattern has significant role in process of Association Rule Mining. It, however, results in the number of frequent itemsets and rule which decreases the efficiency and effectiveness in mining process because the users have to filter a number of rules of excavating in order to get the important rule. The obstacles can also be handled by excavating rule process at the frequent closed itemsets. In addition, management in retail industry is able to consider the result of analysis for basis selling data to catch the users needs pattern to construct business policies. The analysis of retail database is not easy one since the use of single minsup for all the item implicitly assumes that all item in the database has sameness in character and frequency. In fact, the different item has distinction in criterion to consider its significance, and multiple minimum support used can describe the basis character and frequency of the items. This research is conducted to apply the structure of MIS-tree to excavate the frequent closed itemsets by using multiple minimum support. Alike structure of FP-tree, MIS-tree is diagram tree which is developed of FP-tree. The result shows that structure of frequent closed itemsets and multiple minimum sport are able to reduce a number of redundant rule resulted in rule mining process.

Kata kunci: Data Mining, Association Rule Mining, Frequent Closed Itemsets, Multiple Minimum Support.

PENDAHULUAN

Data mining merupakan proses penggalian pola yang penting dan tersembunyi dari data yang sangat besar. Salah satu topik penting dalam *data mining* adalah *Association Rule Mining*. *Association Rule Mining* [1] digunakan untuk menemukan relasi antar item yang ada pada basisdata transaksi. Sejak *Association Rule Mining* dikenalkan, telah banyak penelitian yang dilakukan untuk menemukan metode yang efektif untuk melakukan penggalian *frequent itemset*.

Penggalian *frequent pattern mining* seringkali menghasilkan sejumlah besar *frequent itemsets* dan *rule*. Hal ini mengurangi tidak hanya efisiensi namun juga keefektifan dari proses *mining*. Elemen kunci yang membuat *Association Rule Mining* dapat dijalankan adalah *minimum support* (*minsup*). *Minsup* digunakan untuk memangkas atau memperkecil ruang pencarian *frequent itemset* dan juga untuk membatasi jumlah *rule* yang akan dihasilkan. Penentuan nilai *minsup* yang tepat juga menjadi topik penelitian yang menarik.

Terdapat alternatif yang menarik yang diajukan oleh Pasquier dkk [2] sebagai ganti *mining complete set* dari *frequent itemset*. *Association Rule Mining* hanya diperlukan untuk menemukan *frequent closed itemsets*. Implikasi yang penting dari pernyataan tersebut adalah bahwa *mining frequent closed itemsets* memiliki kekuatan yang sama dengan *mining complete set* dari *frequent itemsets*. *Mining frequent closed itemsets* mampu mengurangi jumlah *rule* yang *redundant* yang dihasilkan sehingga menaikkan efisiensi dan keefektifan dari proses *mining*.

Menggunakan *single minsup* secara implisit berarti mengasumsikan bahwa semua *item* pada basisdata memiliki sifat dan frekuensi yang sama. Akan tetapi tidak demikian yang belaku pada kenyataannya. Pada basisdata *retail*, umumnya *item* yang berhubungan dengan keperluan sehari-hari, barang konsumsi, dan barang-barang dengan harga rendah akan dibeli lebih sering daripada barang mewah atau barang dengan harga mahal. Pada situasi tersebut, jika *minsup* yang digunakan terlalu tinggi, maka semua pola yang ditemukan akan berhubungan dengan barang-barang harga murah. Padahal barang tersebut hanya memberikan keuntungan sedikit. Namun, jika *minsup* yang diberikan telalu rendah, maka *rule* yang dihasilkan akan sangat banyak, yang mungkin saja banyak yang tidak berguna. Hal ini akan menyebabkan kesulitan bagi pengambil keputusan untuk memahami pola yang ada yang dihasilkan dari *data mining*.

Untuk menyelesaikan masalah tersebut, Liu dkk [3] telah mengembangkan model *association rule*. Model ini memperbolehkan *user* untuk menggunakan *multiple minimum support*. *Multiple minimum support* itu sendiri digunakan untuk menggambarkan sifat dasar serta frekuensi yang berbeda dari item yang ada.

