5 Affinity and Co-Citation Analysis Approaches

In this chapter, we present several Web community analysis approaches, such as affinity, co-citation etc, for capturing underlying relationships among Web pages. In Sect. 5.1, we start presenting a new Web page similarity measurement, which incorporates hyperlink transitivity and page importance. Then, Sect. 5.2 gives a hierarchical clustering algorithm based on page correlation matrix. In Sect. 5.3, we adapt a concept of affinity and utilize it to represent the relationship between two pages. The permutation of affinity matrix is utilized to achieve highest global affinity. Moreover, the affinity-based clustering algorithms are given in this section. The Co-Citation algorithm is discussed in detail in Sect. 5.4 and an extended algorithm is described as well.

5.1 Web Page Similarity Measurement

A Web page similarity usually refers to a certain page space. Since we are concerned about clustering Web-searched results in this work, we focus on a page space that is related to the user's query topics. The ideas and analysis techniques in the following sections, however, could also be used to other concerned Web spaces, such as those in (Weiss et al. 1996) and (Pitkow and Pirolli 1997). In this section, we firstly establish a page source (space) that is related to the query topics. Within this page source, we incorporate hyperlink transitivity and page importance to propose a new page similarity measurement.

5.1.1 Page Source Construction

The page source construction is based on the Web-searched results. For users, they are usually concerned about a part of searched results; say the first r highest-ranked pages returned by the search engine. From the hyperlink analysis point of view, the pages that link to or are linked to these r highest-ranked pages are also related to the query topics to some extent. Therefore, the page source S with respect to user's query topics is constructed as follow:

Step 1: Select r highest-ranked pages from the searched results to form a root page set *R*.
Step 2: For each page *p* in *R*, select up to *B* pages, which point to *p* and whose domain names are different from that of *p*, and add them to the back vicinity set *BV* of *R*.
Step 3: For each page *p* in *R*, select up to *F* pages, which are pointed to by *p* and whose domain names are different from that of *p*, and add them to the forward vicinity set *FV* of *R*.
Step 4: Page source S is constructed by uniting sets *R*, *BV*, *FV* and adding original links between pages in *S*.

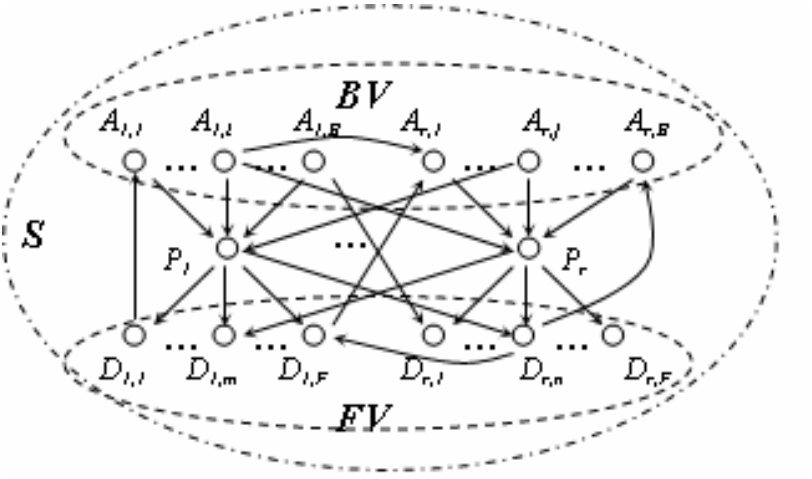


Fig. 5.1. Structure of the page source *S*

In the above page source construction algorithm, parameters *B* and *F* are used to guarantee that the page source *S* is of a reasonable size. For example, we choose the value 200 for *B* and *F* from our experiment experience. When constructing sets *BV* and *FV*, it is required that for each page *p* in *R*, the domain names of its parent pages and child pages are different from the domain name of the page *p*. This requirement filters those parent and child pages coming from the same Website where the page *p* is located. The reason, as indicated in (Bharat and Henzinger 1998; Wang and Kitsuregawa 2001), is that the links within the same Website are more likely to reveal the inner structure than to imply a certain semantic relationship.

During the page source construction procedure, it is possible to bring some mirror pages into the page source. There are several reasons for not being required to remove these mirror pages. Firstly, there is no standard currently to identify whether two pages are mirror pages or not just from their linkage

analysis, and identifying mirror pages will add extra computing cost. Secondly, if two pages are mirror pages, they have the same hyperlink structure and are most likely to be clustered into one cluster, in which the user or an algorithm can identify them easily. Therefore, keeping a proper mirror page redundancy in the page source S is reasonable.

It is worth indicating that the Web pages and their linkage information required for page source construction could be obtained in many ways. For example, the child pages of a certain page and their links can be directly obtained from that page, while the parent pages of that page and their links can be found by the functions provided by some Web browsers, such as the search function *link:URL* provided by *AltaVista* and *Google*. Bharat et al (Bharat et al. 1998) proposed a specific system to obtain linkage information from the Web. Usually, Web search engines use crawlers (spiders) to obtain the Web pages with hyperlink information, and the obtained information is stored in a specific database for further use (Brin and L. Page 1998).

5.1.2 Page Weight Definition

The role each page plays in similarity measurement is different in a concerned page source S. For instance, two kinds of pages need to be noticed. The first one is the page whose *out-link contribution* to S (i.e. the number of pages in S that are pointed to by this page) is greater than the average out-link contribution of all the pages in the page source S. Another kind is the page whose *in-link contribution* to S (i.e. the number of pages in S that point to this page) is greater than the average in-link contribution of all the pages in the page source S. The pages of the first kind are called *index* pages in (Botafogo and Shneiderman 1991) (*hub* pages in (Kleinberg 1999)), and those of the second kind are called *reference* pages in (Botafogo and Shneiderman 1991) (*authority* pages in(Kleinberg 1999)). These pages are most likely to reflect certain topics related to the query within the concerned page source. If two pages are linked by or linking to some pages of these kinds, these two pages are more likely to be located in the same topic group and have higher similarity.

It also needs to be noticed that index Web pages in common sense, such as personal bookmark pages and index pages on some special-purpose Web sites, might not be the index pages in the concerned page source S if their out-link contribution to S is below the average out-link contribution in S. For the same reason, some pages with high in-degrees on the Web, such as home pages of commonly used search engines, might not be the reference pages in the concerned page source S. For simplicity, we filter the home pages of commonly used search engines (e.g. *Yahoo!*, *AltaVista*, *Google* and *Excite*) from the concerned page source S, since these pages are not related to any

specific topics. To measure the importance of each page *within the concerned page source*, we define a weight for each page.

For each page P_i in the page source S, similar to the HITS algorithm in (Kleinberg 1999), we associate a non-negative *in-weight* $P_{i,in}$ and a non-negative *out-weight* $P_{i,out}$ with it. Due to the hyperlink transitivity in the page source, the *in-weight* and *out-weight* for the page P_i in S are iteratively calculated as follows (Kleinberg 1999):

$$P_{i,in} = \sum_{P_j \in S, P_j \to P_i} P_{j,out} ,$$

$$P_{i,out} = \sum_{P_j \in S, P_i \to P_j} P_{j,in} .$$

In order to guarantee the convergence of the above iterative operations, it is required that the in-weight vector and out-weight vector are normalized after each iteration, i.e.

$$\sum_{P_i \in S} P_{i,in}^2 = 1, \ \sum_{P_i \in S} P_{i,out}^2 = 1 .$$

We denote the average in-weight of S as μ, and the average out-degree of S as λ. That is

$$\mu = \sum_{P_i \in S} P_{i,in} / size(S) , \ \lambda = \sum_{P_i \in S} P_{i,out} / size(S) ,$$

where $size(S)$ is the number of pages in S. Then the page weight for P_i is defined as

$$w_i = 1 + \max((P_{i,in} - \mu)/(M_{in} - m_{in}), (P_{i,out} - \lambda)/(M_{out} - m_{out})) \quad (5.1)$$

where M_{in}, m_{in}, M_{out} and m_{out} are defined as follow:

$$M_{in} = \max_{P_j \in S}(P_{j,in}) , \quad m_{in} = \min_{P_j \in S}(P_{j,in}) ,$$

$$M_{out} = \max_{P_j \in S}(P_{j,out}) , \ m_{out} = \min_{P_j \in S}(P_{j,out}) .$$

The page weight definition in (1) indicates that if a page's in-weight and out-weight in S are below their corresponding average values μ and λ, its weight will be less than 1, which means its influence to the similarity measurement is relatively less. For the same reason, if a page's in-weight or out-weight in S is above the average value (e.g. an index page or a reference page), its weight will be greater than 1 and its influence to the similarity measurement is relatively greater. In other words, the page weight defined in (1) reflects the importance of each page's role in the concerned page source. This page importance will be incorporated in the page similarity measurement.

