







Obtaining WAPO-Structure Through Inverted Indexes

Úrsula Torres-Parejo¹ , Jesús R. Campaña² , Maria-Amparo Vila² ,
and Miguel Delgado² 

¹ Department of Statistics and Operational Research, University of Cádiz,
Cádiz, Spain

`ursula.torres@uca.es`

² Department of Computer Science and Artificial Intelligence, University of Granada,
Granada, Spain

Abstract. In order to represent texts preserving their semantics, in earlier work we proposed the WAPO-Structure, which is an intermediate form of representation that allows related terms to remain together. This intermediate form can be visualized through a tag cloud, which in turn serves as a textual navigation and retrieval tool. WAPO-Structures were obtained through a modification of the APriori algorithm, which spends a lot of processing time computing frequent sequences, for which it must perform numerous readings on the text until finding the frequent sequences of maximal level.

In this paper we present an alternative method for the generation of the WAPO-Structure from the inverted indexes of the text. This method saves processing time in texts for which an inverted index is already computed.

Keywords: Inverted indexes · Implications · Primary rules
Content representation · Text processing · Semantics
Frequent sequences · Text retrieval

1 Introduction

One of the main problems of information management in textual databases is the amount of unstructured text that is difficult to recover, due to the way of processing it. Many retrieval systems perform only syntactic text processing, which means that much of the content identification capability is lost in the retrieved text.

Frequent ordered itemsets preserve the semantics of the text since they allow related terms to remain ordered and united. This is achieved through the APO (Ordered APriori)-Structure [9], which represents the text through an intermediate form facilitating its processing, representing the content of the information and allowing greater precision and recall with the query results.

The WAPO-Structure introduces weights into the APO-Structure, so that the ordered sets can be visualized through a tag cloud with different font sizes.

In this way, the tag cloud works as an assistant for the query formulation and as a tool for exploring the contents of the database.

There are many methods to obtain the WAPO-Structure, such as all those for obtaining frequent itemsets [1,2,11] with slight modifications. But if the inverted indexes [4] are available, the processing time is considerably reduced, since it does not have to perform repeated readings on the database to calculate support values.

Hipp et al. [6] compare several of the algorithms in terms of performance for obtaining frequent itemsets, verifying that all have a similar behavior with respect to the execution time, although spending different time depending on tasks. While some of the algorithms compared use most of the time to determine the support of the candidate itemsets below level four, the Apriori algorithm finds the greatest difficulty in calculating support for itemsets of level four or higher.

In previous work [8,9] we obtained the WAPO-Structure from a modification of the Apriori algorithm [1]. In this paper, we propose its generation from a complete inverted index which is more advantageous as the level of the itemsets increases. With the Apriori algorithm, each time we add an item to an itemset, a new reading of the database is required, while with the inverted indexes, the maximum level itemsets are located in a single reading.

To the better understanding of the terminology used in this paper, we establish some previous concept definitions in Table 1.

Table 1. Concept definitions

Expression	Definition
Term	A word or group of words
Mono-term	Single word
Item	An individual article or unit, especially one that is part of a list, collection, or set
Sequence	Ordered list of items

This paper is organized as follows. Section 2 defines APO-Structure and WAPO-Structure. Section 3 gives the definition of Full Inverted Index. Section 4 explains the method for obtaining the WAPO-Structure from a complete inverted index. Section 5 illustrates this process with a practical example and, finally, we end with a brief discussion and conclusions in Sect. 6.

2 APO-Structure and WAPO-Structure

The lack of structure in textual attributes complicates their automatic processing. In [8,9] we see how the APO and WAPO Structures provide the mathematical structure to obtain the semantics inherent to the text, facilitating the

processing. These semantics are achieved by allowing the frequently related terms to remain together.

The complete process is carried out in five steps: Selecting the textual attribute, syntactic preprocessing, semantic preprocessing, structure generation and displaying of the structure.

The WAPO-Structure provides weight into the APO-Structure.

