a part of the query formulation and are combined together by the logical operator OR. In further analysis, these descriptors are not considered, and they are removed from the term-document matrix and from the relevant neighborhood of the marked set. This is done because if we use such a descriptor as an element of a subrequest containing more than one descriptor, the result of the search will be a subset of the documents that were found by using a subrequest consisting of only this descriptor.

It is clear that not all query formulations will contain subrequests consisting of one descriptor. In some IR systems, query formulations containing subrequests consisting of one descriptor will be encountered often, in others they will be encountered very rarely. It depends on the lengths of document profiles in the system. For example, if in one IR system the average length of a document profile is 10 descriptors, and in the other the average length is 80, then it is clear that in the first case query formulations will often contain subrequests consisting of one descriptor, and in the second IR system it will be a rare occurrence. Moreover, the second system might contain subrequests, for example, consisting of more than 15 descriptors. Our algorithm should be able to construct subrequests consisting of any number of descriptors.

It would seem that to construct subrequests consisting of more than one descriptor we can again use the criterion Ψ_i to compute the importance, from the point of view of the search, of any combination of descriptors from the relevant neighborhood. In other words, we can compute the ratio of the occurrence frequency of any combination of descriptors in the relevant neighborhood of the marked set to the relative occurrence frequency of this combination in the document profiles of the entire collection of documents in the system. But it is easy to see that this approach requires an incredibly large number of computations and therefore is not acceptable in practice. If we could find a way to compute the relative occurrence frequency of any set of descriptors from the relative occurrence frequency of each descriptor in the set, then the number of required computations would be much smaller and practically feasible. Next we describe our idea of how to decide when a subrequest should be formed by more than one descriptor.

We use the assumption that if several descriptors have values of Ψ_i, which are smaller than some bound L but still are very close to L, then the search based on the subrequest consisting of two such descriptors (which are "almost very important") will give us good results.

In other words, the descriptors whose Ψ_i values are greater than some lower bound L_1 but not greater than L; that is,

$$L_1 < \Psi_i \leq L$$

can be used in creating subrequests consisting of two descriptors. Analogous considerations lead to the construction of subrequests consisting of three descriptors. These subrequests include descriptors whose Ψ_i values are close enough to

L_1 and are greater than some bound L_2; that is,

$$L_2 < \Psi_i \leq L_1.$$

The same is true for subrequests consisting of four descriptors:

$$L_3 < \Psi_i \leq L_2.$$

Hence, after removing from the matrix all the descriptors whose Ψ_i values are greater than L, the algorithm transforms the matrix by placing the descriptors from right to left in decreasing order of their Ψ_i values. Then the matrix is divided into several intervals with predetermined upper and lower bounds for each interval.

For example, assuming a bound $L = 100$, the matrix in Figure 7.5 after the removal of descriptor D (value = 125), descriptor I (value = 108), and descriptor J (value = 102) is transformed into the matrix shown in Figure 7.6.

In Figure 7.6 the numbers in parenthesis represent the intervals for the Ψ_i values of the descriptors. Beneath these numbers are the descriptor names, together with their corresponding values. We can see that the first interval

$$(L_1, L]$$

includes descriptors L and C with $\Psi_i = 87$ and $\Psi_C = 94$. The second interval (L_2, L_1) contains descriptors M, B, A, and G. The third interval does not contain any descriptors because none of the descriptors from the relevant neighborhood

			L_5=30			L_3=50						L=100
		L_6=20			L_4=40		L_2=60				L_1=80	
	(7)	(6)	(5)		(4)	(3)	(2)				(1)	
		K	H	E	F		M	B	A	G	L	C
		Ψ=24	Ψ=35	Ψ=38	Ψ=48		Ψ=66	Ψ=68	Ψ=72	Ψ=79	Ψ=87	Ψ=94
1		x					x		x	x		x
2		x		x	x			x		x	x	x
3		x			x		x	x	x			x
4		x	x						x	x	x	x
5		x		x	x		x	x		x	x	x
6		x			x		x	x		x	x	x

Figure 7.6
Transformed matrix.