OUTLINE OF A DYNAMIC SELF-TUNING AND ADAPTIVE INFORMATION RETRIEVAL SYSTEM

Hans-Ludwig Hausen

Institut für Software - Technologie
Gesellschaft für Mathematik und Datenverarbeitung
Schloß Birlinghoven, D-5205 St.Augustin 1, F.R. Germany

Abstract:

A self-tuning adaptive information retrieval system as an extension of the concept of a "classical" document retrieval system, is outlined. This system accepts documents and search requests in natural language, as well as the system-proposals previously produced by the system itself or prepared by the system operator. It produces a system-proposal that consists of a list of documents ranked according to their relevance to the query.
Incorporated into the system is a system valuation subsystem that uses weighted relevance judgements. This subsystem gives as output an effectiveness value and an efficiency value: both together measure the quality of an information retrieval system.
The computation of the quality values and the values themselves are independent of a specific implementation. The retrieval process in this system consists of two parts, namely a query-document match and a query-query match.

Keywords:

document retrieval, sytem valuation, document query correlation, query query correlation

Contents:

1.Motivation

Most existing document retrieval systems employ matching procedures that are based only on syntactic relations between documents and queries, as indexed by descriptors or keywords and phrases /Salt71/,/Doyl75/,/Mare74/. The frequent user of an information retrieval system should have the opportunity to influence a system with the knowledge that he has acquired about the stored information. An improvement in this direction is the extension of the retrieval process by procedures for query feedback or for document feedback, as mentioned in /Salt72/. This kind of an information retrieval system enrichment leads to more acceptable results from a retrieval process if it is possible to combine query feedback and document feedback in the search process, as outlined by /Konr71/ and as realized in the FAKYR system /Bock75/.
Nevertheless, this kind of improvement is not powerful enough to give the user an opportunity to direct the retrieval process with his own "language". To clarify what this means, let us concentrate for a moment on the problems concerned with query-indexing. A query in a system like the ones mentioned above is mapped onto the system by various text-processing techniques (e.g. trivial word elimination, synonym and homonym recognition, word stem analysis). The query-mapping in these systems is done by means of a document-term thesaurus, i.e. a list of terms which occur in the documents. This thesaurus is, roughly speaking, the language of the system but but the language of its user.
If one wants to provide more support to the user in his work with an information retrieval system, one should adapt the system to the user's needs instead of constraining the user by system dependent viewpoints. A possibility of doing this is for the system to learn through interaction with the user his view on the stored information. The user's view can then be extracted from the queries formulated in his own language, and from such proposals which are accepted by the user (in short: user-satisfying proposals). In doing this a query-term thesaurus (i.e. a list of terms used in the queries), a set of system proposals previously produced, and a document-term thesaurus are incorporated into the retrieval process.
The search process now consists of two parts, namely the "classical" retrieval process as in SMART /Salt68/ or FAKYR /Bock75/, and the new process of matching a query against the previously produced proposal and the user-satisfying proposals of the system. The result of both processes is a set of two partial proposals, each consisting of a list of documents and their correlation with the query considered. The complete proposal of the system is built up then as a parameterized composition of the two part-proposals.
In such an information retrieval system the user's point of view comes into the system via the collective proposals accepted as an answer to a certain query. The user of an information retrieval system now looks at the information stored in the system via, again roughly speaking, the document-term language, which is close to the system, and the query-term language, which is very close to the user's language if the system provides a natural-language-like query language as in FAKYR.
In the introduction or installation phase of such a dynamic self-tuning adaptive information retrieval system, that is the period before the system has produced enough satisfactory system-proposals, the system either operates like a "classical" system, or else uses prepared answers to characteristic queries in order to create a set of initial proposals. A characteristic query in the sense of such a DYSTAIR system is a query which covers one part of the documents (e.g. a cluster of documents).

2. The Model of the System

The model of the DYSTAIR system is an extension of the system model of FAKYR as described in /Bock76/. Most of the components of the DYSTAIR system are well known in document retrieval systems /Mare74/, /Doyl75/. The storage of the processed queries together with proposals accepted by the user is a new idea. The improvement provided by the DYSTAIR system is the opportunity of collecting information from the search process and thereby collecting user viewpoints, which may be useful for an iteration of a search process or for processing queries similar to each other. A new component within a retrieval system is the query-document-matrix, where the documents relevant to an certain query are stored. The main difference between FAKYR or other SMART-like systems and the DYSTAIR system is in the retrieval process.