5.1.3 Page Correlation Matrix

For each Web page, its correlation with other pages, via linkages, is expressed in two ways: one is *out-links* from it, another is *in-links* to it. In this work, the similarity between two pages is measured by their own correlations with other pages in the page source S, rather than being derived directly from the links between them. For measuring the page correlation, we firstly give the following definitions.

Definition 1. If page A has a direct link to page B, then the length of path from page A to page B is 1, denoted as $l(A,B) = 1$. If page A has a link to page B via n other pages, then $l(A,B) = n+1$. The distance from page A to page B, denoted as $sl(A,B)$, is the shortest path length from A to B, i.e. $sl(A,B) = \min(l(A,B))$. The length of path from a page to itself is zero, i.e. $l(A,A) = 0$. If there are no links from page A to page B (direct or indirect), then $l(A,B) = \infty$.

It can be inferred from this definition that $l(A,B) = \infty$ does not imply $l(B,A) = \infty$, because there might still exist links from page B to page A in this case.

Definition 2. The correlation weight between two pages i and j ($i \neq j$), denoted as $w_{i,j}$, is the maximal weight of their weights, i.e. $w_{i,j} = \max(w_i,w_j)$ where w_i and w_j are the page weights for pages i and j respectively. If $i = j$, $w_{i,j}$ is defined as 1.

The following definition defines how much two pages correlate with each other if there is a direct link between them.

Definition 3. Correlation factor, denoted as F, $0<F<1$, is a constant that measures the correlation rate between two page with direct link, i.e. if page A has a direct link to page B, then the correlation rate from page A to page B is F.

How to determine the value of this correlation factor F to more precisely reflect the correlation relationship between pages is beyond the scope of this work. Further research could be done in this area. In this work, similar to the work in (Weiss et al. 1996), the value of F is chosen as 1/2. That means if page A has a direct link to page B, the correlation from page A to page B is 50%. It is argued that not every pair of pages that are hyperlinked has 50% semantic relationship with each other. However, in the context of the Web, the research focuses on finding certain statistical regularities from a large number of pages. Therefore, certain imprecise relationship descriptions are permitted as in (Weiss et al. 1996). For general purpose, we still use F in the following algorithm to represent this correlation factor. It can be seen that hyperlinks between pages are used to measure the correlations between pages, rather than to directly measure the similarities.

With the above definitions, a correlation degree between any two pages can be defined. This correlation degree depends on the value of correlation factor F, the distance between the two pages (the farther the distance, the less the correlation degree), and the correlation weights of involved pages along the shortest path. The following definition gives this function.

Definition 4. The *correlation degree* from page i to page j, denoted as c_{ij}, is defined as

$$c_{ij} = w_{i,k1} w_{k1,k2} \cdots w_{kn,j} F^{sl(i,j)}, \tag{5.2}$$

where F are correlation factor, $sl(i,j)$ is the distance from page i to page j, and $w_{i,k1}$, $w_{k1,k2}$, …, $w_{kn,j}$ are correlation weights of the pages i, $k1$, $k2$, …, kn, j that form the distance $sl(i,j)$, i.e. $i \to k1 \to k2 \to \ldots \to kn \to j$. If $i = j$, then c_{ij} is defined as 1.

For the concerned page source S, we suppose the size of the root set R is m, the size of the vicinity set $V = BV \cup FV$ is n. Then the correlation degrees of all the pages in S can be expressed in a $(m+n)\times(m+n)$ matrix $C = (c_{ij})_{(m+n)\times(m+n)}$, called *correlation matrix*. This correlation matrix C is a numerical format that converts the hyperlinks (direct or indirect) between pages in S into the correlation degrees, incorporating the hyperlink transitivity and page importance.

The key for computing the correlation degree c_{ij} in (2) is the distance $sl(i,j)$ between any two pages i and j in S. This distance can be computed via some operations on the matrix elements of a special matrix called *primary correlation matrix*. The primary correlation matrix $A = (a_{ij})_{(m+n)\times(m+n)}$ is constructed as follows:

$$a_{ij} = \begin{cases} F & \text{if there is a direct link from } i \text{ to } j, i \neq j \\ 1 & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}$$

Based on this primary correlation matrix, the algorithm for computing the distance $sl(i,j)$ between any two pages i and j is described as follows:

Step 1: For each page $i \in S$, choose $factor = F$ and go to Step 2;
Step 2: For each element a_{ij}, if $a_{ij} = factor$, then set $k = 1$ and go to Step 3. If there is no element a_{ij} ($j = 1, \ldots, m+n$) such that $a_{ij} = factor$, then go back to Step 1;
Step 3: If $a_{jk} \neq 0$ and $a_{jk} \neq 1$, calculate $factor*a_{jk}$;
Step 4: If $factor*a_{jk} > a_{ik}$, then replace a_{ik} with $factor*a_{jk}$, change $k = k+1$ and go back to Step 3. Otherwise, change $k = k+1$ and go back to Step 3;
Step 5: Change $factor = factor*F$ and go to Step 2 until there are no changes to all element values a_{ij} ;

Step 6: Go back to Step 1 until all the pages in S have been considered.

After element values of matrix A are updated by the above algorithm, the distance from page i to page j is

$$sl(i, j) = [\log a_{ij} / \log F] .$$

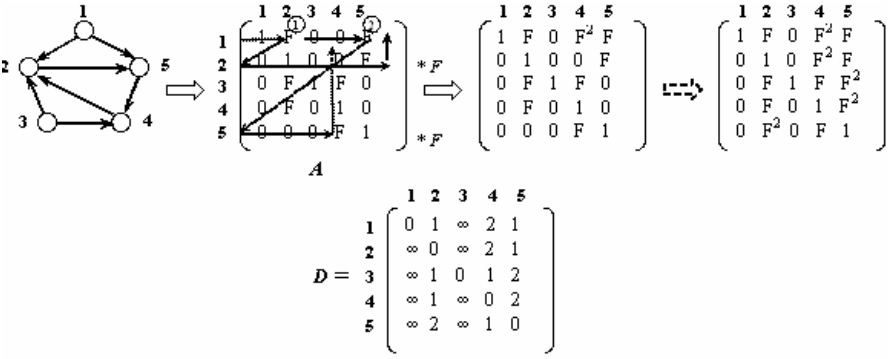


Fig. 5.2. Example of computing distance between pages

The example in Fig. 5.2 gives an intuitive execution demonstration of the above algorithm. In this example, five pages (numbered 1 to 5) and their linkages are represented as a directed graph. Their primary correlation matrix A is also shown in the figure. The dashed arrows in matrix A show the first level operation sequence ($factor = F$) of the above algorithm for page 1. The procedure of other level operations for other pages is similar except for changing the values of variable *factor* according to the above algorithm. The final updated primary correlation matrix and the corresponding distance matrix D are presented in the figure. It is clear from these results that although there are several paths from page 1 to page 4, the distance from page 1 to page 4 is 2, which is consistent with the real situation. The situation is the same for page 3 and page 5 in this example.

