Jurnal Ilmiah
ILMU KOMPUTER
Universitas Udayana
**Vol. 6, No. 2, September 2013** **ISSN 1979-5661**

# METODE WEIGHTED MAXIMUM CAPTURING UNTUK KLASTERISASI DOKUMEN BERBASIS FREQUENT ITEMSETS

**Gede Aditra Pradnyana[1], Arif Djunaidy[2]**

[1] Jurusan Teknik Informatika, Fakultas Teknologi Informasi, Institut Teknologi Sepuluh Nopember
Kampus ITS Keputih Sukolilo, Surabaya, Jawa Timur
[2] Jurusan Sistem Informasi, Fakultas Teknologi Informasi, Institut Teknologi Sepuluh Nopember
Kampus ITS Keputih Sukolilo, Surabaya, Jawa Timur
Email: gede.aditra@gmail.com

## ABSTRAK

*Klasterisasi dokumen berbasis frequent itemsets merupakan salah satu metode klasterisasi dokumen baru yang dapat digunakan untuk mengatasi masalah tingginya ruang dimensi dari dokumen yang akan diklasterisasi. Teknik maximum capturing merupakan salah satu algoritma klasterisasi dokumen berbasis frequent itemsets yang mampu menghasilkan kualitas klasterisasi yang lebih baik dibandingkan dengan yang dihasilkan oleh algoritma sejenis lainnya. Teknik maximum capturing ini masih memiliki kekurangan atau kelemahan, yaitu: tidak diperhitungkannya bobot suatu kata (item) dalam frequent itemsets saat perhitungan kemiripan dokumen dan dalam proses pembentukan klaster tidak memperhitungkan informasi global dari klaster yang telah terbentuk sebelumnya. Dalam penelitian ini dikembangkan suatu metode baru untuk klasterisasi dokumen dengan berbasis frequent itemsets yaitu metode weighted maximum capturing (WMC), untuk memperbaiki kekurangan teknik maximum capturing sehingga kualitas akurasi hasil klasterisasi dokumen dapat ditingkatkan. Pada metode weighted maximum capturing ini kemiripan dokumen dihitung dengan menggabungkan metode cosine similarity dan jaccard coefficient berdasarkan jumlah frequent itemsets yang sama yang dimiliki sehingga bobot dari item dalam itemsets dapat diperhitungkan, sedangkan pada proses pembentukan klaster diadaptasi algoritma single linkage agglomerative hierarchical clustering. Hasil uji coba dengan data uji Reuters 21578 menunjukkan nilai F-measure dan purity dari metode WMC lebih baik dibandingkan dengan metode awal, yaitu sebesar 0,723 untuk nilai F-measure dengan rasio perbaikan 2,8% dan 0,73 untuk nilai purity dengan rasio perbaikan 3,3%.*

**Kata kunci:** Klasterisasi dokumen, *frequent itemsets*, *weighted maximum capturing*, *cosine similarity*, *jaccard coefficient*

*Document clustering based on frequent itemsets is one of the new document clustering method that can be used to overcome the problem of high-dimensional space of the document being clustered. Maximum capturing technique is one document clustering algorithm based frequent itemsets that can generate better clustering quality compared to those produced by other similar algorithms. The maximum capturing technique still has the lack or weakness, ie: not accounting for the weight of a word (item) in the calculation of frequent itemsets when the document similarity and the cluster formation process does not take into account the global information of the cluster previously formed. In this research developed a new method for clustering documents based frequent itemset namely weighted maximum capturing method (WMC), to correct deficiencies maksimum capturing accuracy so that the quality of document clustering results can be improved. In the weighted maximum capturing method, document similarity is computed by combining the cosine similarity method and Jaccard coefficient based on the same number of frequent itemsets owned so that the weight of items in itemsets can be taken into account, while in the process of constructing cluster adapted single linkage agglomerative hierarchical clustering algorithm. Experimental results show the value of F-measure and purity of WMC method is better than the earlier method, that is equal to 0.723 for the F-measure with improvement ratio of 2.8% and a purity value of 0.73 with improvement ratio 3.3%.*

**Keywords:** document clustering, frequent itemsets, weighted maximum capturing, cosine similarity, jaccard coefficient

## 1. PENDAHULUAN

Peningkatan jumlah dokumen dalam format teks yang cukup signifikan membuat proses pengelompokkan atau klasterisasi dokumen (*document clustering*) menjadi penting. Zhao dan Karypis (2004) mendefinisikan klasterisasi sebagai proses yang membagi suatu set objek menjadi

beberapa jumlah kelompok (klaster) secara spesifik. Klasterisasi dokumen bertujuan membagi dokumen dalam beberapa kelompok sedemikian hingga dokumen-dokumen dalam klaster yang sama (*intra*-klaster) memiliki kesamaan yang tinggi, sementara dokumen-dokumen dalam klaster yang berbeda (*inter*-klaster) memiliki kesamaan yang rendah (Jain, dkk., 1999; Steinbach, dkk., 2000; Zhao dan Karypis, 2004).

Secara umum terdapat dua teknik utama dalam proses klasterisasi yaitu *hierarchical* dan *partitional clustering* (Jain, dkk., 1999). Algoritma-algoritma klasterisasi *hierarchical* dan *partitional* sebenarnya belum sepenuhnya dapat mengatasi tantangan dalam kasus klasterisasi dokumen, misalnya seperti: algoritma tersebut masih mengklaster ruang vektor berdimensi tinggi (*high dimensional vector space*) sepenuhnya dan *centroid* ataupun rata-rata dari klaster yang terbentuk tidak menyediakan deskripsi yang dimengerti mengenai klaster tersebut. Dalam model ruang vektor (*vector space model*), penggunaan kata-kata yang berdiri sendiri (*individual words*) akan mengakibatkan tingginya dimensi vektor. Di lain sisi sebenarnya tidak semua dokumen dalam koleksi juga mengandung semua indeks kata yang digunakan dalam vektor. Permasalahan ini mendorong pengembangan metode baru dalam klasterisasi dokumen yang tidak didasari penggunaan model ruang vektor (Beil dkk., 2002).