Berdasarkan latar belakang permasalahan yang telah dijelaskan sebelumnya, maka penelitian ini melakukan penggalian *frequent closed itemsets* dengan menggunakan *multiple minimum support*. Penggalian akan dilakukan dengan menggunakan struktur *Multiple Item Support Tree* (*MIS-Tree*) dan Algoritma *CLOSET* [4]. *MIS-tree* merupakan struktur pohon yang dikembangkan serupa dengan struktur *FP-tree* untuk menyimpan informasi yang ringkas dan penting mengenai *frequent pattern*. Algoritma *CLOSET* merupakan algoritma untuk menggali *frequent closed itemsets*, namun dengan menggunakan *single minimum support*.

ASSOCIATION RULE MINING

Sebuah transaksi $I = \{i_1, i_2, i_3, \ldots, i_d\}$ adalah himpunan *item* yang ditransaksikan, sedangkan $T = \{t_1, t_2, t_3, \ldots, t_n\}$ adalah himpunan transaksi. Setiap transaksi t_i terdiri dari *item* yang merupakan *subset* dari *I*. Sebuah *itemset X* adalah himpunan bagian tidak kosong dari *I*.

Support dari sebuah *itemset X*, disimbolkan dengan *sup(X)*, adalah jumlah transaksi yang mengandung *itemset X*. *Association rule R*: $X \rightarrow Y$ adalah implikasi antara dua *itemset X* dan *Y*, dimana $X, Y \subset I$ dan $X \cap Y = \emptyset$. Nilai *support* dari *rule* disebut dengan $sup(X \rightarrow Y)$, didefinisikan sebagai $sup(X \cup Y)$. *Confidence* dari *rule*, disebut dengan $conf(X \rightarrow Y)$, didefinisikan sebagai $\frac{sup(X \cup Y)}{sup(X)}$. Untuk menemukan *association rule* dari sebuah transaksi diperlukan nilai batasan yaitu *minimum support* (*minsup*) dan *minimum confidence* (*minconf*).

Frequent Closed Itemset

Menurut Liu dkk [3], sebuah *itemset Y* adalah *closed itemsets* jika *Y* adalah *frequent* dan tidak terdapat *superset* langsung $Y' \supset Y$ sedemikian hingga $sup(Y') = sup(Y)$. *Frequent itemsets* sendiri adalah *itemsets* yang nilai *support*-nya lebih besar atau sama dengan *minsup* yang telah ditentukan.

Misalkan diketahui sebuah transaksi $I = \{i_1, i_2, i_3, \ldots, i_d\}$ adalah himpunan item yang ditransaksikan dan $T = \{t_1, t_2, t_3, \ldots, t_n\}$ adalah himpunan transaksi. Setiap transaksi t_i terdiri dari item yang merupakan *subset* dari *I*. Sebuah *itemset X* adalah himpunan bagian tidak kosong dari *I*. *Support* dari sebuah *itemset X*, disimbolkan dengan *sup(X)*, adalah jumlah transaksi yang mengandung *itemset X*. Sebuah *itemset Y* adalah *closed itemset* jika *Y* adalah *frequent itemset* dan tidak terdapat superset langsung $Y' \supset Y$ sedemikian hingga $sup(Y') = sup(Y)$. *Frequent closed itemset* yang dapat digali dari basisdata transaksi pada Tabel 1 adalah {*f, c, b, fb, cb, cp, fcam, fcamp*}.

FP-Growth

FP-tree merupakan pengembangan dari struktur *prefix-tree* untuk menyimpan *frequent pattern* yang telah dikompres. *FP-growth* menggunakan struktur *FP-tree* untuk menemukan himpunan lengkap dari *frequent pattern*. Algoritma *FP-growth* menghasilkan jumlah kandidat yang sedikit dan jumlah pembacaan *dataset* yang minimal sehingga waktu responnya cepat [4].

Sebuah *FP-tree* terdiri dari sebuah *root* yang berisi *null*, himpunan *item prefix subtree* sebagai anak dari *root* dan sebuah tabel *header* berisi *frequent-item*. Setiap *node* pada *prefix subtree* terdiri dari *item-name*, *count*, dan *node-link*. *Count* menunjukkan jumlah transaksi pada basisdata yang menjadi bagian *prefix* yang direpresentasikan oleh *node*, dan *node-link* yang menghubungkan *node* ke *node* selanjutnya pada *FP-tree* yang memiliki *item-name* sama. Tabel *header* terdiri dari dua *field* yaitu *item-name* dan *head of node-link*. *FP-tree* hanya berisi beberapa *item* yang *frequent* yang telah diurutkan berdasarkan nilai *support*-nya.

