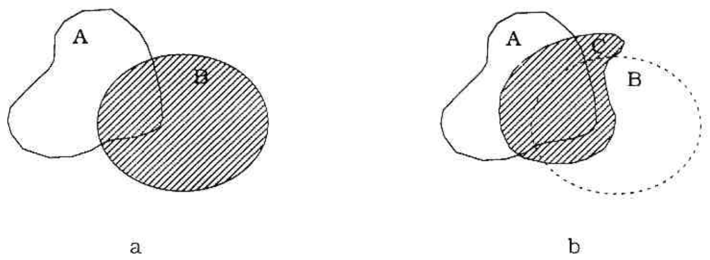

Figure 9.1
Adaptation in a static collection.

As an illustration, review Figure 9.1(a) and note that region A corresponds to the actual POIN of some user. (By "actual" we mean a POIN that has to be satisfied as the result of the search.) Because the request, as a rule, does not represent POIN exactly—see attribute 2 of POIN—region B corresponds to information about actual POIN that is received by the system. The query formulation is constructed by the system on the basis of region B rather than A, and hence the search in the system is performed to satisfy region B. The part of the POIN corresponding to A − B is not known to the system, and the information about this part of POIN (this "additional" information) is needed for adaptation. When some part of this information (contained in the documents corresponding to area A ∩ B) is received by the feedback mechanism and is used to change the system into a different state, we may have a situation like that represented in Figure 9.1(b). Area C corresponds to the new output and, as the figure shows, this output covers a bigger part of area A. Also, the level of noise corresponding to area C − A is lower than in the previous output. Hence, we see the "direction" of adaptation (in the ideal case the system will eventually produce an output that exactly corresponds to A). It is clear that at the next iteration the additional information will be represented by some part of the area (A ∩ C) − B and so on.

Note that the analysis of adaptation to region A is based on the assumption that from one iteration to another the POIN does not change at all or changes slightly. During retrospective search or in the case of one session in an on-line system, this seems plausible. Because the search is performed in the same collection of documents, then it follows that in a static collection of documents the character of adaptation is determined by attribute 2 of POIN. In other words, the system is adapting to area A, which is, as a rule, stable, and the adaptation consists of removing the inaccuracy in the POIN representation in the original query formulation.

In the case of a dynamic collection (for example, in the case of SDI) the situation is different. A search is performed in time intervals (for example, once a month) in a new collection that was accumulated during this time period. Because the thematical limits of POIN may change in time (attribute 4 of POIN), which is quite common, the system has to adapt not to fixed area A but to constantly changing area A. Also, often the user does not realize that there is a change in his or her POIN.

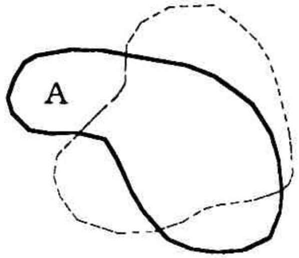

Figure 9.2
Adaptation in a dynamic collection.

Figure 9.2 illustrates this process. The area enclosed within the dotted line represents POIN at the time of the original request, and the area enclosed within the unbroken line corresponds to the current POIN, possibly after several iterations. The system has to satisfy the current POIN and, hence, the additional information to be supplied to the mechanism of feedback has to contain details about the POIN changed in time and try to adapt to this new POIN. Because thematical boundaries are constantly changing, the adaptation process will also be continuous. The IR system has to constantly monitor the POIN of every user (the user does not even have to be aware of this process) and change the query formulation to better reflect the new area A. Hence, it is clear that in the dynamic collection the character of adaptation is determined by attribute 4 of POIN. Now, after clarifying the importance of the type of the collection for the development of methods for adaptive feedback, we take a look at the approaches and ideas underlying the process of adaptation.

9.3

Some Approaches to the Realization of Adaptive Feedback

Automatic feedback for a system with Boolean search is traditionally one of the more complex processes to realize. This is also clear from the existing literature, which has very few publications containing constructive suggestions. Automatic feedback methods were introduced for Boolean systems more than 25 years ago (Voiskunskii & Frants, 1971). Almost simultaneously with the creation of the first method (algorithm) for automatic construction of query formulation, the M-algorithm developed in 1970 (see Chapter 7), the algorithm