*Frequent itemsets* muncul untuk mengatasi masalah ini. Beberapa peneliti (Beil, dkk., 2002; Fung, dkk., 2003; Li, dkk., 2008; Zhang, dkk., 2010) telah menerapkan konsep *frequent itemsets* dalam klasterisasi dokumen. Diawali oleh Beil, dkk (2002) yang menggunakan *frequent* item untuk merepresentasikan kandidat dari suatu klaster. Fung, dkk (2003) kemudian mengenalkan penggunaan global *frequent itemsets* dan klaster *frequent itemsets* untuk kemudian digunakan dalam proses *clustering* dokumen secara hirarki. Penelitian yang dilakukan oleh Li, dkk (2008) menambahkan aspek urutan (*sequence*) item dari suatu *frequent itemsets* untuk digunakan dalam klasterisasi dokumen. Dari percobaan yang dilakukan pada penelitian-penelitian diatas, kualitas hasil klasterisasi dengan konsep *frequent itemsets* lebih baik dari algoritma-algoritma *clustering* konvensional seperti *K-means*, *bisecting K-means*, dan UPGMA. Hasil dari penelitian-penelitian diatas juga membuktikan bahwa konsep *frequent itemsets* yang digunakan dapat melakukan reduksi dimensi dengan baik yang berkibat meningkatnya efisiensi dan skalabilitas proses *clustering* dokumen. Selanjutnya, Zhang, dkk (2010) mengajukan metode *text clustering* dengan *frequent itemsets* yang terbukti lebih baik dari metode-metode berbasis *frequent itemsets* sebelumnya dalam hal akurasi hasil klasterisasi. Metode yang diajukan menggunakan teknik *maximum capturing* dalam membentuk klaster dari *frequent itemsets* yang ada. Dalam proses klasterisasi dokumen, metode yang diajukan oleh Zhang, dkk (2010) ini juga mengadaptasi algoritma *minimum spanning tree* ke dalam metode klasterisasi *partitional*.

Metode klasterisasi dokumen berbasis *frequent itemsets* dengan teknik *maximum capturing* masih memiliki beberapa kekurangan. Pertama, metode *maximum capturing* yang digunakan tidak memperhatikan bobot kata (item) pada *frequent itemset* saat pengalian *frequent itemsets* dan penggukuran kemirpan (*similarity*) antar dokumen dalam membentuk suatu klaster. Pada saat proses pembentukan klaster, kemiripan suatu dokumen dengan dokumen lain pada teknik *maximum capturing* hanya diukur berdasarkan jumlah *frequent itemsets* yang sama yang dimiliki dokumen tersebut. Semakin banyak jumlah *frequent itemsets* yang sama dimiliki maka akan semakin mirip dokumen tersebut. Suatu *frequent itemsets* dikatakan sama apabila memiliki item yang sama tanpa memperhitungkan bobot dari item-item tersebut, padahal bobot ini akan sangat berpengaruh terhadap kemiripan dokumen. Hal ini juga tentu saja akan mengurangi akurasi hasil klasterisasi karena suatu kata yang sering muncul di suatu dokumen seharusnya memiliki bobot yang lebih besar dari kata yang jarang muncul. Kedua, Adaptasi metode *minimum spanning tree* ke dalam proses pembentukan klaster kurang tepat, karena tidak memperhatikan kemiripan global saat melakukan pemutakhiran kemiripan klaster yang baru terbentuk dengan klaster lain. Pada pembentukan suatu klaster dengan adaptasi *minimum spanning tree* jika dokumen *A* berada dalam klaster yang sama dengan dokumen *B* dan *C*, kemudian *A* memiliki kemiripan yang tinggi dengan dokumen *D*, maka dokumen *D* tersebut akan dimasukkan ke dalam klaster yang sama yang juga berisikan dokumen *B* dan *C* tersebut. Hal ini akan mengurangi kualitas hasil klasterisasi jika ternyata dokumen *D* ini memiliki kemiripan yang rendah dengan dokumen *B* dan *C*. Pada penelitian ini diusulkan suatu metode untuk klasterisasi dokumen yang berbasis *frequent itemsets* dengan metode *weighted maximum capturing*, untuk memperbaiki kelemahan dari metode *maximum capturing* sehingga kualitas hasil klasterisasi menjadi lebih baik.

## 2. METODE *WEIGHTED MAXIMUM CAPTURING* (WMC) UNTUK KLASTERISASI DOKUMEN BERBASIS *FREQUENT ITEMSETS*

Dalam penelitian ini dikembangkan suatu metode baru dalam klasterisasi dokumen berbasis *frequent itemsets* dengan metode maximum capturing. Metode ini mengadaptasi algoritma

single lingkage aglomerative hierarchical clustering pada saat proses pembentuk klaster dokumen. Proses pembobotan dilakukan saat perhitungan kemiripan antar dokumen dengan menggabungkan metode *cosine similarity* dan *jaccard coefficient* berdasarkan jumlah *frequent itemsets* yang sama yang dimiliki antar dokumen. Secara umum metode yang diajukan terdiri dari dua proses utama, yaitu proses penggalian *frequent itemsets* dan proses pembentukan klaster dokumen.

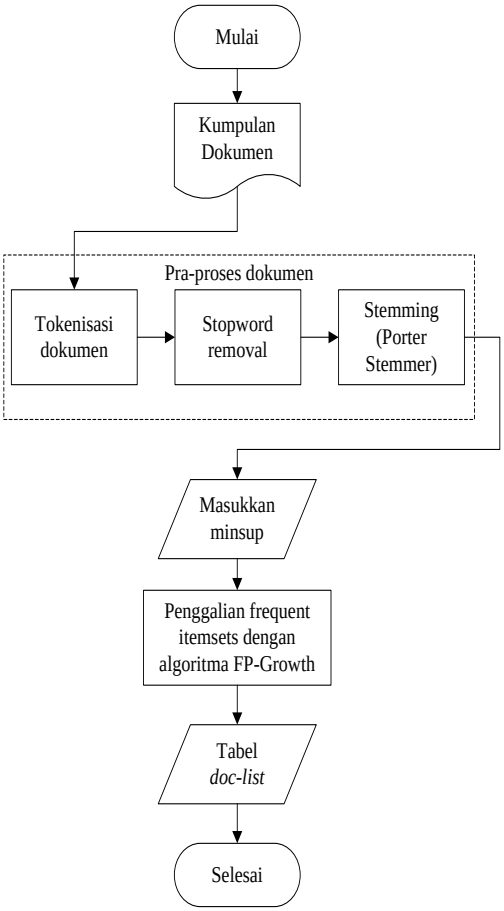
### 2.1 Proses Penggalian *Frequent Itemsets* dari Kumpulan Dokumen

Dalam penelitian ini algoritma *FP-growth* diaplikasikan dalam penggalian *frequent itemsets* dari sekumpulan dokumen tanpa penentuan *minsup*. Algoritma *FP-growth* pada umumnya digunakan pada basis data transaksional. Oleh karena itu, dalam penelitian ini terdapat beberapa proses yang dilakukan pada algoritma *FP-growth* tersebut agar dapat digunakan dalam proses klasterisasi dokumen dengan efektif. Secara keseluruhan alur dari proses penggalian *frequent itemsets* dari sekumpulan dokumen dapat dilihat dalam Gambar 1.