2.1 APO-Structure [9]

Definition 1. AP-Seq (AP-Sequences)

Let $X = \{x_1, x_2, \dots, x_n\}$ be a referential set of items and R a sequence of frequent itemsets. R is then an AP-Seq if and only if:

1.
$$\forall Z = (z_1, z_2, \dots, z_k) \in R \Rightarrow \begin{cases} (z_1, z_2, \dots, z_{k-1}) \in R \\ (z_2, z_3, \dots, z_k) \in R \end{cases} \quad \forall k \in [2, n] \quad (1)$$

2. $\exists Y \in R$ such that:

$$card(Y) = \max_{Z \in R} (card(Z)) \text{ and } \nexists Y' \in R \mid card(Y') = card(Y) \quad (2)$$

$$\forall Z \in R \implies Z \subseteq Y \quad (3)$$

The sequence Y with higher order is the spanning sequence of R , $R = g(Y)$, in other works $g(Y)$ is the AP-Seq with spanning sequence Y , being the cardinal of Y the level of $g(Y)$.

Example 1. Let $X = \{\text{intelligence, online, measure, test, partner}\}$.

Let $R = \{<\text{intelligence, test, partner}>, <\text{intelligence, test}>, <\text{test, partner}>, <\text{intelligence}>, <\text{test}>, <\text{partner}>\}$.

Then, R is an AP-Seq with spanning sequence $Y = <\text{intelligence, test, partner}>$. All the other sequences are included in it with the same order position between the terms.

Definition 2. APO-Structure

Let $X = \{x_1, \dots, x_n\}$ be a referential set of items and $S = \{A, B, \dots\}$ a set of frequent item-seqs with a cardinal higher than or equal to one and A, B, \dots AP-Seqs such as:

$$\forall A, B \in S; A \not\subseteq B, B \not\subseteq A \text{ and } B \neq A$$

An APO-Structure generated by S , $E = g(A, B, \dots)$, is the set of AP-Seqs whose spanning sequences are A, B, \dots

2.2 WAPO-Structure [9]

Definition 3. Frequent weighted item-seq of an AP-Seq

Let $R = g(Y)$ be an AP-Seq with a referential set of items X . It is said that $\tilde{\alpha}_t \subseteq Y$ is a frequent weighted item-seq from Y if:

$$\tilde{\alpha}_t = [\alpha_t, \omega_t]. \tag{4}$$

where α_t is a frequent term sequence and ω_t is its weight or frequency.

WAPO-Structures are structures composed of weighted AP-Seqs which are AP-Seq composed of weighted item-seqs.

Definition 4. WAPO-Structure

Let $X = \{x_1, x_2, \dots, x_n\}$ be a referential set of items and $\tilde{S} = \{\tilde{A}, \tilde{B}, \dots\}$ a set of frequent weighted item-seqs with a cardinal higher than or equal to one, and $\tilde{A}, \tilde{B}, \dots$ weighted AP-Seqs such as:

$$\forall A, B \in S; A \not\subseteq B, B \not\subseteq A \text{ and } B \neq A. \tag{5}$$

A WAPO-Structure generated by $\tilde{S}, \tilde{E} = g(\tilde{A}, \tilde{B}, \dots)$ is the set of AP-Seqs whose spanning sequences are $\tilde{A}, \tilde{B}, \dots$

Note 1. We express the spanning sequence \tilde{A} as well as $\tilde{g}(A)$.

Example 2. Let us suppose a database containing tuples in Table 2.