This distance computation algorithm could be adapted for computing the correlation degrees (5.2). The above algorithm also provides a numerical method to find the shortest path between any two nodes in a directed graph. If page correlation weights are not considered in computing the correlation degrees c_{ij}, the above algorithm could be directly used to produce correlation matrix C.

5.1.4 Page Similarity

In this work, we focus on clustering Web-searched pages in the root set R with a new page similarity measurement. The new page similarity is measured by the page correlation degrees within the concerned page source. For simplicity and better understanding of this new similarity, we divide the correlation matrix C into four blocks (sub-matrices) as follows:

$$C = (c_{ij})_{(m+n)\times(m+n)} = \begin{array}{c} \\ \mathrm{R} \\ \mathrm{V} \end{array} \begin{array}{c} \begin{array}{cc} \mathrm{R} & \mathrm{V} \end{array} \\ \begin{bmatrix} \textcircled{1} & \textcircled{2} \\ \textcircled{3} & \textcircled{4} \end{bmatrix} \end{array}_{(m+n)\times(m+n)}$$

The elements in sub-matrix 1 represent the correlation relationships between the pages in R. Similarly, the elements in sub-matrices 2 and 3 represent the correlation relationships between the pages in R and V, and sub-matrix 4 gives the correlation relationships between the pages in V. It can be seen that the correlation degrees related with the pages in R are located in three sub-matrices 1, 2 and 3. Therefore, the similarity measurement for the pages in R only refers to the elements in these three sub-matrices.

Note: If the similarity between any two pages in the whole source space S is to be measured, the whole correlation matrix C will be used and the similarity definition is the same as follows.

In the correlation matrix C, the row vector that corresponds to each page i in R is in the form of

$$row_i = (c_{i,1}, c_{i,2}, ..., c_{i,m+n}), \quad i = 1,2,...,m .$$

From the construction of matrix C, it is known that row_i represents *out-link* relationship of page i in R with all the pages in S, and element values in this row vector indicate the correlation degrees of this page to the linked pages. Similarly, the column vector that is in the form of

$$col_i = (c_{1,i}, c_{2,i}, ..., c_{m+n,i}), \quad i = 1,2,...,m ,$$

represents *in-link* relationship of page i in R with all the pages in S, and its element values indicate the correlation degrees from the pages in S to page i.

Each page i in R, therefore, is represented as two correlation vectors: row_i and col_i. For any two pages i and j in R, their *out-link similarity* is defined as

$$sim_{i,j}^{out} = \frac{(row_i, row_j)}{\| row_i \| \cdot \| row_j \|},$$

where

$$(row_i, row_j) = \sum_{k=1}^{m+n} c_{i,k} c_{j,k}, \qquad \| row_i \| = (\sum_{k=1}^{m+n} c_{i,k}^2)^{1/2}.$$

Similarly, their *in-link similarity* is defined as

$$sim_{i,j}^{in} = \frac{(col_i, col_j)}{\| col_i \| \cdot \| col_j \|}.$$

Then the similarity between any two pages i and j in R is defined as

$$sim(i, j) = \alpha_{ij} \cdot sim_{i,j}^{out} + \beta_{ij} \cdot sim_{i.j}^{in}, \tag{5.3}$$

where α_{ij} and β_{ij} are the weights for out-link and in-link similarities respectively.

The similarity weights α_{ij} and β_{ij} are determined dynamically as:

$$\alpha_{ij} = \frac{\| row_i \| + \| row_j \|}{MOD_{ij}}, \quad \beta_{ij} = \frac{\| col_i \| + \| col_j \|}{MOD_{ij}},$$

where $MOD_{ij} = \| row_i \| + \| row_j \| + \| col_i \| + \| col_j \|$. As a special case, for any pair of pages *i* and *j* in *R*, if their out-link modes $\|row\|$ and in-link modes $\|col\|$ are approximately the same, the weights α_{ij} and β_{ij} could be simply chosen as $(\alpha_{ij}, \beta_{ij}) = (1/2, 1/2)$.

It is argued that hyperlink transitivity would bring noise factors into the page similarity measurement. One source of the noise factors is the noise pages that are not query topic related but are densely linked with each other in the page source. The noise pages will have unreasonably high page weights and mislead the page correlation degrees. These noise pages, however, can be eliminated from the page source by many existing algorithms, such as (Bharat and Henzinger 1998; Hou and Zhang 2002; Hou et al. 2002). Therefore, in this work, we can reasonably assume that the pages in the page source are query topic related. Another noise factor source is taking every path between two pages into consideration in the page correlation degree measurement. Under this situation, minor page correlations between two pages could be accumulated such that the final correlation degrees are unreasonably increased in some cases. In this work, however, the page correlation degree only takes the shortest path between two pages into account, so the noise factors are omitted. On the other hand, the page correlation degree decreases quickly with the increase of the shortest path length (distance). Therefore, the contribution of hyperlink transitivity to the page correlation degree is minor if there exists a long distance between two pages. This would also eliminate many noise factors in the page similarity measurement.

The above page similarity measurement is derived from the page correlation degrees, rather than the direct hyperlinks between the pages. It seems that

this idea comes from the co-citation analysis. The intension of this new similarity, however, is different from that of co-citation analysis based similarities. In the co-citation analysis, the influence of each page to the similarity measurement is the same, and the similarity between any two pages only depends on the number of direct common pages (common parent and child pages). In this new similarity measurement (5.3), the influence of each page to the similarity is different, which is reflected by the page weight. Furthermore, this new similarity not only depends on the number of direct common pages, but also depends on the number of indirect common pages and the correlation degrees of the involved pages. Fig. 5.3 gives an example that shows these intrinsic differences.

In this example, the values for the weights α_{ij} and β_{ij} are simply chosen as $(\alpha_{ij}, \beta_{ij}) = (1/2, 1/2)$. The number in a pair of parentheses beside a page number is the weight of that page, and the number beside a link arrow indicates the correlation degree between the two pages if the correlation factor $F = 1/2$. For the situation (a) in this example, if the co-citation analysis is applied, the similarity of pages 1 and 2 is the same as that of pages 2 and 3. But, if the similarity measurement (5.3) is applied to this situation, we get $sim(1,2) = 0.09$ and $sim(2,3) = 0.17$. The similarity $sim(2,3)$ is greater than $sim(1,2)$ because the common page 5 of pages 2 and 3 are more important than common page 4 of pages 1 and 2. The simple co-citation analysis, however, is unable to reflect this difference.

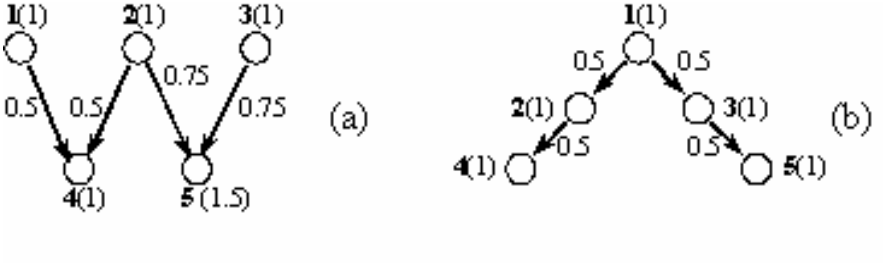


Fig. 5.3. Example of the similarity measurement

For the situation (b), the similarity between pages 4 and 5 is zero if the co-citation analysis is applied, because they have no direct common (parent) pages. Actually, there still exists a relative weak relationship between them via page 1, and their similarity should not be zero. By applying (5.3) to this situation, we get $sim(4,5) = 0.02$, which reflects the influence of the indirect common pages to the page similarity measurement. If pages 2 and 3 have higher page weights, $sim(4,5)$ would be higher.