Seperti pada Gambar 1, proses penggalian *frequent itemsets* dari sekumpulan dokumen terdiri dari beberapa tahap. Adapun tahapan dalam menggali *frequent itemsets* dari sekumpulan dokumen dapat diuraikan sebagai berikut:

a. Pra-proses dokumen, yang terdiri dari beberapa tahap atau proses yaitu :
   - Tokenisasi dokumen : proses pemisahan setiap kata (*term*) yang terkandung dalam sebuah dokumen. Setelah proses ini selesai dilanjutkan dengan proses *case folding* yaitu proses penyamaan bentuk (*case*) suatu kata. Tidak semua kata dalam dokumen konsisten dalam pengunaan huruf capital, oleh karena itu proses *case folding* dibutuhkan untuk mengkonversi keseluruhan kata menjadi suatu bentuk standar yang umumnya berupa huruf kecil (*small case*).
   - *Stopwords removal* : proses penghilangan kata-kata tidak penting (*stopwords*) dari suatu dokumen. Proses penghilangan dilakukan dengan pencocokan kata-kata yang ada dalam dokumen terhadap sebuah tabel yang berisikan daftar kata-kata tidak penting (*stoplist*).
   - *Stemming*: proses pengembalian suatu kata berimbuhan ke bentuk dasarnya (*root word*). Pada penelitian ini digunakan algoritma *Porter Stemmer* untuk bahasa Inggris dalam mencari kata dasar (*root word*) dari suatu kata.

b. Setelah tahap pra-proses dokumen selesai, selanjutnya akan dilakukan penggalian *frequent itemsets* dengan nilai *minimum support* (minsup) yang dimasukkan pengguna




Gambar 1. Diagram Alir Proses Penggalian Frequent Itemsets

c. Proses penggalian *frequent itemsets* pada metode yang dikembangkan menggunakan algoritma *FP-growth* (Han, dkk.,2000). Struktur data yang digunakan untuk mencari *frequent itemset* dengan algoritma *FP-growth* adalah perluasan dari penggunaan sebuah pohon *prefix*, yang biasa disebut *FP-tree*. Dengan menggunakan *FP-tree*, algoritma *FP-growth* dapat langsung mengekstrak *frequent itemsets* dari *FP-tree* yang telah terbentuk dengan menggunakan konsep *divide* and *conquer*. Proses implementasi algoritma *FP-growth* pada penelitian ini sesuai dengan Borgelt dalam (Borgelt, 2005).

d. Proses selanjutnya adalah proses pembuatan tabel *Doc-List* berdasarkan *frequent itemsets* yang berhasil digali. Tabel *Doc-List* dibuat dengan cara mencocokan item-item yang terdapat pada setiap *frequent itemsets* yang dihasilkan dari proses penggalian dengan item-item yang dimiliki oleh sebuah dokumen. Tabel *Doc-List* yang dihasilkan selanjutnya akan digunakan dalam proses pembentukan klaster dokumen.

### 2.2 Proses Pembentukan Klaster Dokumen

Setelah *frequent itemsets* dari masing-masing dokumen berhasil digali dan telah dihasilkan tabel *Doc-List*, selanjutnya dilakukan proses pembentukan klaster dokumen berdasarkan tabel *Doc-List* yang diperoleh. Pada proses pembentukan klaster dalam penelitian ini dilakukan perbaikan terhadap metode *maximum capturing* yang mengadaptasi metode *minimum spanning tree* agar akurasi hasil klasterisasi dapat ditingkatkan. Perbaikan dilakukan terhadap dua sub-proses utama dari proses pembentukan klaster, yaitu sub-proses perhitungan kemiripan (*similarity*) antar dokumen berdasarkan tabel *Doc-List* yang terbentuk dan sub-proses rekonstruksi klaster yang akan dibentuk. Secara keseluruhan alur dari proses pembentukan klaster berdasarkan tabel *Doc-List* dapat dilihat dalam Gambar 2.

Seperti yang terlihat pada Gambar 2, proses pembentukan klaster berbasis *frequent itemsets* dengan metode *weighted maximum capturing* terdiri dari beberapa tahap, yaitu:

a. Perhitungan kemiripan antar dokumen. Perhitungan kemiripan antar dokumen menggunakan metode yang merupakan penggabungan antara *jaccard coefficient* sebagai metode *asymetrical binary similarity* dengan metode *cosine similarity*. *Jaccard coefficient* digunakan untuk melihat kemiripan jumlah *frequent itemsets* yang dimiliki dari pasangan dokumen, sedangkan metode *cosine similarity* digunakan untuk menghitung kemiripan antar *frequent itemsets* tersebut berdasarkan bobot dari tiap item dalam *itemset* tersebut. Metode *cosine similarity* merupakan metode yang digunakan untuk menghitung tingkat kemiripan antar dua buah objek. Secara umum penghitungan metode ini didasarkan pada *vector space similarity measure.* Metode *cosine similarity* ini menghitung kemiripan antara dua buah objek yang dinyatakan dalam dua buah vektor dengan menggunakan *keywords* (kata kunci) dari sebuah dokumen sebagai ukuran.

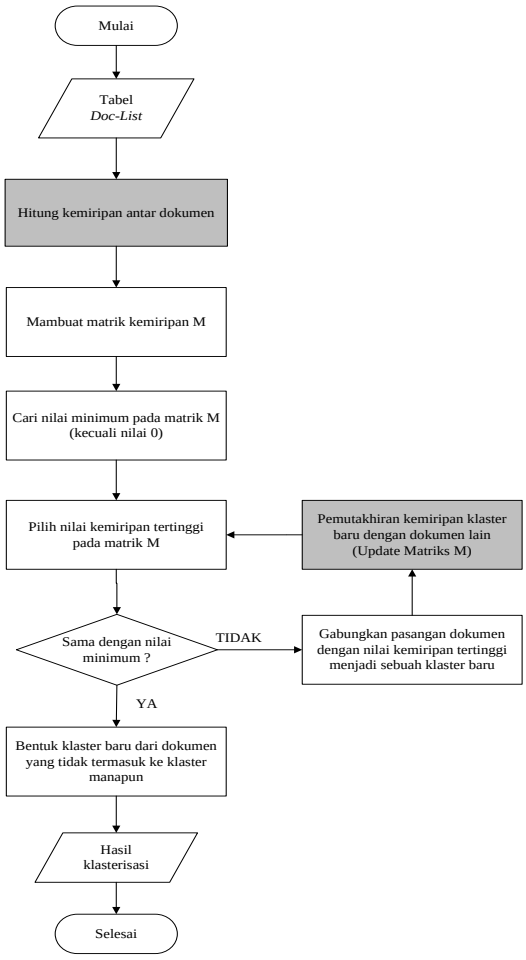
   Jika *Q* adalah vektor dokumen pertama dan *D* adalah vektor dokumen kedua, yang merupakan dua buah vektor dalam ruang berdimensi-*n*, dan $\theta$ adalah sudut yang dibentuk oleh kedua vektor tersebut. Maka