It is assumed that documents, queries and system-proposals are represented and handled as multidimensional vectors in an abstract space. The DYSTAIR system may then be described by the following components:

a: A set DT of n1 terms used for document indexing, i.e. the document-term thesaurus.

b: A set QT of n2 terms used for query indexing, i.e. the query-term thesaurus.

c: A set DO of n3 documents in natural language.

d: A set QO of n4 queries in natural language.

e: A set BT of n5 trivial words, i.e. the banal-word thesaurus.

f: A transformation FD on the set of documents DO mapping each document onto the abstract vector space D spanned by the terms of the document-term thesaurus DT.

g: A transformation FQQ on the set of queries QO mapping each query onto the abstract vector space QQ spanned by the terms of the query-term thesaurus QT.

h: A transformation FQD on the set of queries QO mapping each query onto the abstract vector space QD spanned by the terms of the document-term thesaurus DT.

i: A set P of previously produced or prepared system-proposals p, where each p ∈ P is a subset of D.

j: A search function FSD on the Cartesian product of D and QD which induces a partial ordering on the set D. FSD takes a query q and produces the partial proposals ppd ∈ PPD, depending on the correlation of the query and the documents in D. PPD is the set of proposals produced by FSD.

k: A search function FSQ on QQ, which takes q and produces the partial proposals ppq ∈ PPQ, depending on the correlation of the query and the queries previously posed or prepared storing in QQ. PPQ is the set of proposals produced by FSQ.

l: A transformation FP on the Cartesian product of the PPQ and PPD which creates the complete system-proposal PP, which is a partial ordering on DO. The model of the classical information retrieval system does not contain step g, k and l. In this system step j produces directly the complete system proposal.

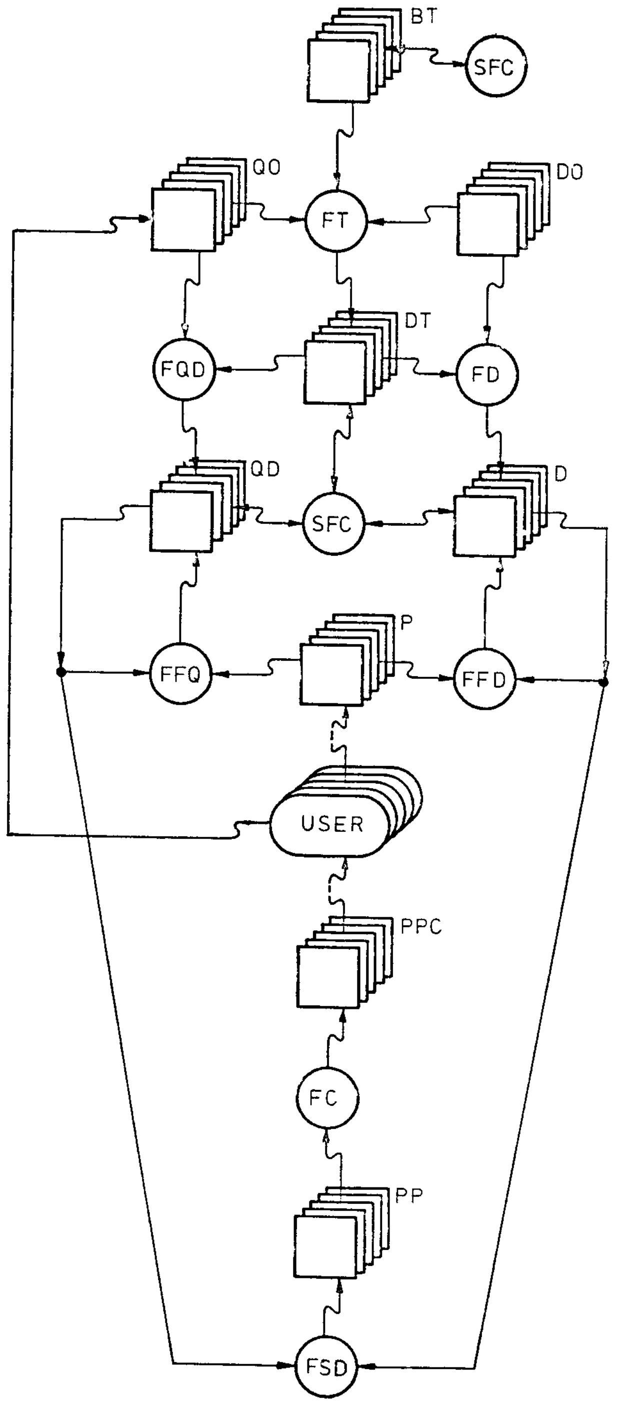
m: A transformation FC on the complete proposal PP that generates a subset PPC of PP depending on a selection parameter, i.e. the cut-off value. PPC is the proposal the system presents to the user as the answer to the query considered.

n: A set of various classification functions SFC which can be applied for clustering BT, DT, QT, QQ, QD or P.