Setelah *FP-tree* terbentuk, Algoritma *FP-growth* dijalankan secara rekursif untuk membangun *conditional pattern base* dan *conditional FP-tree* untuk setiap *frequent item* yang selanjutnya digunakan untuk men-*generate* semua *frequent itemset*

MIS-Tree

Pada *MIS-tree*, definisi umum dari *minimum support* diubah. Secara umum nilai *minsup* adalah sama untuk semua *item*, namun pada model ini setiap *item* dalam basisdata dapat memiliki nilai *minsup*-nya sendiri-sendiri yang disebut dengan *minimum item support* (*MIS*). Artinya *user* dapat memberikan nilai *MIS* yang berbeda untuk *item* yang berbeda. Dengan memberikan nilai *MIS* yang berbeda untuk setiap *item*, maka hal tersebut akan merefleksikan sifat alamiah dari *item* itu sendiri dan mengakomodasikan adanya variasi frekuensi dalam basisdata.

MIS-tree adalah pengembangan dari struktur *FP-tree* [5]. Ia merupakan struktur pohon yang digunakan untuk menyimpan *frequent item* dengan *multiple minimum support*.

Multiple Item Support

Misalkan $I=\{a_1, a_2, \ldots, a_m\}$ adalah himpunan *item* dan $MIS(a_i)$ menunjukkan nilai *MIS* untuk *item* a_i. Maka nilai *MIS* dari *itemset* $A=\{a_1, a_2, \ldots, a_k\}$ $(1 \le k \le m)$ adalah: $min[MIS(a_1), MIS(a_2), \ldots, MIS(a_k)]$ [5].

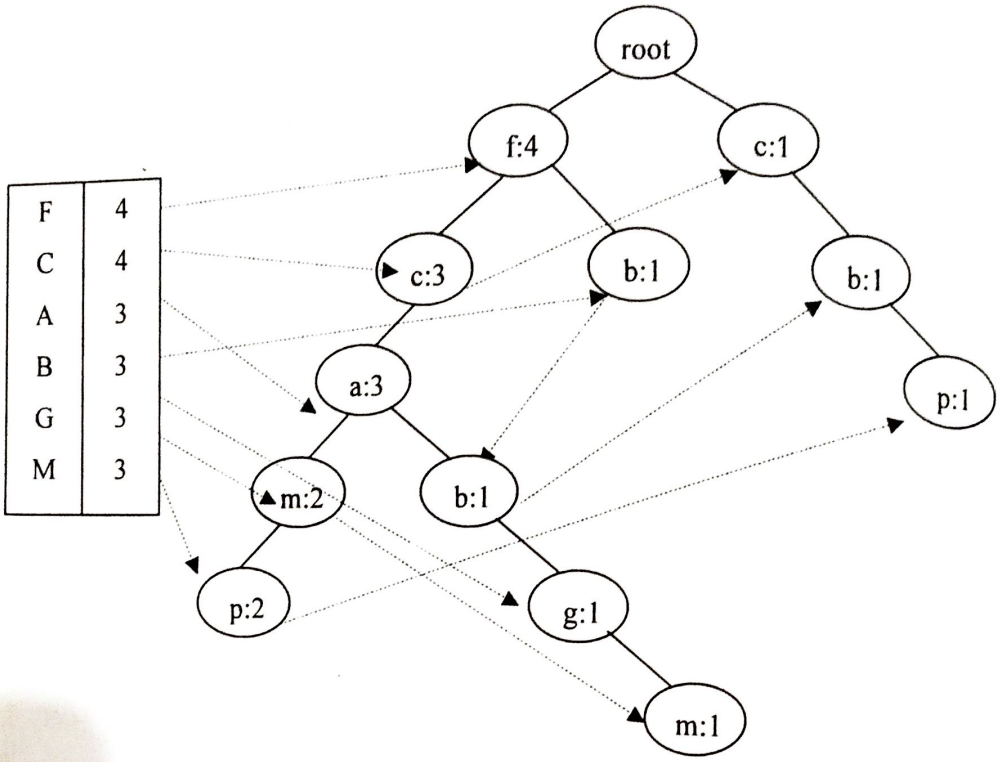
Sebagai contoh dalam basisdata terdapat *item bread, shoes,* dan *clothes*. Nilai *MIS* ditentukan *MIS*(*bread*) = 2%, *MIS*(*shoes*) = 0.1%, dan *MIS*(*clothes*) = 0,2%. Jika nilai *support* dari *itemset* {*clothes, bread*} = 0,15%, maka *itemset* {*clothes, bread*} adalah *infrequent* dikarenakan nilai *MIS* dari *itemset* {*clothes, bread*} = *min*[*MIS*(*clothes*), *MIS*(*bread*)] = 0,2% (lebih besar dari 0,15%).