$$Q \bullet D = |Q||D|\cos\theta, \quad (1)$$

dimana Q ● D adalah hasil perkalian dalam (*inner product*) kedua vektor, sedangkan

$$|D| = \sqrt{\sum_{i=1}^{n} D_i^2} \quad (2)$$

$$|Q| = \sqrt{\sum_{i=1}^{n} Q_i^2} \quad (3)$$




Gambar 2. Proses Pembentukan Klaster dengan Metode *Weighted Maximum Capturing*

merupakan panjang vektor atau jarak *euclidean* suatu vektor dengan titik nol. Metode pengukuran kemiripan ini memiliki beberapa keuntungan, yaitu adanya normalisasi terhadap panjang dokumen. Hal ini memperkecil pengaruh panjang dokumen. Jarak *euclidean* (panjang) kedua vektor digunakan sebagai faktor normalisasi. Hal ini diperlukan karena dokumen yang panjang cenderung mendapatkan nilai yang besar dibandingkan dengan dokumen yang lebih pendek. Perhitungan *cosine similarity* yang memperhitungkan perhitungan pembobotan kata pada suatu dokumen dapat dinyatakan dengan perumusan :

$$CosSim(d_i, q_i) = \frac{q_i \bullet d_i}{|q_i||d_i|} = \frac{\sum_{j=1}^{t}(q_{ij}.d_{ij})}{\sqrt{\sum_{j=1}^{t}(q_{ij})^2.\sum_{j=1}^{t}(d_{ij})^2}} \quad (4)$$

Dimana $q_{ij}$ adalah bobot istilah *j* pada dokumen $i = tf_{ij} * idf_j$ dan $d_{ij}$ adalah bobot istilah *j* pada dokumen $i = tf_{ij} * idf_j$.

Metode *jaccard coefficient* adalah sebuah metode yang sering digunakan untuk membandingkan kemiripan, perbedaan, dan jarak suatu set data. *Jaccard coefficient* atau yang terkadang disebut *tanimoto coefficient* membandingkan bobot dari jumlah istilah atau kata yang muncul bersama pada dokumen dengan bobot istilah atau kata yang tidak muncul pada kedua dokumen. Perhitungan kemiripan dua buah dokumen dengan *jaccard coefficien*t dapat dilihat pada persamaan berikut (Niwattanakul, dkk., 2013):

$$Sim(Dok_1, Dok_2) = \frac{Dok_1 \cap Dok_2}{Dok_1 \cup Dok_2} \quad (5)$$

*Dok1*∩*Dok2* berisikan jumlah atribut yang sama yang dimiliki oleh kedua dokumen (*intersection*), sedangkan *Dok1*∪*Dok2* berisikan jumlah kata yang dimiliki oleh kedua dokumen ditambah dengan jumlah kata *Dok2* yang tidak dimiliki oleh *Dok1* ditambah jumlah kata *Dok1* yang tidak dimiliki oleh *Dok2* (*union*). *Jaccard coefficient* merupakan metode pengukuran kemiripan dari suatu atribut biner yang tidak simetris. Melihat suatu objek dalam format biner akan memungkinkan pengguna untuk mengukur kemiripan menjadi lebih mudah dengan menentukan objek A dan B terdiri dari *n* fitur kemudian menggunakan ukuran dari jumlah properti yang sama yang dimiliki objek A dan B.

Tabel 1 memperlihatkan contoh tabel *Doc-List* dari sekumpulan dokumen hasil dari proses penggalian. Dalam Tabel 1 kolom *frequent itemsets* berisikan informasi *frequent itemsets* dari setiap dokumen beserta frekuensi kemunculan item-item dari *frequent itemsets* tersebut pada dokumen yang sesuai.

Tabel 1 Contoh Tabel *Doc-List* dari Kumpulan Dokumen

| ID Dokumen | *Frequent Itemsets* |
|---|---|
| *Dok1* | *{$I_1$: 5 , $I_3$: 2, $I_4$: 3}, { $I_2$: 4, $I_5$: 3}* |
| *Dok2* | *{$I_1$ : 4, $I_3$: 3, $I_4$: 4}, { $I_1$: 4, $I_2$: 5}, { $I_2$: 5, $I_5$: 8}* |
| *Dok3* | *{$I_2$ : 3, $I_4$: 5, $I_5$: 4}, { $I_1$: 3, $I_2$: 5,}* |
| *Dok4* | *{ $I_2$: 4, $I_5$: 6,} { $I_1$:3, $I_2$:4,}* |
| *Dok5* | *{$I_1$: 2 , $I_3$: 3, $I_4$: 4}* |

Berdasarkan contoh dalam Tabel 1, proses perhitungan kemiripan antara dokumen *Dok1* dengan dokumen *Dok2* dapat diuraikan sebagai berikut:

a) Perhitungan bobot dari masing-masing item dalam *frequent itemset* suatu dokumen sesuai dengan metode pembobotan TF-IDF seperti pada persamaan (6).

$$w_{i,j} = tf_{i,j}\, x\, log_2\left(\frac{N}{df_j}1\right) \quad (6)$$

dimana $w_{ij}$ adalah bobot *term j* pada dokumen *i*, $tf_{ij}$ adalah frekuensi *term* j pada dokumen *i*, *N* adalah jumlah total dokumen yang diproses, dan $df_j$ adalah jumlah dokumen yang memiliki *term j* didalamnya. Sebagai contoh, berikut ini adalah perhitungan bobot item $I_1$ yang dimiliki oleh *Dok1*:

$$w_{i,j} = tf_{i,j}\, x\, log_2\left(\frac{N}{df_j} + 1\right)$$
$$= 5\, x\, log_2\left(\frac{5}{5} + 1\right) = 1,51 \quad (2.9)$$

Perhitungan serupa juga dilakukan pada seluruh item yang terdapat dalam *Dok2*. Hasil perhitungan bobot dari masing-masing item yang terdapat dalam *Dok1* dan *Dok2* terlihat dalam Tabel 2.