Table 2. Tuples in the Example 2

n	Item-seqs
1	<intelligence, test, online, measure>
2	<measure, intelligence, test>
3	<measure, online>

Setting a support of 2 in terms of absolute frequency to consider an item-seq to be frequent, we obtain the following structures:

$$\begin{aligned}
 \text{APO - Structure} &: g(\langle \text{intelligence, test} \rangle, \langle \text{online} \rangle, \langle \text{measure} \rangle) \\
 &= (\langle \text{intelligence, test} \rangle, \langle \text{intelligence} \rangle, \langle \text{test} \rangle, \\
 &\quad \langle \text{online} \rangle, \langle \text{measure} \rangle) \\
 \text{WAPO - Structure} &: \tilde{g}(\langle \text{intelligence, test} \rangle, \langle \text{online} \rangle, \langle \text{measure} \rangle) \\
 &= (\langle \text{intelligence, test} \rangle, 2), (\langle \text{intelligence} \rangle, 2), \\
 &\quad (\langle \text{test} \rangle, 2), (\langle \text{online} \rangle, 2), (\langle \text{measure} \rangle, 3)
 \end{aligned}$$

3 Complete Inverted Index

The inverted indexes are widely used in information retrieval [3,7], as well as for other applications [5,10]. In this work we use the definitions given in [4], to understand them, some notations are established in Table 3.

Table 3. Notations

Symbol	Definition
Σ	Finite non-empty alphabet
Σ^*	Set of all items on Σ
λ	Empty word
Σ^+	$\Sigma^* - \{\lambda\}$
S	Finite set of text words $S \subseteq \Sigma^+$
$Sub(S)$	Set of sub-strings in S

Note 2. If $\omega = xyz$ for the terms $x, y, z \in \Sigma^* \Rightarrow y$ is a subterm of ω , x is a prefix of ω and z is a suffix of ω .

Definition 5. Complete Inverted Index [4]

Given a finite alphabet Σ , a set of terms $k \subseteq \Sigma^+$ and a set of texts $S \subseteq \Sigma^+$, a complete inverted index for (Σ, k, S) is an abstract data type that implements the following functions:

1. **find:** $\Sigma^+ \rightarrow k \cup \{\lambda\}$, where $find(\omega)$ is the largest prefix x of ω with $x \in k \cup \{\lambda\}$ and x occurs in S , that is, x is a subset of terms of a text in S .
2. **freq:** $k \rightarrow \mathbb{N}$, where $freq(\omega)$ is the number of times that ω occurs is a subset of terms of a text in S .
3. **locations:** $k \rightarrow 2^{N \times N}$, where $locations(\omega)$ is the number of ordered pairs indicating the number of the text in which ω occurs and its position within the text.

Definition 6. Rule of S [4]

A rule of S (r_S) is a production $x \rightarrow_s \gamma x \beta$ where $x \in sub(S), \gamma, \beta \in \Sigma^*$ that occurs each time that x is preceded by γ and followed by β in S .

Definition 7. Primary rule of S [4]

$t_S : x \rightarrow_s \gamma x \beta$ is a primary rule of S if it is a rule S and γ and β are sets of terms of the highest possible order, that is, $\nexists \delta, \tau \in \Sigma^*$ with $\delta, \tau \neq \lambda$ such as $x \rightarrow_s \delta \gamma x \beta \tau$ be a rule of S .

Definition 8. Implication of x in S [4]

If $x \rightarrow_s \gamma x \beta$ is a primary rule of S , then $\gamma x \beta$ is called implication of x in S and is denoted $imp_S(x) : P(S) = \{imp_S(x) : x \in sub(S)\}$

The members of $P(S)$ are called subsets of primary terms of S .

4 Obtaining the WAPO-Structure Through Complete Inverted Indexes

It is possible to obtain the WAPO-Structure from the APO-Structure through inverted indexes mainly in two ways:

The first is through the Apriori algorithm, in a similar way as the AprioriTid and AprioriHybrid algorithms work [1], with a slight modification to introduce order and weight into the itemsets.

These algorithms construct iteratively the set of frequent terms, using the frequent itemsets found in a step to build the candidate itemsets and check if they are frequent in the next step.

In the first step, the support of elementary items or items of level 1 is calculated and determines which of these items are considered frequent according to the minimum support. In each subsequent step, it starts with a “seed” set consisting of itemsets found in the previous step combined with each other to generate the candidate itemsets deciding which of these are, in turn, frequent.