Tabel 1. Basisdata Transaksi.

TID	*Item*	*Item* terurut sesuai *MIS*
100	a c f m p	f c a m p
200	a c d f m p	f c a m p d
300	a b c f g m	f c a b g m
400	b f i	f b i
500	b c n p	c b p n

Tabel 2. Nilai *MIS* Tiap *Item*.

Item	*F*	*C*	*A*	*B*	*G*	*M*	*P*	*D*	*I*	*N*
MIS	4 (80%)	4 (80%)	3 (60%)	3 (60%)	3 (60%)	3 (60%)	2 (40%)	2 (40%)	2 (40%)	2 (40%)



Gambar 1. *MIS-Tree* Lengkap dan Kompak.

```
ALGORITMA: Frequent Closed Itemsets dengan Multiple Minimum Support
(FCI_MIS)
INPUT: 1. database transaksi, 2. nilai MIS untuk setiap item, 3. MIS-tree
OUTPUT: himpunan lengkap frequent closed itemsets
METODE: Panggil fungsi FCI_MIS(MIS-tree, null)
Procedure FCI_MIS(tree, f)
{   for f anggota tabel header do
    {   bangun proyeksi MIS-tree β dengan prefiks f
        Update tabel header   //Pangkas item yang
                    //tidak frequent dan //Merge pohon setelah //dipangkas
        If tree β ≠null then
  Panggil Get_FCI_MIS(tree β, f, MIS(f))     }}

Procedure Get_FCI_MIS(tree, f, MIS(f))
{   for f anggota header tabel do
      {  buat conditional database dengan prefiks f
         Update tabel header
         If tree β ≠null then
          {    Panggil Get_FCI_MIS(tree β, f, MIS(f))
       If f.support ≠ f.superset.support and f bukan subset dari frequent
       closed itemsets yang telah ditemukan
                    Close=true;    }}}
```

Gambar 2. Algoritma Penggalian *Frequent Closed Itemsets*.

Tabel 3. Karakteristik *Dataset*.

Dataset	# *Tuples*	# *Item*
Mushroom	8124	120
Gazelle	59601	498
T1014D100k	100k	433329169

MIS-tree dilihat dari data transaksi pada Tabel 1 dengan nilai *MIS* tiap *item* pada Tabel 2. Langkah pertama adalah membuat tabel *header* yang berisi nilai *MIS* tiap *item*. Pada *MIS-tree*, *item-item* yang *infrequent* tetap dimasukkan dalam tabel *header*, namun nantinya akan dihapus pada proses *pruning*. Setiap *item* pada tabel *header* dihubungkan ke *node* pada *tree* yang mempunyai nama yang sama melalui *head of node-link*. Setelah tabel *header* terbentuk langkah selanjutnya adalah membuat *root* dengan nilai *null*. Transaksi pertama dibaca dari basisdata untuk membuat cabang pertama dari *MIS-tree*: (*f:1*), (*c:1*), (*a:1*), (*m:1*), (*p:1*)). Transaksi kedua (*f, c, a, m, p, d*) memiliki *prefix* yang sama (*f, c, a, m, p*) dengan jalur yang sudah ada. Sehingga *count* dari setiap *node* sepanjang *prefix* dinaikkan 1 dan sisa *item* (*d*) pada transaksi kedua akan dibuatkan *node* baru sebagai anak dari *node p:2*, dan demikian seterusnya sampai dengan transaksi dalam basisdata habis.

