for automatic feedback was developed on the basis of the M-algorithm. As documented by a series of papers published from 1971 through 1974, this algorithm was substantially improved. The available experimental evidence indicates that substantial increases in effectiveness can be produced by the feedback procedures in Boolean systems.

It should be pointed out that researchers, in discussing the automatization of feedback, as a rule only refer to the automatization of the construction of query formulations. In some sense this is reasonable because, as was discussed earlier, from the earliest development of IR systems the developers knew that they could improve the expediency of control by correcting query formulations, and it is impossible to automatically correct query formulations without a method for automatically constructing query formulations. However, automatization of the construction of query formulations is not the same as automatization of all processes of feedback. Therefore, in this chapter we will not consider the methods of the automatic construction of query formulations (although the authors of these methods use them to automatize feedback in IR systems); however, these methods were described in Chapter 7 as they applied to the automatic indexing of search requests.

We start with a brief description of the method published in 1974 (Voiskunskii & Frants, 1974). This method is of interest not only because it was the first method of automatic feedback for IR systems with Boolean search but also because it was the first method to be used in the realization of automatic feedback for dynamic collections. The problem of automatization was solved by creating two separate algorithms: one algorithm for correcting query formulations in order to increase the precision level of the search and another algorithm for correcting query formulations in order to increase the recall level of the search. We describe both algorithms as follows.

The first algorithm is based on the following assumptions:

1. The decrease in the number of nonpertinent documents in the output formed by the corrected query formulation has to exceed (by a predefined factor) the decrease in the number of pertinent documents in the output formed by the corrected query formulation.

Within the framework of this assumption the following statement is true.

1.1. The predefined factor depends on both the number of pertinent documents in the output formed by the original query formulation (before correction) and the decrease in the number of pertinent documents.

We can formalize 1 and 1.1 as follows. Let $N_1$ and $R_1$ be, respectively, the number of nonpertinent and pertinent documents in the output formed by the original query formulation (before correction), and let $N_2$ and $R_2$ be, respectively, the same numbers for the corrected query formulation. Then the assumptions state that



$$(N_1 - N_2)/(R_1 - R_2) > f(R_1, R_2). \quad (1)$$

The predefined factor, denoted by $f(R_1, R_2)$ depends on the number of pertinent documents in both outputs (before and after the correction of the query formulation). This factor will be computed for each individual user, and the computation could be based on both the experimental results and the analytical investigations of this function.

The construction of the algorithm should take into account another consideration.

1.2. The satisfaction of inequality (1) on one collection of documents could be accidental and before making a decision about correcting a query formulation it is necessary to make sure that the satisfaction of inequality (1) was not accidental.

Now we describe an algorithm that is based on assumptions 1, 1.1, and 1.2.

After the system receives the user's evaluation of the output, the number of pertinent ($R_1$) and nonpertinent ($N_1$) documents is computed. Then every subrequest in the original query formulation is analyzed to see if it should be eliminated in the corrected query formulation to increase the quality of the search. This is done by performing the search among the documents in the output using the original query formulation, which has been modified by removal of the subrequest under consideration. In the new output, the numbers of pertinent ($R_2$) and nonpertinent ($N_2$) documents are computed (this could be done automatically using the user's evaluation of the original output). The inequality (1) is checked, and the information about the satisfaction (or the lack of satisfaction) of this inequality is recorded. This process is performed on a predefined number of collections of documents, which represents one evaluation cycle. The obtained information is used to accumulate statistical data that allow the algorithm to make a decision with respect to keeping the subrequest in the final version of the corrected query formulation. The evaluation cycle ends when enough statistical data are obtained.

The algorithm checks inequality (1) for every subrequest for each collection of documents in a corresponding evaluation cycle. If the query formulation is obtained again, then for each subrequest a new evaluation cycle begins immediately. If a query formulation is used after an evaluation cycle (and, naturally, after correction), then an evaluation cycle for a subrequest begins after inequality (1) is satisfied. If, as the result of the first evaluation cycle, more than one subrequest becomes a candidate for removal, then the subrequest removed first will be the one with the highest average value of $(N_1 - N_2)/(R_1 - R_2) -$