Tabel 2. Bobot Item pada *Dok1* dan *Dok2*

| Item | Bobot pada *Dok1* | Bobot pada *Dok2* |
|---|---|---|
| $I_1$ | 1,51 | 1,20 |
| $I_2$ | 1,40 | 1,76 |
| $I_3$ | 0,85 | 1,28 |
| $I_4$ | 1,06 | 1,40 |
| $I_5$ | 1,06 | 2,82 |

b) Perhitungan kemiripan antara *frequent itemsets* yang sama dari masing-masing dokumen dengan metode *cosine similarity* sesuai persamaan (2).
Berdasarkan contoh yang terdapat pada Tabel 1 dan Tabel 2 , kemiripan *frequent itemsets* yang terdapat pada *Dok1* dan *Dok2* dapat dihitung sebagai berikut:

$$Cos(Q,D)_{134} = \frac{(1,51\, x\, 1,20 + 0,85\, x\, 1,28 + 1,06 x 1,40)}{\sqrt{1,51^2 + 0,85^2 + 1,06^2}\, x \sqrt{1,20^2 + 1,28^2 + 1,40^2}}$$
$$= \frac{4,384}{2,031\, x\, 2,245} = \frac{4,384}{4,56} = 0,96$$
$$CosSim(Q,D)_{25} = \frac{(1,40\, x\, 1,76 + 1,06\, x\, 2.82)}{\sqrt{1,40^2 + 1,06^2}\, x \sqrt{1,76^2 + 2,82^2}}$$
$$= \frac{5,453}{1,756\, x\, 3,324} = \frac{5,453}{5,836} = 0,93$$

c) Perhitungan kemiripan antara *Dok1* dan *Dok2* dengan metode *jaccard coefficient*. Berdasarkan contoh dalam Tabel 1, kemiripan antara dokumen *Dok1* dan dokumen *Dok2* dapat dihitung sebagai berikut:

$$Sim(Dok_1, Dok_2) = \frac{Dok_1 \cap Dok_2}{Dok_1 \cup Dok_2} = \frac{0,96 + 0,93}{5}$$
$$= 0,378$$

a. Pembuatan matrik *similarity M*. Setelah semua kemiripan antar dokumen diperoleh, proses selanjutnya adalah membuat matriks kemiripan. Matriks *similarity* atau kemiripan *M* adalah

suatu matrik yang menggambarkan nilai kemiripan antara dokumen satu dengan yang lainnya. Berikut ini adalah contoh matriks kemiripan dari lima buah dokumen:

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | 0 | 5 | 4 | 5 | 10 |
| 2 | | 0 | 8 | 6 | 9 |
| 3 | | | 0 | 7 | 3 |
| 4 | | | | 0 | 9 |
| 5 | | | | | 0 |

b. Pencarian nilai minimum pada matriks *M*, kecuali nilai 0. Nilai minimum pada contoh matriks diatas adalah 3.
c. Penggabungan dokumen dengan kemiripan tertinggi menjadi sebuah klaster baru. Pada contoh matriks *M* di atas, nilai kemiripan tertinggi adalah 10, sehingga dokumen 1 dan dokumen 2 dapat digabung menjadi klaster (1,5).
d. Pemutakhiran (*update*) kemiripan klaster yang baru terbentuk dengan klaster lain. Kemiripan klaster baru dengan suatu klaster adalah nilai kemiripan terendah antara anggota dari klaster baru tersebut. Berdasarkan contoh klaster baru yang terbentuk pada langkah d diatas, berikut adalah contoh perhitungan kemiripan klaster baru dengan klaster lain:

$$d_{(15)2} = \min \{d_{12}, d_{52}\} = \min \{5, 9\} = 5$$
$$d_{(15)3} = \min \{d_{13}, d_{53}\} = \min \{4, 3\} = 3$$
$$d_{(15)4} = \min \{d_{14}, d_{54}\} = \min \{5, 9\} = 5$$

Seperti yang terlihat pada perhitungan diatas, kemiripan antara klaster baru (1,5) dengan dokumen 2 adalah nilai kemiripan minimum antara dokumen 1 dengan dokumen 2 (yaitu 5) dan dokumen 5 dengan dokumen 2 (yaitu 9). Jadi nilai kemiripan antara klaster baru (1,5) dengan dokumen 2 adalah 5.
e. Pemutakhiran matrik kemiripan *M*. Setelah terbentuk klaster baru dan kemiripan antara klaster baru tersebut dengan klaster atau dokumen lain diperoleh, dilakukan proses pemutakhiran matrik kemiripan *M*. Berikut adalah contoh matrik hasil pemutakhiran matrik M yang terdapat pada langkah b di atas.

| | 2 | 3 | 4 | 1,5 |
|---|---|---|---|---|
| 2 | 0 | 8 | 6 | 5 |
| 3 | | 0 | 7 | 3 |
| 4 | | | 0 | 5 |
| 1,5 | | | | 0 |

f. Ulangi langkah d-f sampai di temukan nilai kemiripan maksimum yang sama dengan nilai kemiripan minimum yang ditemukan pada langkah c. Dari contoh matrik M yang terdapat pada langkah f diatas, berikut adalah proses lengkap pembuatan klaster yang dilakukan:

- Pemilihan nilai kemiripan terbesar pada matrik *M*. Terpilih kemiripan antara dokumen 2 dan dokumen 3, sehingga terbentuk klaster (2,3)

$$max\ (d_{ik}) = d_{23} = 8$$

- Pemutakhiran kemiripan klaster (2,3) dengan klaster atau dokumen lain.

$$d_{(23)4} = \min \{d_{24}, d_{34}\} = \min \{6, 7\} = 6$$
$$d_{(23)(15)} = \min \{d_{2(15)}, d_{3(15)}\} = \min \{5, 3\} = 3$$

- Pemutakhiran matrik M. Setelah diperoleh kemiripan antara klaster baru (2,3) dengan klaster atau dokumen lain, dilanjutkan dengan proses pemutakhiran matrik *M*.

| | 2,3 | 4 | 1,5 |
|---|---|---|---|
| 2,3 | 0 | 6 | 3 |
| 4 | | 0 | 5 |
| 1,5 | | | 0 |

- Pemilihan nilai kemiripan terbesar pada matrik M di atas. Terpilih kemiripan antara klaster (2,3) dengan dokumen 4, sehingga terbentuk klaster (2,3,4)

$$max\ (d_{ik}) = d_{(23)4} = 6$$

- Pemutakhiran kemiripan klaster (2,3,4) dengan klaster atau dokumen lain.

$$d_{(234)(15)} = \min \{d_{(23)(15)}, d_{4(15)}\} = \min \{3, 5\} = 3$$

- Oleh karena nilai kemiripan maksimum sama dengan nilai kemiripan minimum maka proses pembentukan klaster dihentikan.

g. Apabila terdapat dokumen yang belum termasuk klaster manapun, maka dokumen-dokumen ini digunakan untuk membentuk klaster baru.