To do this, the Apriori algorithm requires at each step to re-read data, but the AprioriTid has the property that it is not necessary to go through the entire database to calculate the support of the candidate itemsets after the first step. For this purpose, a codification of the candidate itemsets found in the previous step is created, before deciding whether they are frequent in the subsequent step. In successive steps, the size of this coding is becoming much smaller than that of the database, saving a lot of reading effort.

This coding is the one that we can perform through the inverted index, to later apply the Apriori algorithm, just instead of going through the entire database at each step, only the inverted indexes are read, which indicate the ordered positions of each term in the text, discarding those that do not correspond with a frequent itemset for the later step.

The second way is the one proposed in this article and consists of identifying the implications of x_i^1 with the spanning sets of the APO-Structure, being x_i^1 each of the frequent itemsets of level equal to 1. Obviously, we would have to identify those implications that, in turn, were frequent.

To do this, the primary rules of x_i^1 (tuples containing the term in question) are previously obtained and the maximals are selected, storing the frequent ones. Those not frequent are divided into sub-rules, deciding which of these are, in turn, frequent. Once the set of all the frequent rules is obtained, we eliminate the redundant and not maximal ones and the remaining rules are what we will call “frequent implications of x_i^1 ”, identifying them with the spanning sets of the APO-Structure.

Finally, we use the function **freq** to obtain the weight of the item-seqs in the APO-Structure and generate the WAPO-Structure.

Next, we define the set of frequent implications of x in S , where all the frequent implications of x_i^1 are stored for identifying them with the spanning sets of the APO-Structure.

Definition 9. Set of frequent implications of x in S

Let $P(S) = \{imp_S(x) : x \in sub(S)\}$ be the set of all the implications in S , we call $Pf(S)$ the set of all the frequent implications in S , then $Pf_{x_i^1}(S) = \{imp_S^j(x_i^1) : x \in sub(S)\}$ with $x_i^1 =$ frequent itemset of level 1 and j each of the frequent implications of the itemset i .

Definition 10. Correspondence between frequent implications and the spanning sets of the APO-Structure

Let $E = g(A, B, \dots, K, \dots)$ be the APO-Structure generated by the sets A, B, \dots, K, \dots , then:

$A = imp_S^1(x_1^1), B = imp_S^2(x_1^1), \dots, K = imp_S^1(x_2^1), \dots$ removing redundancies.

The following algorithm specifies the process in more detail. For its application it is necessary to have the complete inverted index for all the terms of the base of texts S .

Algorithm

1. Identify frequent mono-terms according to support:
 $\omega_i = x_i^1 \ i \in S$
 If $freq(\omega_i) > support \Rightarrow \omega_i$ is frequent (denoted as ω_i^f)
2. Calculate the primary rules of ω_i^f :
 If $\omega_i^f \in t_j$ and $t_j =$ maximal tuple in S
 $\Rightarrow t_j$ primary rule of ω_i (denoted as r_k)
3. Remove redundancies
4. Check if the rules obtained are frequent:
 If $freq(r_k) > support \Rightarrow r_k$ frequent rule (denoted as r_k^f)
5. Store the frequent rules in the set of frequent implications:
 Input r_k^f en $Pf(S)$
6. Split non-frequent rules into sub-rules:
 If $freq(r_k) < support \Rightarrow r_k$ no frequent rule (denoted as $\overline{r_k^f}$)
 $\forall (\overline{r_k^f}) = \{i_1, \dots, i_n\}$ non-frequent rule of $S \Rightarrow$
 $\{i_i, \dots, i_{n-1}\} \ y \ \{i_2, \dots, i_n\}$ rule of S .
7. Go back to step 4.
8. Remove redundancies in $Pf(S)$
9. Identify each of the frequent implications in $Pf(S)$ with the spanning sets of the APO-Structure.

5 Example of How to Obtain APO-Structure and WAPO-Structure Through Implications

Let us suppose we obtain the item-seqs listed in Table 4 from a database after cleaning the text.

The function **locations**(ω) for all mono-terms is presented in Table 5 and the image of the function **freq**(ω) in Table 6.