5.2 Hierarchical Web Page Clustering

With the page similarity measurement and the correlation matrix C, a hierarchical Web page clustering algorithm could be established. This hierarchical clustering algorithm consists of two phases. The first one is single layer clustering, in which the pages in R are clustered at the same level without hierarchy. The second phrase is hierarchical clustering, in which the pages in the clusters produced by the first phase are clustered further to form a cluster hierarchical structure. Fig. 5.4 gives this hierarchical clustering diagram.

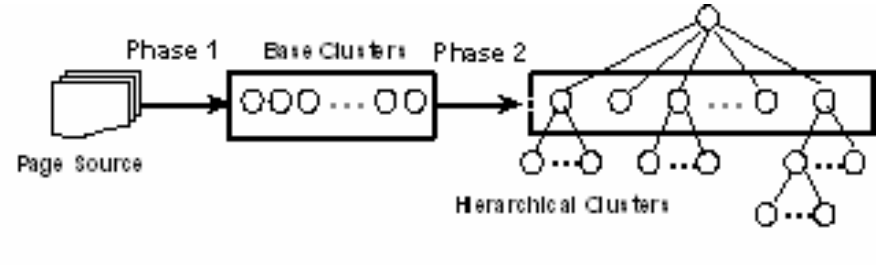


Fig. 5.4. Hierarchical clustering diagram

The details of the hierarchical clustering algorithm are described as follows.

Phase 1: Single Layer Clustering
[**Input**]: A set of Web pages $R = \{p_1, p_2, \ldots, p_m\}$, clustering threshold T.
[**Output**]: A set of clusters $CL = \{CL_i\}$.
[**Algorithm**]: *BaseCluster*(R, T)
Step 1. Select the first page p_1 as the initial cluster CL_1 and the centroid of this cluster, i.e. $CL_1 = \{p_1\}$ and $CE_1 = p_1$.
Step 2: For each page $p_i \in R$, calculate the similarity between p_i and the centroid of each existing cluster $sim(p_i, CE_j)$.
Step 3: If $sim(p_i, CE_k) = \max_j(sim(p_i, CE_j)) > T$, then add p_i to the cluster CL_k and recalculate the centroid CE_k of this cluster that consists of two vectors

$$CE_k^{row} = \frac{1}{|CL_k|}\sum_{j \in CL_k} row_j \,, CE_k^{col} = \frac{1}{|CL_k|}\sum_{j \in CL_k} col_j \,,$$

where $|CL_k|$ is the number of pages in CL_k. Otherwise, p_i itself initiates a new cluster and is the centroid of this new cluster.

Step 4: If there are still pages to be clustered (i.e. pages that have not been clustered or a page that itself is a cluster), go back to Step 2 until all cluster centroids no longer change.
Step 5: Return clusters $CL = \{CL_i\}$.

The above phase 1 of the clustering algorithm produces a set of single layer clusters called *base clusters*. Recursively applying the above algorithm, with increasing clustering threshold T, to each base cluster would produce downward hierarchical clusters. This procedure is stopped when the number of pages in each leaf cluster is below a certain predefined threshold NP. Then the whole hierarchical cluster structure is produced. The procedure is described as phase 2 of the clustering algorithm.

Phase 2: Hierarchical Clustering
[**Input**]: A set of base clusters $CL = \{CL_i\}$, parameter NP and clustering threshold T in phase 1.
[**Output**]: Hierarchical clusters $HCL = \{HCL_i\}$.
[**Algorithm**]: *HierarchyCluster*(CL, NP, T)
Step 1: Set $HCL = CL$, and let CL to be the set of clusters at layer 1 (base layer), i.e. $CL^1 = \{CL_i^1\} = \{CL_i\}$. Assign $l = 1$ and $T' = T$.
Step 2: Recursively increase T', l and call algorithm *BaseCluster*(CL_i^l, T') for those clusters CL_i^l in CL^l that contain more than NP pages. Add the clusters at each layer to HCL.
Step 3: Return the produced set of hierarchical clusters HCL.

The clustering threshold T in the algorithm is determined by practical requirement. It should guarantee that the pages are clustered into a reasonable number of clusters. For example, T could be chosen as the average page similarity of all the pages in R. The increase rate for the hierarchical clustering threshold T' could be chosen as a certain percentage of the threshold T.

The parameter NP (e.g. 10) is used to control the number of downward levels of the hierarchical cluster structure. If the number of pages in a cluster $\leq NP$, this cluster should not be divided into some smaller clusters (at a lower level) any more. If the hierarchical cluster structure is for Web page navigation, the value of NP is usually determined by the number of pages in a cluster that users can tolerate for navigation. Proper NP value would also be able to reduce the execution cost of the algorithm.

It can be inferred from the phase 1 of the algorithm that a page in R only belongs to a cluster. In practice, a page might belong to multiple clusters. This requirement can be easily met by only changing the clustering condition in the step 3 of the phase 1, i.e. changing the condition " If $sim(p_i, CE_k) =$

$\max_j(sim(p_i, CE_j)) > T$" to "If $sim(p_i, CE_k) > T$". For computation simplicity, we still assume that a page only belongs to a cluster.

As stated in (Wen et al. 2001), for this kind of hierarchical clustering algorithm, it has been proven (Wang 1997) that the algorithm is independent of the order in which the pages are presented to the algorithm if the pages are properly normalized. Since the page normalization is guaranteed in the similarity measurement (3), the above hierarchical clustering algorithm is independent of the page order. It is not difficult to prove that the complexity of this algorithm is $O(M*N*logN)$, where M is the number of generated clusters and N is the number of pages to be clustered.

5.3 Matrix-Based Clustering Algorithms

In this work, we still focus on the page source constructed in Sect. 5.1.1, and adopt the concepts and symbols in that section. For the concerned page source S, we suppose the size of the root set R is m, the size of the vicinity set $V = BV \cup FV$ is n. With the new page similarity measurement (3), a new $m \times m$ symmetric matrix SM, called *similarity matrix* for R, can be constructed as $SM = (sm_{i,j})_{m \times m}$ for all the pages in the root set R, where

$$sm_{i,j} = \begin{cases} sim(i, j) & \text{if } i \neq j, \\ 1 & \text{if } i = j. \end{cases}$$

The matrix-based Web page clustering is implemented by partitioning the page similarity matrix. With the partition of the similarity matrix, the pages are accordingly clustered into clusters. To guarantee the effectiveness of matrix-based algorithms in page clustering, it is needed to conduct similarity matrix permutation before partition.

5.3.1 Similarity Matrix Permutation

The similarity matrix permutation is to put those closely related pages together in the similarity matrix SM, such that the page position in the matrix more reasonably reflects the relevance between pages within the whole range of concerned pages. For measuring how close two pages are related, we define the *affinity* of two pages i and $j \in R$ as:

$$AF(i, j) = \sum_{k=1}^{m} sm_{i,k} \times sm_{j,k}$$

The corresponding affinity matrix is denoted as *AF*. Two pages with higher affinity would be more related with each other and should have more of a chance to be put in the same cluster. However, since the pages in the matrix have mutual effects, the final page positions of the similarity matrix should be determined within the whole range of concerned pages in the matrix. For globally optimising the page position, we define the *global affinity* of matrix *SM* as

$$GA(SM) = \sum_{i=1}^{m} \sum_{j=1}^{m} AF(i, j)[AF(i, j-1) + AF(i, j+1)] \quad (5.4)$$

where $AF(i,0) = AF(i, m+1) = 0$. $GA(SM)$ contains all the affinities of pages in *R* with their neighbouring pages. The higher the $GA(SM)$, the more likely the closely related pages are put together as neighbouring pages. The purpose of the similarity matrix permutation is to get the highest $GA(SM)$, under which the close related pages are located closely to each other in the matrix.