## 3. HASIL DAN PEMBAHASAN

Pada bagian ini akan dilakukan evaluasi kinerja dari metode klasterisasi dokumen yang diajukan dengan melihat hasil klasterisasi dan perbandingan dengan algoritma sebelumnya. Metode yang diajukan diimplementasikan dengan bahasa pemrograman Java pada sebuah komputer dengan spesifikasi Intel *Core* i7 1.9 GHz, 4GB.

### 3.1 Data Uji

Pengukuran kinerja dari algoritma dalam penelitian ini dilakukan dengan menggunakan data

uji *Reuters 21578 Dataset*. Set data uji coba ini diperoleh dari *UCI Knowledge Discovery in Database* (http://kdd.ics.uci.edu/databases/). Set data *Reuters 21578* merupakan set data dokumen dalam Bahasa Inggris yang terdiri dari 21.578 dokumen, yang dihimpun dari *Reuters Newswire* pada tahun 1987. Tabel 3 memperlihatkan karakteristik dari data uji *Reuters 21578* yang digunakan dalam penelitian ini. Sedangkan Tabel 4 memperlihatkan kategori beserta jumlah dokumen untuk masing-masing kategori dari data uji *Reuters 21578*.

Tabel 3. Karakteritik Data Uji *Reuters 21578*

| Karakteristik | *Reuters 21578* |
|---|---|
| Jumlah Item / Kata | 13.639 |
| Rerata Panjang Dokumen | 89 |
| Jumlah Record | 5.000 |
| Jumlah Kategori | 10 |

Tabel 4. Kategori Data Uji *Reuters 21578*

| Topik / Kategori | Jumlah Dokumen |
|---|---|
| Acq | 1.440 |
| Coffee | 119 |
| Crude | 379 |
| Earn | 1.552 |
| Interest | 343 |
| Money-fx | 317 |
| Money-supply | 108 |
| Ship | 216 |
| Sugar | 146 |
| Trade | 380 |
| Total Dokumen | 5.000 |

### 3.2 Metode Evaluasi Hasil Klasterisasi

Untuk evaluasi dari hasil proses klasterisasi dokumen digunakan pengukuran nilai *F-measure* dan *Purity*. *F-measure* merupakan kombinasi *harmonic* dari nilai *recall* dan *precision* yang digunakan dalam sistem temu kembali informasi. Menggunakan data uji yang sudah dijelaskan sebelumnya, setiap klaster yang dihasilkan dapat dianggap sebagai hasil dari suatu *query*, sedangkan setiap kumpulan dokumen yang belum diklasifikasi dapat dianggap sebagai kumpulan dokumen yang diharapkan dari *query* tersebut. Jadi nilai *precision P(i,j)* dan nilai *recall R(i,j)* pada setiap klaster ke-*j* dari kelas ke-*i* dapat dihitung.

Jika $n_i$ adalah jumlah dari anggota kelas ke-*i*, $n_j$ adalah jumlah anggota dari klaster *ke-j*, dan $n_{ij}$ adalah jumlah dari anggota kelas *ke-i* yang berada pada klaster *ke-j*, maka *P(i,j)* dan *R(i,j)* dapat dihitung dengan persamaan (X) dan (X) berikut (Dalli, 2003):

$$P(i,j) = \frac{n_{ij}}{n_j}, \quad (7)$$

$$R(i,j) = \frac{n_{ij}}{n_i}. \quad (8)$$

Nilai *F-measure* kelas ke-*i* pada klaster ke-*j* dan nilai keseluruhan *F-measure* dinyatakan pada persamaan-persamaan berikut:

$$F(i,j) = \frac{2 \times P(i,j) \times R(i,j)}{P(i,j) + R(i,j)} \quad (9)$$

$$F = \sum_j \frac{n_i}{n} \max\{F(i,j)\}, \quad (10)$$

dimana *n* adalah jumlah semua dokumen, $n_i$ adalah jumlah dokumen pada kelas ke-*i*, dan *max{F(i,j)}* adalah nilai *F(i,j)* terbesar yang ditemukan pada kelas ke-*i* untuk keseluruhan klaster *ke-j*. Secara umum, nilai *F-measure* yang tinggi merepresentasikan hasil klasterisasi yang baik (Steinbach, dkk., 2000).

Nilai *Purity* dari suatu klaster merepresentasikan bagian dari suatu klaster sesuai dengan kelas terbesar dari suatu dokumen dimasukkan ke dalam klaster tersebut, maka *Purity* dari klaster ke-*j* didefinisikan sebagai berikut (Zhao dan Karypis, 2004):

$$Purity(j) = \frac{1}{n_j} \max_i n_{ij} \quad (11)$$

Nilai *Purity* secara keseluruhan dari proses klasterisasi merupakan penjumlahan dari setiap nilai *Purity* klaster:

$$Purity = \sum_j \frac{n_j}{n} Purity\ (j). \quad (12)$$

Secara umum, nilai *Purity* yang lebih tinggi menunjukkan hasil klasterisasi yang lebih baik.

### 3.3 Hasil Uji Coba dan Analisis

Dalam uji coba ini akan dibandingkan metode *weighted maximum capturing* (WMC) yang dikembangkan dalam penelitian ini dengan algoritma klasterisasi dokumen berbasis *frequent itemsets* dengan teknik *maximum capturing* (MC) dalam hal akurasi hasil klasterisasi. Perbandingan dilakukan dengan menggunakan data uji coba *Reuters-21578* yang berjumlah 5.000 dokumen. Untuk mengetahui perbandingan akurasi hasil klasterisasi, dibandingkan nilai *F-measure* dan *purity* dari masing-masing algoritma dengan menggunakan jumlah dokumen 2.500 sampai 5.000 dengan interval 250 dokumen.