Oleh karena dalam tabel *header* masih terdapat *item* yang tidak *frequent*, maka diperlukan sebuah proses pemangkasan (*pruning*). *Item* yang tidak *frequent*, yaitu {*d, i, n*}, terjadi karena nilai *support*-nya lebih kecil daripada nilai *MIS*. Struktur pohon juga mengalami penyesuaian karena penghapusan *item-item* tersebut dari tabel *header*. Setelah proses pemangkasan, dimungkinkan *node-node* dari *MIS-tree* mempunyai nama yang sama, sehingga perlu dilakukan proses penggabungan (*merge*). Untuk membuat bentuk kompak dilakukan penelusuran pada pohon dan ditemukan bahwa *node* (*m:2*) mempunyai dua anak dengan nama yang sama yaitu *p*. Dilakukan penggabungan dua *node* tersebut menjadi sebuah *node item-name* = *p*, dan *count* diisi dengan jumlah *count* dari kedua *node* tersebut. Bentuk *MIS-tree* yang lengkap dan kompak dapat dilihat pada Gambar 1.

Algoritma *CLOSET*

Ide dasar untuk menggali *frequent closed itemsets* pada Algoritma *CLOSET* adalah dengan menggunakan teknik *devide and conquer*. Caranya adalah sebagai berikut:

1. Membuat *conditional pattern base* dan *conditional FP-Tree* untuk setiap *item* yang *frequent* secara *bottom up* dengan mengacu pada tabel *header*.
2. Mengulangi proses pada Langkah untuk setiap *conditional FP-Tree* yang terbentuk sampai dengan *FP-Tree* kosong atau tinggal memiliki 1 jalur saja.

Conditional pattern base harus dibangun untuk semua *item* yang terdapat pada tabel *header*. Berdasarkan *conditional FP-tree* yang terbentuk, akan ditemukan kandidat *frequent closed itemsets*. Algoritma penggalian *frequent closed itemsets* dapat dituliskan seperti pada Gambar 2.

HASIL DAN PEMBAHASAN

Algoritma penggalian *frequent closed itemsets* ini diimplementasikan menggunakan Bahasa Pemrograman *C++*. Uji coba dilakukan dengan menggunakan *dataset* yang biasa digunakan untuk menguji Algoritma *Association Rule Mining* yaitu basisdata transaksi yang terdapat secara *online*. Untuk data uji coba dipilih dua jenis *dataset* yaitu sebuah *dataset* kecil dan tiga buah *dataset* besar yang diambil dari Pei dkk [4].

Pemberian nilai *MIS* menggunakan formula yang terdapat pada Persamaan (1) [3].

$$MIS(a_i) = \begin{cases} M(a_i) & M(a_i) > MIN \\ MIN & untukyanglain \end{cases} \quad (1)$$

$$M(a_i) = \sigma \times S(a_i)$$

Dimana:

$S(a_i)$ = *support* dari *item* a_i

MIN = nilai *MIS* terkecil dari semua *item*

$\sigma (0 \le \sigma \le 1)$ = parameter yang mengatur relasi antara *MIS* dengan *support* sebenarnya dari *item-item* yang terdapat pada basisdata.

Jika σ = 0, maka hanya ada satu nilai *MIS* untuk semua *item*, yaitu sama dengan *Association Rule Mining* dengan menggunakan *single minsup*. *Dataset* besar yang digunakan yaitu *T1014D100K*, *Mushroom*, dan *Gazelle*. Masing-masing *dataset* tersebut memiliki