Table 4. Item-seqs after text cleaning

n_i	Item-seqs(i)
1	$\langle pink, yellow, blue \rangle$
2	$\langle green, pink, yellow \rangle$
3	$\langle green, blue \rangle$
4	$\langle orange, pink, yellow, blue \rangle$
5	$\langle green, pink \rangle$
6	$\langle yellow, green \rangle$
7	$\langle green, pink, yellow \rangle$

Table 5. locations(ω)

Term	n_1	n_2	n_3	n_4	n_5	n_6	n_7
pink	(1,1)	(2,2)		(4,2)	(5,2)		(7,2)
yellow	(1,2)	(2,3)		(4,3)		(6,1)	(7,3)
blue	(1,3)		(3,2)	(4,4)			
green		(2,1)	(3,1)		(5,1)	(6,2)	(7,1)
orange				(4,1)			

Table 6. freq(ω)

ω_i	Locations(ω_i)	F_{ω_i}
pink	{(1,1), (2,2), (4,2), (5,2), (7,2)}	5
yellow	{(1,2), (2,3), (4,3), (6,1), (7,3)}	5
blue	{(1,3), (3,2), (4,4)}	3
green	{(2,1), (3,1), (5,1), (6,2), (7,1)}	5
orange	{(4,1)}	1

Let us consider a minimum support for an item-seq to be frequent greater or equal than an absolute frequency of 2. In the current example the frequent item-seqs of cardinality 1 are the mono-terms $\langle pink \rangle$, $\langle yellow \rangle$, $\langle blue \rangle$ and $\langle green \rangle$.

We compute the implications for each of these frequent item-seqs of cardinality 1. To do it first, it is necessary to compute their rules. The rules for item-seq $\langle pink \rangle$ along with each rule frequency are shown in Table 7, where r_i represents the rule i and $freq_i$ represents its frequency.

Then, the primary rules are identified. We select only maximal rules. Rule r_4 it is not a primary rule, as it is contained in rule r_2 . Rule r_1 it is neither a primary rule as it is contained in rule r_3 .

Table 7. Rules for item-seq $\langle pink \rangle$

i	$r_i(\langle pink \rangle)$	$freq_i(r_i(\langle pink \rangle))$
1	$\langle pink, yellow, blue \rangle$	2
2	$\langle green, pink, yellow \rangle$	2
3	$\langle orange, pink, yellow, blue \rangle$	1
4	$\langle green, pink \rangle$	3

Table 8. Primary rules for item-seq $\langle pink \rangle$

i	$t_i(\langle pink \rangle)$	$freq_i(t_i(\langle pink \rangle))$
1	$\langle green, pink, yellow \rangle$	2
2	$\langle orange, pink, yellow, blue \rangle$	1

Table 9. Subrules for (t_2)

i	$r'_i(\langle t_2 \rangle)$	$freq_i(r'_i(\langle t_2 \rangle))$
1	$\langle orange, pink, yellow \rangle$	1
2	$\langle pink, yellow, blue \rangle$	2

Table 10. Subrules for (r'_1)

i	$r''_i(\langle r'_1 \rangle)$	$freq_i(r''_i(\langle r'_1 \rangle))$
1	$\langle orange, pink \rangle$	1
2	$\langle pink, yellow \rangle$	4

Table 8 shows the primary rules for item-seq $\langle pink \rangle$, where t_i represents the primary rule i and $freq_i$ its frequency. Of these primary rules, only t_1 is frequent regarding the support, so we store it in the set of frequent implications Pf(S).

Since t_2 is not frequent, it is divided into two sub-rules that we can see in Table 9, where r'_i represents the sub-rule i and $freq_i$ its frequency. In this case, r'_2 is frequent and comes from a non-frequent primary rule, so r'_2 becomes a frequent primary rule (since there is no frequent higher-order rule) and it is stored in the set of frequent implications Pf(S).