The highest $GA(SM)$ can be obtained by swapping the positions of every pair of columns (accordingly rows) in matrix *AF*. In fact, we denote the permuted affinity matrix as *PA*. Similar to the work in (Özsu and Valduriez 1991), the algorithm for generating *PA* with the highest $GA(SM)$ consists of three steps:

1. **Initiation**. Place and fix one of the columns of *AF* arbitrarily into *PA*.
2. **Iteration**. Pick each of the remaining *m-i* columns (where *i* is the number of columns already placed in *PA*) and try to place them in the remaining *i*+1 positions in the *PA*. Choose the placement that makes the greatest contribution to the global affinity. Continue this step until no more columns remain to be placed.
3. **Row ordering**. Once the column ordering is determined, the placement of the rows should also be changed so that their relative positions match the relative positions of the columns.

When the highest $GA(SM)$ is achieved, the page positions in *SM* are permuted according to the actual page positions in the permuted affinity matrix *PA*. As a result, the closely related pages are located closely to each other in the new permuted similarity matrix. For simplicity, hereafter, we still denote this permuted similarity matrix as *SM*.

Fig. 5.5 gives an example of the similarity matrix permutation. There are nine pages (marked P_1, P_2, … , P_9) in this example. The original and permuted similarity matrices are shown in Fig. 5.5(a) and (b) separately. It can be seen that the closely related pages are located closely to each other in the permuted similarity matrix (b) with the highest global affinity.

5.3.2 Clustering Algorithm from a Matrix Partition

Matrix-based page clustering is implemented by decomposing the permuted matrix *SM* into four sub-matrices along its main diagonal, i.e.

$$SM = (sm_{i,j})_{m\times m} = \begin{pmatrix} SM_{1,1} & SM_{1,2} \\ SM_{2,1} & SM_{2,2} \end{pmatrix}_{m\times m}.$$

Since the rows (or columns) of the permuted similarity matrix *SM* correspond to the pages to be clustered, the pages corresponding to the sub-matrices $SM_{1,1}$ and $SM_{2,2}$ form two clusters, while the elements of sub-matrix $SM_{1,2}$ (or $SM_{2,1}$, since $SM^T_{2,1} = SM_{1,2}$) represent similarities between the pages that separately belong to these two clusters.

It is clear that the partition of matrix *SM* is equivalent to finding a dividing point *D* along the main diagonal of *SM*. To find this dividing point *D*, we define a measurement for the sub-matrix $SM_{p,q}$ ($1\le p, q \le 2$) as

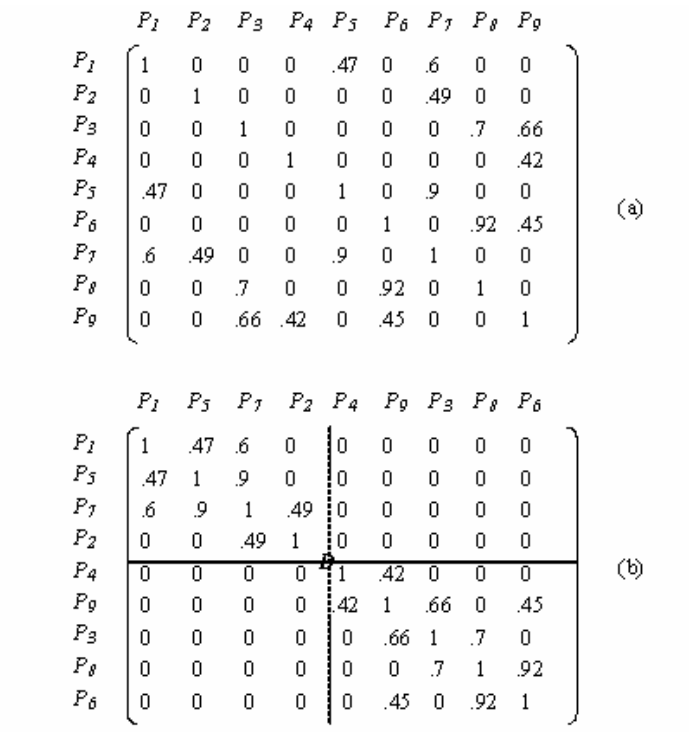
$$M(SM_{p.q}) = \sum_{i=(p-1)*d+1}^{d+(m-d)*(p-1)} \sum_{j=(q-1)*d+1}^{d+(m-d)*(q-1)} sm_{i,j}\ ,\ 1\le p, q \le 2,$$

where *d* stands for the row (and column) number of *D*. The dividing point *D* is selected such that the following function is maximized

$$.F_D = M(SM_{1,1})* M(SM_{2,2}) - M(SM_{1,2})* M(SM_{2,1}) \qquad (5.5)$$

Therefore, the determination of the dividing point *D* makes the pages with high affinity to be located in the same cluster (sub-matrix), and the similarity between the clusters to be low. Once the dividing point *D* is determined, two clusters $SM_{1,1}$ and $SM_{2,2}$ are settled down. For instance, the pages in the example of Fig. 5.5 are clustered into two clusters: $SM_{1,1} = \{P_1, P_5, P_7, P_2\}$, $SM_{2,2} = \{P_4, P_9, P_3, P_8, P_6\}$, while the row (an column) number of *D* is 4.

This matrix partition could be recursively applied to the matrices $SM_{1,1}$ and $SM_{2,2}$ until the number of pages in every new produced cluster is less than or equal to a preferred number *pn* (e.g. 20). All clusters produced during this procedure hierarchically cluster the Web pages. Fig. 5.6 shows this clustering diagram. The clustering procedure is depicted as the following Algorithm1, *Clustering1*, where | *SM* | stands for the number of rows (columns) of the square matrix *SM*.



	P_1	P_2	P_3	P_4	P_5	P_6	P_7	P_8	P_9
P_1	1	0	0	0	.47	0	.6	0	0
P_2	0	1	0	0	0	0	.49	0	0
P_3	0	0	1	0	0	0	0	.7	.66
P_4	0	0	0	1	0	0	0	0	.42
P_5	.47	0	0	0	1	0	.9	0	0
P_6	0	0	0	0	0	1	0	.92	.45
P_7	.6	.49	0	0	.9	0	1	0	0
P_8	0	0	.7	0	0	.92	0	1	0
P_9	0	0	.66	.42	0	.45	0	0	1

(a)

	P_1	P_5	P_7	P_2	P_4	P_9	P_3	P_8	P_6
P_1	1	.47	.6	0	0	0	0	0	0
P_5	.47	1	.9	0	0	0	0	0	0
P_7	.6	.9	1	.49	0	0	0	0	0
P_2	0	0	.49	1	0	0	0	0	0
P_4	0	0	0	0	1	.42	0	0	0
P_9	0	0	0	0	.42	1	.66	0	.45
P_3	0	0	0	0	0	.66	1	.7	0
P_8	0	0	0	0	0	0	.7	1	.92
P_6	0	0	0	0	0	.45	0	.92	1

(b)

Fig. 5.5. (a) A similarity matrix. (b) The permuted matrix of (a)

[**Algorithm1**] *Clustering1* (*SM, pn*)
[**Input**] *SM*: similarity matrix; *pn*: preferred page number in each cluster;
[**Output**] $CL = \{CL_i\}$: a set of hierarchical clusters;
Begin

Set $CL = \varnothing$; Permute *SM* such that (5.4) is maximized;
Decompose *SM* such that (5.5) is maximized;
If $| SM_{1,1} | \leq pn$, **then** do
converting $SM_{1,1}$ into the next CL_i; $CL = CL \cup \{CL_i\}$;
else do
converting $SM_{1,1}$ into the next CL_i; $CL = CL \cup \{ CL_i \}$;
Clustering1 ($SM_{1,1}$, *pn*);
If $| SM_{2,2} | \leq pn$, **then** do
converting $SM_{2,2}$ into the next CL_i; $CL = CL \cup \{CL_i\}$;
else do
converting $SM_{2,2}$ into the next CL_i; $CL = CL \cup \{ CL_i \}$;
Clustering1 ($SM_{2,2}$, *pn*);
Return *CL*;