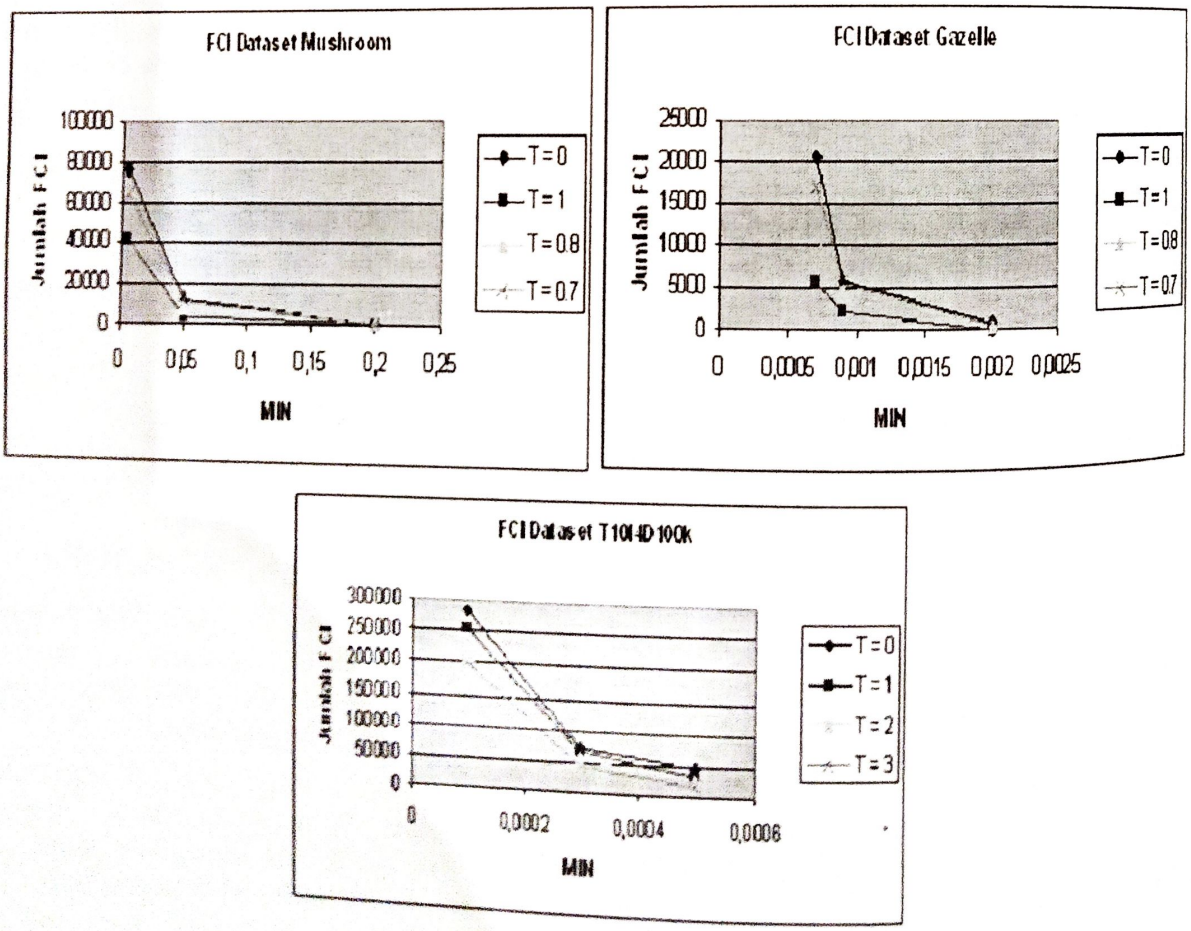
karakteristik yang berbeda. Karakteristik dari *dataset* besar ini dituliskan pada Tabel 3.

Masing-masing *dataset* disimpan dalam sebuah *file teks* dengan ekstensi *txt* dan memiliki format setiap baris pada *file* mewakili sebuah transaksi. Sebuah transaksi terdiri dari kumpulan kode *item* yang dipisahkan oleh spasi. Hasil dari uji coba penggalian *frequent closed itemsets* terhadap ketiga *dataset* tersebut terlihat pada Gambar 3.

Uji coba dilakukan dengan memasukkan beberapa nilai T dan *MIN* yang berbeda. Pemberian nilai T yang tinggi menunjukkan bahwa nilai *MIS* yang digunakan juga tinggi. Nilai $\sigma = 0$ berarti bahwa *minimum support* yang digunakan adalah tunggal dikarenakan semua *MIS* diisi dengan nilai *MIN*. Sedangkan nilai $\sigma = 1$ menunjukkan bahwa *MIS* diisi dengan *support* dari setiap *item*, kecuali untuk *item* yang nilai *support*-nya lebih kecil dari *MIN*.