Since r'_1 is not frequent, it is divided into two sub-rules r''_i (see Table 10). The rule r''_2 is frequent and it is stored in Pf(S), however it is not a primary rule since there is another maximal rule in Pf(S) containing it, so it will be removed from this set. The r''_1 rule is not frequent, so the operations of dividing into sub-rules, checking the frequencies and storing the frequent rules would be repeated.

Finally, two frequent implications for $\langle pink \rangle$ in Pf(S) have been obtained. They are shown in Table 11.

The same procedure is applied for the item-seqs $\langle yellow \rangle$, $\langle blue \rangle$ and $\langle green \rangle$, obtaining their frequent implications. They are shown in Tables 12, 13 and 14.

Table 11. Frequent implications for $\langle pink \rangle$

i	$imp_i(\langle pink \rangle)$	$freq_i(imp_i(\langle pink \rangle))$
1	$\langle green, pink, yellow \rangle$	2
2	$\langle pink, yellow, blue \rangle$	2

Table 12. Frequent implications for $\langle yellow \rangle$

i	$imp_i(\langle yellow \rangle)$	$freq_i(imp_i(\langle yellow \rangle))$
1	$\langle pink, yellow, blue \rangle$	2
2	$\langle green, pink, yellow \rangle$	2

Table 13. Frequent implications for $\langle blue \rangle$

i	$imp_i(\langle blue \rangle)$	$freq_i(imp_i(\langle blue \rangle))$
1	$\langle pink, yellow, blue \rangle$	2

In total we have determined the next implications:

- $imp_S(x_1^1) = imp(\langle pink \rangle) \Rightarrow imp^1(x_1^1) = \langle pink, yellow, blue \rangle$ and $imp^2(x_1^1) = \langle green, pink, yellow \rangle$
- $imp_S(x_2^1) = imp(\langle yellow \rangle) \Rightarrow imp^1(x_2^1) = \langle pink, yellow, blue \rangle$ and $imp^2(x_2^1) = \langle green, pink, yellow \rangle$
- $imp_S(x_3^1) = imp(\langle blue \rangle) \Rightarrow imp^1(x_3^1) = \langle pink, yellow, blue \rangle$
- $imp_S(x_4^1) = imp(\langle green \rangle) \Rightarrow imp^1(x_4^1) = \langle green, pink, yellow \rangle$

When duplicate implications are removed, two implications remain:

$$imp^1(x_1^1) = \langle pink, yellow, blue \rangle \text{ and } imp^2(x_1^1) = \langle green, pink, yellow \rangle$$

These implications will be used as the maximal itemseqs in the APO-Structure, which has cardinal 2.

Let E be an APO-Structure, with spanning sequences A and B :

$$E = g(A, B) \Rightarrow A = imp^1(x_1^1) \text{ and } B = imp^2(x_1^1)$$

$$E = g(\langle pink, yellow, blue \rangle, \langle green, pink, yellow \rangle)$$

$$E = (\langle pink, yellow, blue \rangle, \langle green, pink, yellow \rangle, \langle pink, yellow \rangle, \langle yellow, blue \rangle, \langle green, pink \rangle, \langle pink \rangle, \langle yellow \rangle, \langle blue \rangle \text{ and } \langle green \rangle)$$

In order to compute the weight for the WAPO-Structure, we use the function $freq(\omega_i)$ with $i = \text{item-seq} \in \text{APO-Structure}$. The resulting WAPO-Structure is the following:

Table 14. Frequent implications for $\langle green \rangle$

i	$imp_i(\langle green \rangle)$	$freq_i(imp_i(\langle green \rangle))$
1	$\langle green, pink, yellow \rangle$	2

$$\begin{aligned} \tilde{E} &= g((\langle pink, yellow, blue \rangle, 2), (\langle green, pink, yellow \rangle, 2)) \\ \tilde{E} &= ((\langle pink, yellow, blue \rangle, 2), (\langle green, pink, yellow \rangle, 2), \\ &(\langle pink, yellow \rangle, 4), (\langle yellow, blue \rangle, 2), (\langle green, pink \rangle, 3) \\ &(\langle pink \rangle, 5), (\langle yellow \rangle, 5), (\langle blue \rangle, 3) \text{ and } (\langle green \rangle, 5)) \end{aligned}$$

6 Conclusions

The inverted index helps us to identify the primary rules of the frequent terms and the frequent sub-rules of the non-frequent primary rules. The set consisting of all frequent maximal rules is called “the set of frequent implications”. Each of the rules in this set corresponds to a spanning set of the APO-Structure, so we have the WAPO-Structure from these frequent implications and their frequencies.