Proses klasterisasi dokumen yang berbasis *frequent itemsets* memerlukan nilai minimum support (minsup) dalam menentukan suatu itemset merupakan *frequent itemsets* atau tidak. Penentuan nilai minsup yang tepat akan berpengaruh terhadap akurasi hasil klasterisasi. Untuk menemukan nilai minsup yang tepat, maka dilakukan dilakukan uji coba parameter *minsup* dengan rentang nilai 0,01 sampai 0,03 dengan interval nilai 0,005 pada 1.000 dokumen uji coba. Hasil uji coba parameter *minsup* pada pada metode WMC dan MC dapat dilihat pada Tabel 5. Nilai parameter minsup optimal adalah nilai minsup yang menghasil nilai *F-*

*measure* tertinggi. Seperti yang terlihat pada Tabel 5, nilai parameter minsup optimal untuk metode MC adalah 0,015 sedangkan untuk metode WMC adalah 0,01.

Tabel 5. Hasil Uji Coba Parameter *Minimum Support* (*minsup*)

| *minsup* | *F-measure* Metode MC | *F-measure* Metode WMC |
|---|---|---|
| 0,010 | 0,6716 | **0,6943** |
| 0,015 | **0,6819** | 0,6786 |
| 0,020 | 0,5908 | 0,5683 |
| 0,025 | 0,5467 | 0,5948 |
| 0,030 | 0,5410 | 0,6237 |

Dengan menggunakan nilai parameter minsup yang optimal ini selanjutnya dibandingkan akurasi hasil klasterisasi metode WMC yang dikembangkan dengan metode sebelumnya (MC) dalam menangani sejumlah dokumen yang diberikan. Tabel 6 memperlihatkan hasil uji coba perbandingan akurasi hasil klasterisasi dari masing-masing algoritma yang dilihat dari nilai *F-measure* dan *Purity*. Gambar 3 menunjukkan waktu komputasi yang diperlukan masing-masing algoritma dalam menangani sejumlah data yang diberikan.

Dalam Tabel 6 terlihat bahwa nilai *F-measure* dari algoritma WMC selalu lebih tinggi dari algoritma MC pada setiap jumlah dokumen yang diberikan. Pada Tabel 6 dapat dilihat nilai rata-rata *F-measure* dan *purity* dari proses klasterisasi dokumen dengan algoritma *weighted maximum capturing* (WMC) lebih besar dari nilai rata-rata *F-measure* dan *purity* yang dihasilkan oleh proses klasterisasi dokumen dengan algoritma *maximum capturing* (MC). Dimana nilai rata-rata (*mean*) *F-measure* yang dihasilkan oleh algoritma WMC adalah sebesar 0,723 yang lebih tinggi 0,02 dibandingkan algoritma MC (0,703). Rata-rata nilai *F-measure* yang lebih tinggi menunjukkan bahwa akurasi hasil klasterisasi dari algoritma WMC lebih baik dibandingkan algoritma MC. Sejalan dengan nilai *F-measure*, nilai *purity* dari algoritma WMC juga selalu lebih tinggi dibandingkan algoritma MC pada setiap jumlah dokumen yang diberikan. Dimana nilai rata-rata *purity* yang dihasilkan oleh algoritma WMC adalah sebesar 0,730 yang lebih tinggi 0,023 dibandingkan algoritma MC (0,707). Nilai *purity* yang tinggi mencerminkan tingkat kemurnian suatu klaster semakin baik, yang berarti klaster tersebut mengandung sebagian besar dokumen yang memang seharusnya menjadi bagian dari klaster tersebut.

Untuk mengetahui rasio perbaikan akurasi hasil klasterisasi algoritma WMC terhadap algoritma MC, maka dihitung nilai *improvement ratio* (IR) sebagai berikut:

$$IR_F = \frac{F^{\,metode\ WMC} - F^{\,metode\ MC}}{F^{\,metode\ MC}}$$

$$= \frac{0{,}723 - 0{,}703}{0{,}703} = \frac{0{,}02}{0{,}703} = 0{,}028$$

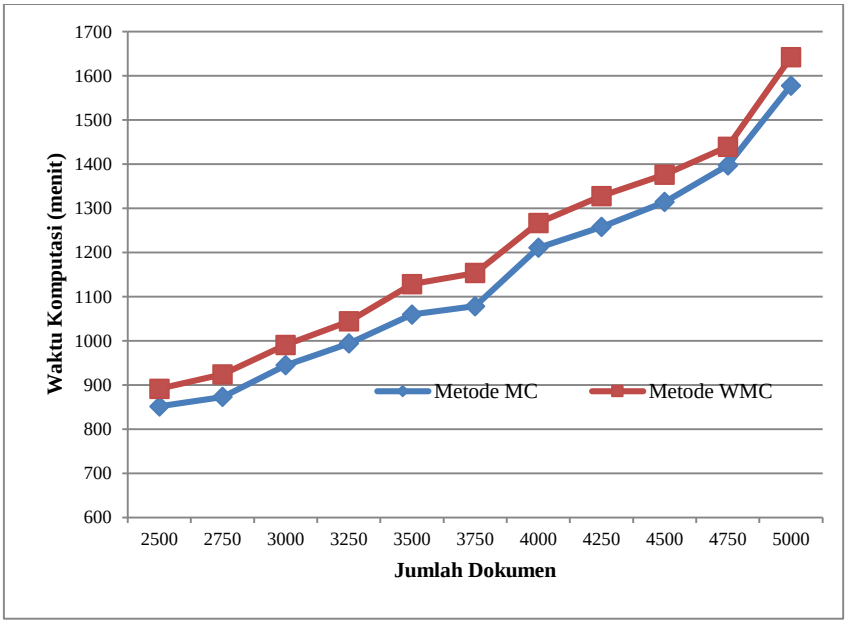
$$IR_{purity} = \frac{Purity^{\,metode\ WMC} - Purity^{\,metode\ MC}}{Purity^{metode\ MC}}$$

$$= \frac{0{,}730 - 0{,}707}{0{,}707} = \frac{0{,}023}{0{,}707} = 0{,}033$$

Berdasarkan perhitungan IR di atas, algoritma WMC yang dikembangkan dalam penelitian ini melakukan perbaikan nilai *F-measure* algoritma awal sebesar 2,8%, dan melakukan perbaikan nilai *purity* algoritma awal sebesar 3,3%. Perubahan yang dilakukan seperti perhitungan kemiripan dokumen dengan memperhatikan bobot dari *frequent itemsets*, dan proses klasterisasi yang memperhitungkan kemiripan global terbukti berhasil meningkatkan akurasi hasil klasterisasi.