End

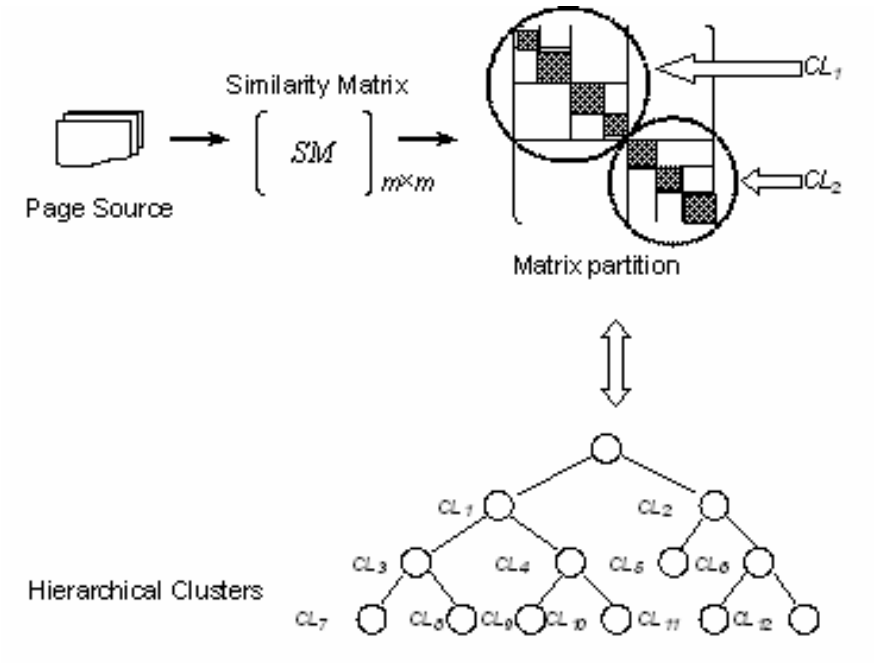


Fig. 5.6. Matrix-based hierarchical clustering diagram

5.3.3 Cluster-Overlapping Algorithm

For the above algorithm *Clustering1*, there exists no overlapping among the clusters that are produced at the same level. Each page belongs to only one of the clusters at the same level. In practice, however, it is reasonable that a page might belong to several same level clusters. On the other hand, the non-zero element values in $SM_{1,2}$ or $SM_{2,1}$ represent the similarity between two pages, named *cross-related pages*, that belong to two different clusters. If a cross-related page in one cluster has a higher similarity with another cluster, it is possible for this page to be added to another cluster (sub-matrix) to form a new cluster in case it will be missed out. For these reasons, the hierarchical clustering algorithm *Clustering 1* could be improved such that the cluster overlapping among the same level clusters is permitted.

To determine whether a cross-related page in one cluster could be added to another cluster, we define a *centroid* of the cluster $SM_{p,p}$ ($1 \leq p \leq 2$) as $CE(SM_{p,p}) = \{CE^{row}(SM_{p,p}), CE^{col}(SM_{p,p})\}$, which consists of two vectors

(row and column vectors) that are constructed from the correlation matrix C as

$$CE^{row}(SM_{p,p}) = \frac{1}{|SM_{p,p}|}\sum_{j\in SM_{p,p}} row_j \,, CE^{col}(SM_{p,p}) = \frac{1}{|SM_{p,p}|}\sum_{j\in SM_{p,p}} col_j \tag{5.6}$$

The centroid of a cluster is a logical page representing this cluster. For a pair of cross-related pages $p\in SM_{1,1}$ and $q\in SM_{2,2}$, if $sim(p, CE(SM_{2,2}\,)) \geq t$, then page p could be added to $SM_{2,2}$ to form a new cluster (sub-matrix) $SM'_{2,2}$ with the dimension being increased by 1, where t is a threshold defined as the average non-zero similarities in $SM_{1,2.}$ The page q could be treated in the similar way. The following Algorithm2 *Extending* depicts this cross-related page treatment.

[**Algorithm2**] *Extending* (*SM*)
[**Input**] *SM*: similarity matrix with sub-matrices $SM_{1,1}$, $SM_{2,2}$, $SM_{1,2}$ and $SM_{2,1}$;
[**Output**] $SM'_{1,1}$, $SM'_{2,2}$: new sub-matrices (clusters) with some added cross-related pages;
Begin
Compute the centroids $CE(SM_{1,1})$ and $CE(SM_{2,2}\,)$ according to (5.6);
Compute the threshold t, which is the average non-zero similarities in $SM_{1,2}$;
Set N_1=[|$SM_{1,1}$|*0.15]; N_2 = [|$SM_{2,2}$| * 0.15]; $N = \min(N_1, N_2)$;
Construct page set P={p|at least one sm_p, $j\neq 0$, $1\leq p\leq d$, $d+1\leq j\leq m$};
Construct page set Q={q|at least one sm_i,$q\neq 0$, $1\leq i\leq d$, $d+1\leq q\leq m$};
Compute P_SM_{22} = {$sim(p, CE(SM_{2,2})|p\in P$, $sim(p, CE(SM_{2,2}\,)\geq t$};
Compute Q_SM_{11} = {sim(q, $CE(SM_{1,1})|q\in Q$, sim(q, $CE(SM_{1,1}) \geq t$};
Add up to N pages in $SM_{2,2}$ that correspond to the N highest values in Q_SM_{11} into $SM_{1,1}$ to form a new sub-matrix $SM'_{1,1}$;
Add up to N pages in $SM_{1,1}$ that correspond to the N highest values in P_SM_{22} into $SM_{2,2}$ to form a new sub-matrix $SM'_{2,2}$;
Return $SM'_{1,1}$ and $SM'_{2,2}$;
End

The parameter d is the row (or column) number of the dividing point D in SM. The parameter N in this algorithm is used to restrict the number of pages to be added to $SM_{1,1}$ and $SM_{2,2}$, which guarantees the recursive execution of the matrix partition. This parameter, on the other hand, also guarantees that the added cross-related pages could not change (or dominate) the main property of the original clusters, i.e. the most number of cross-related pages added to a cluster is less than or equal to 15% of the original page number in this

cluster. This percentage could be adjusted according to the practical requirements.

When n pages that belong to cluster $SM_{2,2}$ are added into cluster $SM_{1,1}$, the corresponding new sub-matrix $SM'_{1,1}$ is formed by adding n columns of $SM_{1,2}$ and n rows of $SM_{2,1}$ into the original $SM_{1,1}$ with the dimension being increased by n. These added columns and rows correspond to these n added pages. The main diagonal elements of the newly produced $n \times n$ lower-right sub-matrix of $SM'_{1,1}$ are set to 1, and other elements in this sub-matrix are set to 0. The construction of $SM'_{1,1}$ is intuitively shown in Fig. 5.7 For the construction of $SM'_{2,2}$, the procedure is the same.

Based on the above cluster overlapping treatment algorithm *Extending*, the matrix-based hierarchical clustering algorithm with cluster overlapping is depicted as the following Algorithm3 *Clustering2*.

[**Algorithm3**] *Clustering2* (*SM, pn*)
[**Input**] *SM*: similarity matrix; *pn*: preferred page number in each cluster;
[**Output**] $CL = \{CL_i\}$: a set of hierarchical clusters;
Begin
Set $CL = \varnothing$; Permute *SM* such that (5.4) is maximized;
Decompose *SM* such that (5.5) is maximized;
$\{SM'_{1,1}, SM'_{2,2}\}$ = *Extending* (*SM*);
If $| SM'_{1,1} | \le pn$, **then** do
converting $SM'_{1,1}$ into the next CL_i; $CL = CL \cup \{CL_i\}$;
else do
converting $SM'_{1,1}$ into the next CL_i; $CL=CL\cup\{CL\}$;
Clustering2 ($SM'_{1,1}$, *pn*);
If $| SM'_{2,2} | \le pn$, **then** do
converting $SM'_{2,2}$ into the next CL_i; $CL = CL\cup\{CL_i\}$;
else do converting $SM'_{2,2}$ into the next CL_i; $CL=CL\cup\{CL_i\}$;
Clustering2 ($SM'_{2,2}$, *pn*);
Return *CL*;
End

This clustering algorithm enables some pages in R to be clustered into several same level clusters, which is reasonable in practice and enables users to find some pages from different paths in the hierarchical cluster structure. It is not difficult to prove that the complexity of the above clustering algorithms is $O(m^2)$, where m is the number of pages to be clustered.