Hasil uji coba pada Gambar 2 menunjukkan bahwa jumlah *frequent closed itemsets* yang berhasil digali menurun selaras dengan kenaikan nilai *MIS*. Jumlah *frequent closed itemsets* yang dihasilkan adalah tinggi pada saat $T = 0$, yaitu ketika digunakan *single minimum support*. Pada $T = 1$, 0,8; dan 0,7; jumlah *frequent closed itemsets* yang ditemukan menurun. Penurunan yang tidak terlalu signifikan hanya terjadi pada *dataset Gazelle*. Hal tersebut dikarenakan *item* yang ada tersebar secara tidak merata pada *dataset Gazelle*. Berkurangnya jumlah *frequent closed itemsets* menunjukkan bahwa penggunaan *multiple minimum support* mampu menaikkan efisiensi dari proses penggalian aturan asosiasi. Hasil *frequent closed itemsets* ini akan digunakan dalam proses penggalian aturan asosiasi selanjutnya.

Untuk memfasilitasi partisipasi *item* yang jarang, *MIS* untuk *item* jarang harus lebih kecil daripada *support*-nya. Hal ini dapat dilakukan dengan mengeset nilai σ sangat rendah. Namun demikian, proses tersebut mungkin menyebabkan *frequent item* juga akan di-*set* dengan nilai *MIS* yang kecil, sehingga akan menghasilkan sejumlah besar *frequent itemsets*.



Gambar 3. Grafik Jumlah *Frequent Closed Itemsets* dengan Variasi Nilai T pada Ketiga *Dataset*.

SIMPULAN

Pada penelitian ini dikembangkan algoritma untuk menggali *frequent closed itemsets* dengan menggunakan multiple *minimum support*. Algoritma yang dikembangkan menggunakan struktur *Multiple Item Support Tree* (*MIS-tree*) untuk menyimpan pola yang ringkas dan penting tentang *frequent pattern*, dan algoritma *CLOSET* untuk menggali *frequent closed itemsets*. Simpulan yang didapatkan adalah sebagai berikut:

1. *Frequent closed itemsets* yang berhasil digali memiliki jumlah dan kombinasi yang sama dengan *frequent closed itemsets* yang berhasil digali dengan menggunakan struktur *lattice*.
2. Jumlah *frequent closed itemsets* yang berhasil digali menurun ketika digunakan *multiple minimum support*. Hasil ini menunjukkan bahwa penggunaan *multiple minimum support* mampu menaikkan efisiensi, yaitu berkurangnya jumlah *frequent closed itemsets* yang berhasil digali yang akan berpengaruh pada proses penggalian aturan asosiasi selanjutnya.

Untuk pengembangan lebih lanjut dari penelitian ini dapat dititikberatkan pada pemberian nilai *MIS* dengan menggunakan suatu formula tertentu yang mampu mengakomodasi kepentingan *item-item* yang jarang. Sebagai pertimbangan, mungkin dapat juga digunakan standar deviasi dari frekuensi kemunculan data sebagai variabel yang mempengaruhi nilai *MIS*.

DAFTAR PUSTAKA

[1] Agrawal R, Imielinski T, and Swami A. Mining association rules between sets of items in large databases. *In Proc. 1993 ACM-SIGMOD Int. Conf. Management of Data (SIGMOD'93)*. 207-216. 1993.

[2] Pasquier N, Bastide Y, Taouil R, and Lakhal L. Discovering Frequent Closed Itemsets for association Rules. *In Proc. 7th Int. Conf. Database Theory (ICDT'99)*. 398-416. 1999.

[3] Liu B, Hsu W, and Ma Y. Mining Association Rules with Multiple Minimum Support. *Proceedings of the ACM SIGKDD In. Conf. on Knowledge Discovery and Data Mining (KDD-99)*. 337-341. 1999.

[4] Pei J, Han J, and Mao R. *CLOSET: An efficient algorithm for mining frequent closed itemsets. Proceedings of the 2000 ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*. 2000. URL: http://www.cs.sfu.ca/~pei/publications-by-year.htm, diakses tanggal 1 Maret 2009.

[5] Hu YH and Chen YL. Mining association rules with multiple minimum supports: a new mining algorithm and a support tuning mechanism. *Decision Support Systems*. 42: 1-24. 2006.