Tabel 6. Perbandingan Nilai *F-measure* dan Purity dari Masing-Masing Metode

| Jumlah Dokumen | F-Measure | | Purity | |
|---|---|---|---|---|
| | Metode MC | Metode WMC | Metode MC | Metode WMC |
| 2.500 | 0,730 | 0,758 | 0,713 | 0,747 |
| 2.750 | 0,711 | 0,731 | 0,729 | 0,732 |
| 3.000 | 0,675 | 0,677 | 0,682 | 0,704 |
| 3.250 | 0,726 | 0,749 | 0,735 | 0,744 |
| 3.500 | 0,671 | 0,694 | 0,677 | 0,695 |
| 3.750 | 0,662 | 0,699 | 0,707 | 0,712 |
| 4.000 | 0,742 | 0,761 | 0,738 | 0,769 |
| 4.250 | 0,688 | 0,694 | 0,692 | 0,713 |
| 4.500 | 0,722 | 0,724 | 0,674 | 0,725 |
| 4.750 | 0,703 | 0,722 | 0,718 | 0,737 |
| 5.000 | 0,698 | 0,742 | 0,711 | 0,748 |
| Rata-Rata | 0,703 | 0,723 | 0,707 | 0,730 |




Gambar 3. Grafik Perbandingan Waktu Komputasi Metode MC dan Metode WMC

Pada Gambar 3 terlihat bahwa waktu komputasi yang dibutuhkan dalam proses klasterisasi dokumen berbanding lurus dengan jumlah dokumen yang digunakan. Semakin banyak jumlah dokumen yang akan diklaster maka semakin lama waktu komputasinya, begitu juga sebaliknya. Seperti yang terlihat juga pada Gambar 5, waktu komputasi dari metode WMC lebih lama jika dibandingkan dengan metode MC. Hal ini disebabkan tingginya kompleksitas komputasi dari metode WMC yang dikembangkan yang memerlukan waktu cukup lama dalam melakukan klasterisasi dokumen. Selain karena mengadaptasi metode klasterisasi secara hirarki yang memiliki kompleksitas tinggi, kompleksitas komputasi yang cukup tinggi pada metode WMC juga disebabkan oleh adanya berbagai proses tambahan untuk melakukan perhitungan bobot tiap item pada masing-masing dokumen dan pengukuran kemiripan dengan kombinasi dua metode.

## 4. KESIMPULAN

Pada paper ini telah diusulkan sebuah metode baru yaitu metode *weighted maximum capturing* untuk klasterisasi dokumen berbasis *frequent item-sets*. Berdasarkan uji coba yang dilakukan, kualitas hasil klasterisasi dokumen oleh algoritma WMC yang dikembangkan lebih baik dibandingkan algoritma awal (algoritma MC). Perbaikan atau perubahan yang dilakukan pada algoritma WMC telah terbukti mampu meningkatkan akurasi hasil klasterisasi. Untuk data uji *Reuters 21578*, rata-rata nilai *F-measure* dari algoritma WMC sebesar 0,723 yang meningkat 0,02 dari algoritma awal dengan rasio perbaikan 2,8%. Sedangkan rata-rata nilai *purity* dari algoritma WMC sebesar 0,73 yang meningkat 0,023 dari algoritma awal dengan rasio perbaikan 3,3%. Dari sisi waktu komputasi, waktu komputasi dari metode WMC dalam melakukan klasterisasi sejumlah dokumen lebih lama jika dibandingkan metode MC.

Lamanya waktu komputasi metode MC disebabkan oleh tingginya kompleksitas komputasi dari metode WMC saat proses pembentukan klaster. Oleh karena itu pengembangan lebih lanjut metode WMC ini masih sangat mungkin dilakukan. Misalnya dengan menggati metode proses pembentukan klaster dengan mengadaptasi metode klasterisasi partisi seperti *k-means*, *bisecting k-means*, dan lain-lain yang memiliki kompleksitas komputasi lebih rendah dibandingkan metode klasterisasi hirarki.

## 5. DAFTAR PUSTAKA

Beil, F., Ester, m. & Xu, X. (2002), "Frequent Term-Based Text Clustering", *Proceeding of The 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, hal.436-42.

Borgelt, C. (2005), “An Implementation of the FP-growth Algorithm”, Workshop Open Source data Mining Software, OSDM'05, Chicago, IL, 1-5.ACM Press, USA.

Dalli, A. (2003), "Adaptation of The *F-measure* to Cluster-Based Lexicon Quality ", *Proceedings of the EACL 2003 Workshop on Evaluation Initiatives in Natural Language Processing: are evaluation methods metrics and resources reusable*, hal.51-56.

Fung, B., Wang, K. & Ester, M. (2003), "Hierarchical Document Clustering using Frequent Itemsets", *Proceeding of The 3rd SIAM International*

Han, J. & Kamber, M., 2001. *Data Mining: Concepts and Techniques*. San Fransisco: Morgan Kaufmann.

Han, J., J. Pei, & Y. Yin. (2000), "Mining FrequentPatterns without Candidate Generation", ACM SIGMOD Int'l Conference on Management of Data.

Jain, A.K., Murty, M.N. & Flynn, P.J. (1999), "Data Clustering : A Review", *ACM Computing Survey Vol. 31, No. 3*, hal.264-323.

Khrisna, M. & Bhavani, D. (2010), "An Efficient Approach for Text Clusteiring Based on Frequent Itemsets", *European Journal of Scientif Research Vol. 42*, hal.399-410.

Li, Y.J., Chung, S.M. & Holt, J.D. (2008), "Text Document Clustering based on Frequent Word Meaning Sequences", *Data & Knowledge Engineering*, hal.381-404.

Niwattanakul, S., Singthongchai, J., Naenudron, E. & Wanapu, S., 2013. Using Jaccard Coefficient for Keywords Similarity. In *Proceedings of the International MultiConference of Engineers and Computer Scientists 2013 Vol I*. Hong Kong, 2013.

Steinbach, M., Krypis, G. & Kumar, V. (2000), "A Comparison of Document Clustering Techniques", *Proceeding Text Mining Workshop, KDD 2000*.

Tan, P.N., Stainbach, M. & Kumar, V., 2006. *Introduction to Data Mining*. 4th ed. New York: Pearson Addison Wesley.

Usha, R.M. (2011), "Review on Frequent Item-Based Dynamic Text Clustering", *International Journal of Computer Science and Information Technology & Security, Vol. 1, No. 2*, hal.98-103.

Wang, K., Xu, C. & Liu, B. (1999), "Clustering Transactions Using Large Items", *Proceeding of The 8th International Conference on Information and Knowledge Management*, hal.483-90.

Zhang, W., Yoshida, T., Tang, X. & Wang, Q. (2010), "Text Clustering using Frequent Itemsets", *Knowledge-Based System 23*, hal.379-88.

Zhao, Y. & Karypis, G. (2005), "Empirical and theoretical comparisons of selected criterion functions for document clustering", *Machine Learning 55 (3)*, hal.50-62.