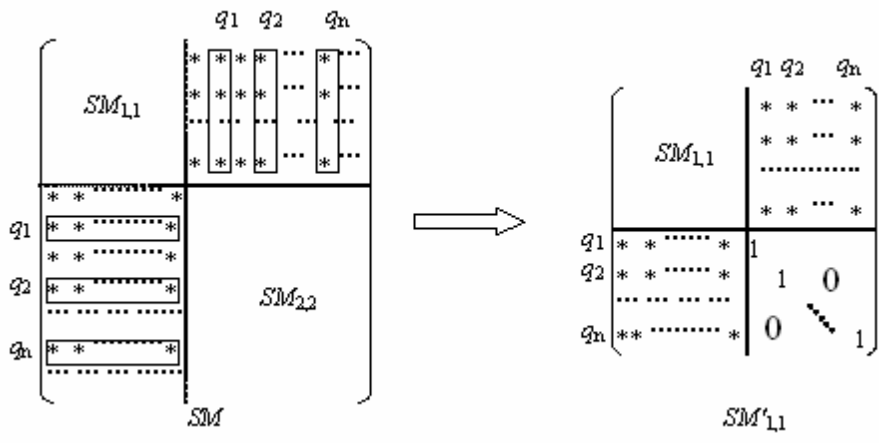


Fig. 5.7. Construction of new sub-matrix $SM'_{1,1}$

5.4 Co-Citation Algorithms

The citation and co-citation analysis were originally developed for scientific literature index and clustering, and then extended to the Web page analysis. For better understanding of the algorithms to be proposed, we firstly present some background knowledge of the citation and co-citation analysis, and then give the Extended Co-Citation algorithm for relevant page finding.

5.4.1 Citation and Co-Citation Analysis

The citation analysis was developed in information science as a tool to identify core sets of articles, authors, or journals of particular fields of study (Larson 1996). The research has long been concerned with the use of citations to produce quantitative estimates of the importance and impact of individual scientific articles, journals or authors. The most well-known measure in this field is Garfield's *impact factor (Garfield 1972)*, which is the average number of citations received by papers (or journals) and was used as a numerical assessment of journals in Journal Citation Reports of the Institution for Scientific Information.

The co-citation analysis has been used to measure the similarity of papers, journals or authors for clustering. For a pair of documents p and q, if they are both cited by a common document, documents p and q is said to be *co-cited*. The number of documents that cite both p and q are referred to as *co-citation degree* of documents p and q. The similarity between two documents is

measured by their co-citation degree. This type of analysis has been shown to be effective in a broad range of disciplines, ranging from author co-citation analysis of scientific subfields to journal co-citation analysis. For example, Chen and Carr (Chen and Carr 1999) used author co-citation analysis to cluster the authors, as well as the the research fields. In the context of the Web, the hyperlinks are regarded as citations beween the pages. If a Web page p has a hyperlink to another page q, page q is said to be cited by the page p. In this sense, citation and co-citation analyses are smoothly extended to the Web page hyperlink analysis. For instance, Larson (Larson 1996), Pitkow and Pirolli (Pitkow and Pirolli 1997) have used the co-citation to meaure the Web page similarities.

The above co-citation analyses, whether for scientific literatures or for Web pages, is mainly for the purpose of clustering, and the page source to which the co-citation analysis is applied is usually a pre-known page set or a Web site. For example, the page source in (Pitkow and Pirolli 1997) was the pages in a Web site of Georgia Institute of Technology, and the page source in (Larson 1996) was a set of pages in *Earth Science* related Web sites. When the co-citation analysis is applied for relevant page finding, however, the situation is different. Since there exists no pre-known page source for the given page and co-citation analysis, the success of co-citation analysis mainly depends on how to effectively construct a page source with respect to the given page. Meanwhile, the constructed page source should be rich in related pages with a reasonable size.

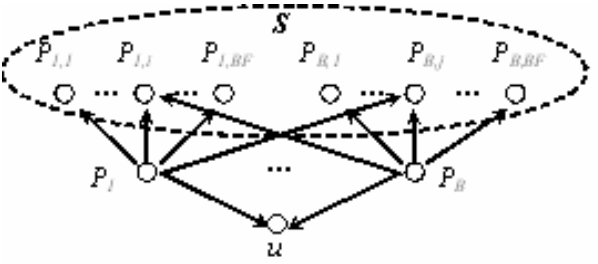


Fig. 5.8. Page source S for the u in *DH Algorithm*

Dean and Henzinger (Dean and Henzinger 1999) proposed a co-citation algorithm to find the relevant pages. Hereafter, we denote it as the *DH Algorithm*. In their work, for a given page (URL) u, the page source S with respect to u is constructed in the following way: the algorithm firstly chooses up to B (e.g. 2000) arbitrary parents of u; for each of these parents p, it adds to S up to BF (e.g. eight) children of p that surround the link from p to u. The elements of S are siblings of u as indicated in Fig. 5.8. Based on this page source S, the co-citation algorithm for finding relevant pages is as follow: for each page s

in *S*, the co-citation degree of *s* and *u* is determined; the algorithm finally returns the 10 pages that have the highest co-citation degrees with *u* as the relevant pages.

Although the *DH Algorithm* is simple and the page source is of a reasonable size (controlled by the parameters *B* and *BF*), the page source construction only refers to the parents of the given page *u*. It is actually based on an assumption that the possible related pages fall into the set of siblings of *u*. Since the child pages of *u*, and accordingly the page set derived from these child pages, are not taken into account in the page source construction, many semantically related pages might be excluded in the page source and the final results may be unsatisfactory. This is because the semantic relationship conveyed by the hyperlinks between two pages is mutual. If a page *p* is said to be semantically relevant (via hyperlinks) to another page *q*, page *q* could also be said to be semantically relevant to page *p*. From this point of view, the children of the given page *u* should be taken into consideration in the page source construction.

5.4.2 Extended Co-Citation Algorithms

For a given page *u*, its semantic details are most likely to be given by its in-view and out-view (Mukherjea and Hara 1997). The in-view is a set of parent pages of *u*, and out-view is a set of child pages of *u*. In other words, the relevant pages with respect to the given page are most likely to be brought into the page source by the in-view and out-view of the given page. The page source for finding relevant pages, therefore, should be derived from the in-view and out-view of the given page, so that the page source is rich in the related pages.

Given a Web page *u*, its parent and child pages could be easily obtained. Indeed, the child pages of *u* can be obtained directly by accessing the page *u*; for the parent pages of *u*, one way to obtain them is to issue an *AltaVista* query of the form *link: u*, which returns a list of pages that point to *u (Bharat and Henzinger 1998)*. The parent and child pages of the given page could also be provided by some professional servers, such as the Connectivity Server (Bharat et al. 1998). After the parent and child pages of *u* are obtained, it is possible to construct a new page source for *u* that is rich in related pages. The new page source is constructed as a directed graph with edges indicating hyperlinks and nodes representing the following pages:

1. Page *u*,
2. Up to *B* parent pages of *u*, and up to *BF* child pages of each parent page that are different from *u*,

3. Up to F child pages of u, and up to FB parent pages of each child page that are different from u.

The parameters B, F, BF and FB are used to keep the page source to a reasonable size. In practice, we choose $B = FB = 200$, $F = BF = 40$. This new page source structure is presented intuitively in Fig. 5.9. Before giving the Extended Co-Citation algorithm for finding relevant pages, we firstly define the following concepts.