The biggest drawback of this method is that when there are many maximal non-frequent item-seqs, a lot of time is lost in the decomposition of these sequences until finding the level in which they are frequent, in order to find the frequent implications. This drawback makes the method more appropriate in a text in which many large repeated sequences are found. In other case, it is preferable to use the Apriori algorithm, which according to Hipp et al. [6] finds the greatest difficulty in calculating support for itemsets of level four or higher and is best known for its ease and simplicity of implementation.

In short, the method proposed in this paper saves reading time by not having to go through the database repeatedly to get the frequent item-seqs as the Apriori algorithm method [1] does. Its application is recommended in databases where most of the frequent sequences are long, but the Apriori algorithm works better in other cases, depending on the characteristics of the text.

Acknowledgements. This work has been partially supported by the “Plan Andaluz de Investigación, Junta de Andalucía” (Spain) under research project P10-TIC6019.

References

1. Agrawal, R., Srikant, R.: Fast algorithms for mining association rules. In: Proceedings of the 20th International Conference in Very Large Data Bases, VLDB, vol. 1215, pp. 487–499. Citeseer (1994)
2. Agrawal, R., Srikant, R.: Mining sequential patterns. In: Proceedings of the Eleventh International Conference on Data Engineering, pp. 3–14. IEEE (1995). <https://doi.org/10.1109/ICDE.1995.380415>

3. Baeza-Yates, R., Ribeiro-Neto, B., et al.: *Modern Information Retrieval*. ACM Press, New York (1999)
4. Blumer, A., Blumer, J., Haussler, D., McConnell, R., Ehrenfeucht, A.: Complete inverted files for efficient text retrieval and analysis. *J. ACM* **34**(3), 578–595 (1987). <https://doi.org/10.1145/28869.28873>
5. Cutting, D., Karger, D., Pedersen, J., Tukey, J.: Scatter/Gather: a cluster-based approach to browsing large document collections. In: *ACM SIGIR Forum*, vol. 51, pp. 148–159. ACM (2017). <https://doi.org/10.1145/3130348.3130362>
6. Hipp, J., Gütntzer, U., Nakhaeizadeh, G.: Algorithms for association rule mining - a general survey and comparison. *SIGKDD Explor. Newsl.* **2**, 58–64 (2000). <https://doi.org/10.1145/360402.360421>
7. Patil, M., Thankachan, S., Shah, R., Hon, W., Vitter, J., Chandrasekaran, S.: Inverted indexes for phrases and strings. In: *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 555–564. ACM (2011). <https://doi.org/10.1145/2009916.2009992>
8. Torres-Parejo, U., Campaña, J.R., Vila, M.A., Delgado, M.: MTCIR: a multi-term tag cloud information retrieval system. *Expert Syst. Appl.* **40**(14), 5448–5455 (2013). <https://doi.org/10.1016/j.eswa.2013.04.010>
9. Torres-Parejo, U., Campaña, J., Vila, M., Delgado, M.: A theoretical model for the automatic generation of tag clouds. *Knowl. Inf. Syst.* **40**(2), 315–347 (2014). <https://doi.org/10.1007/s10115-013-0651-9>
10. Vdorhees, E.: The cluster hypothesis revisited. In: *ACM SIGIR Forum*, vol. 51, pp. 35–43. ACM (2017). <https://doi.org/10.1145/3130348.3130353>
11. Zaki, M.: SPADE: an efficient algorithm for mining frequent sequences. *Mach. Learn.* **42**(1), 31–60 (2001). <https://doi.org/10.1109/ICDE.2004.1320012>