Definition 1: Two pages p_1 and p_2 are *back co-cited* if they have a common parent page. The number of their common parents is their *back co-citation degree* denoted as $b(p_1, p_2)$. Two pages p_1 and p_2 are *forward co-cited* if they have a common child page. The number of their common children is their *forward co-citation degree* denoted as $f(p_1, p_2)$.

Definition 2: The pages are *intrinsic pages* if they have same page domain name.

Definition 3: (Dean and Henzinger 1999): Two pages are *near-duplicate pages* if (a) they each have more than 10 links and (b) they have at least 95% of their links in common.

Based on the above concepts, the complete Extended Co-Citation algorithm to find relevant pages of the given Web page u is as follow:

Step 1: Choose up to B arbitrary parents of u.

Step 2: For each of these parents p, choose up to BF children (different from u) of p that surround the link from p to u. Merge the intrinsic or near-duplicate parent pages, if they exit, as one whose links are the union of the links from the merged intrinsic or near-duplicate parent pages, i.e. let P_u be a set of parent pages of u,

$P_u = \{p_i \mid p_i$ *is a parent page of u without intrinsic and near-duplicate pages,* $i \in [1, B]\}$,

let $S_i = \{s_{i,k} \mid s_{i,k}$ *is a child page of page* p_i, $s_{i,k} \neq u$, $p_i \in P_u$, $k \in [1, BF]\}$, $i \in [1, B]$.

Then Step 1 and 2 produce the following set

$$BS = \bigcup_{i=1}^{B} S_i .$$

Step 3: Choose first F children of u.

Step 4: For each of these children c, choose up to FB parents (different from u) of c with highest in-degree. Merge the intrinsic or near-duplicate child pages, if they exist, as one whose links are the union of the links to the merged intrinsic or near-duplicate child pages, i.e. let C_u be a set of child pages of u,

$C_u = \{c_i \mid c_i$ *is a child page of u without intrinsic and near-duplicate pages,* $i \in [1, F]\}$,

let $A_i = \{a_{i.k} \mid a_{i,k}$ *is a parent page of page* c_i, $a_{i,k}$ *and u are neither intrinsic nor near-duplicate pages*, $c_i \in C_u$, $k \in [1,FB]\}$, $i \in [1, F]$.

Then Step 3 and 4 produce the following set

$$FS = \bigcup_{i=1}^{F} A_i \ .$$

Step 5: For a given selection threshold δ, select pages from *BS* and *FS* such that their back co-citation degrees or forward co-citation degrees with *u* are greater than or equal to δ. These selected pages are relevant pages of *u*, i.e., the relevant page set *RP* of *u* is constructed as:

$$RP = \{\ p_i \mid p_i \in BS \text{ with } b(p_i, u) \geq \delta \ \text{ OR } \ p_i \in FS \text{ with } \ f(p_i, u) \geq \delta\ \}.$$

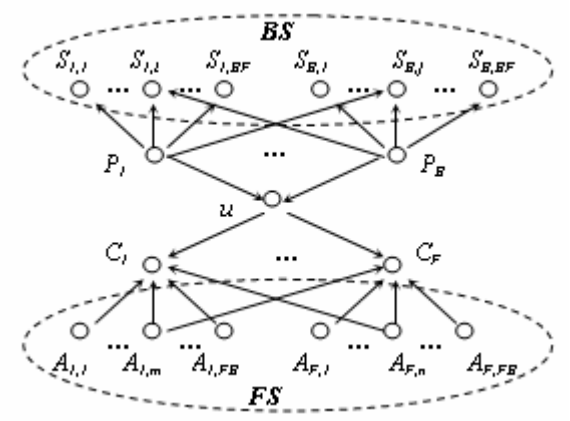


Fig. 5.9. Page source structure for the Extended Co-Citation algorithm

It can be seen from this algorithm that, in the parent page set P_u and child page set C_u of *u*, the intrinsic or near-duplicate pages are merged as one. This treatment is necessary for the success of the algorithm. Firstly, this treatment can prevent the searches from being affected by malicious hyperlinks. In fact, for the pages in a Web site (or server) that are hyperlinked purposely to maliciously improve the page importance for Web search, if they are imported into the page source as the parent pages of the given page u, their children (the siblings of *u*) most likely come from the same site (or server), and the back co-citation degrees of these children with *u* would be unreasonably increased. With the merger of the intrinsic parent pages, the influence of the pages from the same site (or server) is reduced to a reasonable level (i.e. the back co-citation degree of each child page with u is only 1) and the malicious hyperlinks are shielded off. For example, in Fig. 5.10, suppose the parent pages P_1, P_2, P_3 and their children $S_{1.1}$, …, $S_{3.2}$ be intrinsic pages. In situation

(a), the back co-citation degree of page $S_{2.2}$ with u is unreasonably increased to 3, which is the ideal situation the malicious hyperlink creators would like. The situation is the same for the pages $S_{1.2}$ and $S_{3.1}$. With the above algorithm, the situation (a) is treated as the situation (b) where P is a logic page representing the union of P_1, P_2, P_3, and the contribution of each child page from the same site (or server) to the back co-citation degree with u is only 1, no matter how tightly these intrinsic pages are linked together.

Secondly, for those pages that are really relevant to the target page u and located in the same domain name, such as those in Web sites that are concerned about certain topics, the above intrinsic page treatment would probably decrease their relevance to the given page u. However, since we consider the page relevance to the given page within a local Web community (page source), not just within a specific Web site or server, this intrinsic page treatment is still reasonable in this sense. Under this circumstance, there exists a trade-off between avoiding malicious hyperlinks and keeping as much useful information as possible. Actually, if such pages are still considered as relevant pages within the local Web community, they would be finally identified by the algorithm. The above reasons for intrinsic parent page treatment are the same for the intrinsic child page treatment, as well as the near-duplicate page treatment.

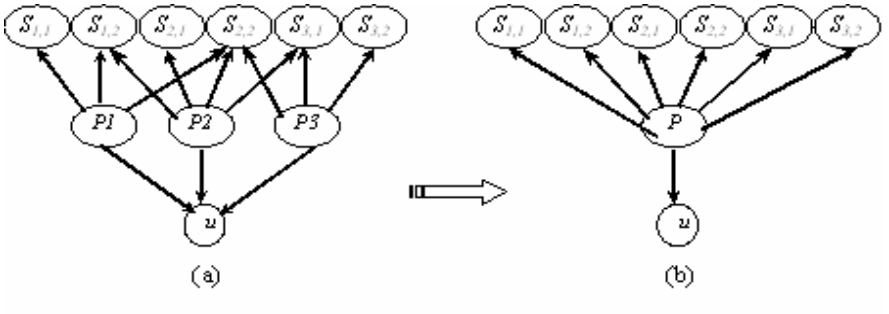


Fig. 5.10. An example of intrinsic page treatment

It is also worth noting that even if the given page u contains active links (i.e. links to hub pages that are also cited by other pages), the algorithm, especially the pages set A_i, can also shield off the influence of malicious hyperlinks from the same site or server or mirror site of u. On the other hand, however, this page set A_i would probably filter those possible relevant pages that come from the same domain name of u. The trade-off between avoiding malicious hyperlinks and keeping useful information still exists in this circumstance. If the algorithm is only used within a specific Web site or domain name, it can be simplified without considering the intrinsic page treatment. In

other words, in the Extended Co-Citation algorithm, the influence of each Web site (or server) to the page relevance measurement is reduced to a reasonable level, and a page's relevance to the given page is determined within a local Web community (page source), rather than only within a specific Web site or server.

http://www.springer.com/978-3-540-27737-8

Web Communities
Analysis and Construction
Zhang, Y.; Xu Yu, J.; Hou, J.
2006, XI, 187 p., Hardcover
ISBN: 978-3-540-27737-8