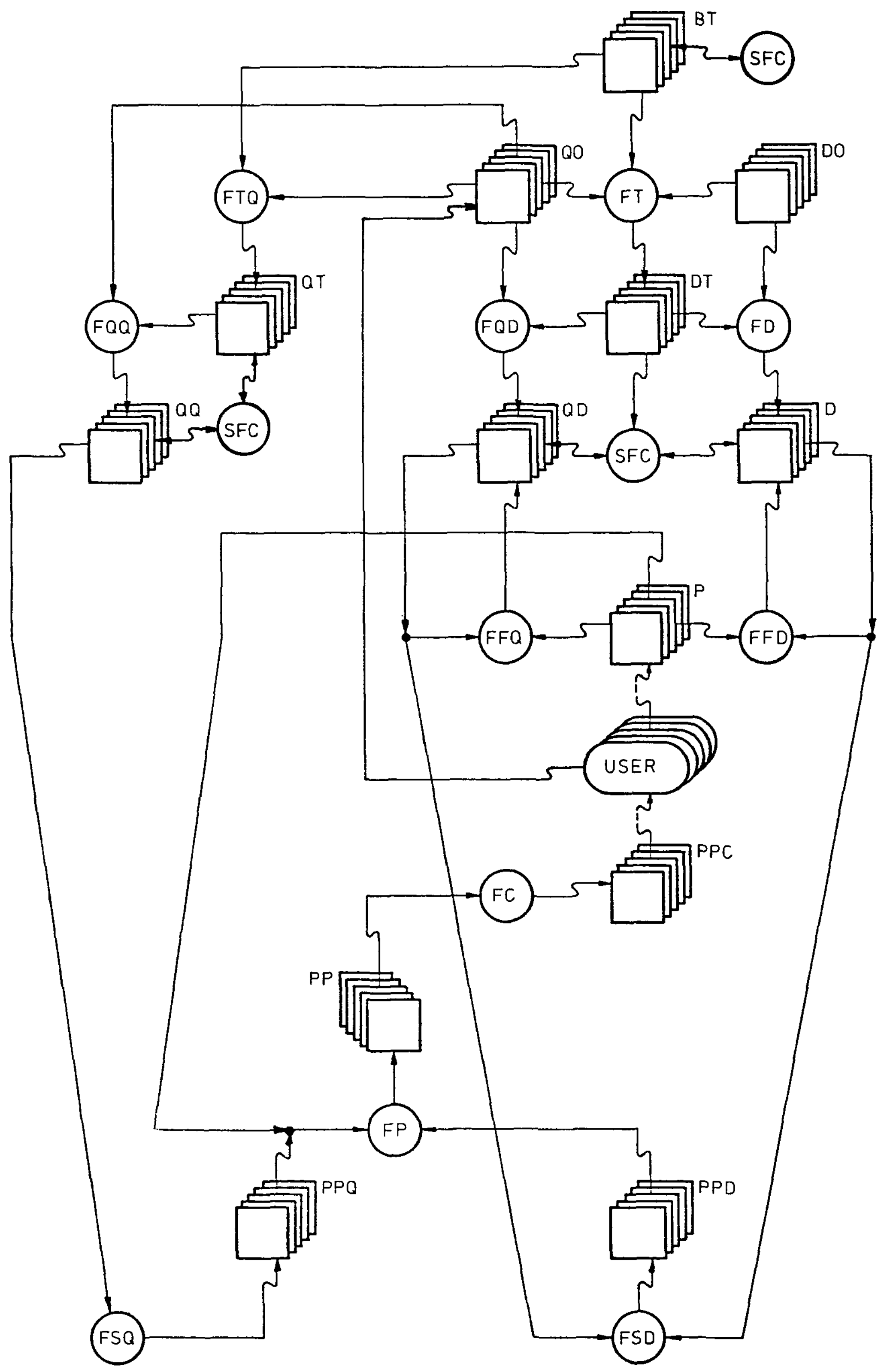
o: A transformation FFQ on the Cartesian product of P, QD and the relevance statements RS into QD, this led to query feedback.

p: A transformation FFD on the Cartesian product of P, QD, D, and the relevance statements RS into D, this led to document feedback.

Model of a classical information retrieval system



Model of the dynamic self-adaptive retrieval system DYSTAIR



Using the defined components of a dynamic self-adaptive information system the retrieval process consists of the following steps:

A: The user poses a query qo in natural language to the system.

B: The query qo is translated by the function FQD into a vector

$$qd = (qd_1, qd_2, \dots, qd_{n1}),$$

where qd_n is the weight of term n of the document-term thesaurus DT in the indexed query qd.

C: The query qo is translated by the function FQQ into a vector

$$qq = (qq_1, qq_2, \dots, qq_{n2}),$$

whereby qq_n is the weight of term n of the query-term thesaurus QT in the indexed query qq.

D: The query vector qd is matched against the document vectors

$$d = (d_1, d_2, \dots, d_{n1})$$

of the abstract vector space D. Thereby a correlation coefficient x_n is computed for each query-document pair (qd,d). This is an application of FSD and

$$cd = (x_1, x_2, \dots, x_{n3})$$

is the result of this step. A d_n in the document vector d represents the weight of term n of the document-term thesaurus DT within the document considered.

E: The query vector qq is matched against the query vectors

$$q = (q_1, q_2, \dots, q_{n2})$$

of the abstract vector space QQ. Thereby a correlation coefficient y_n is computed for each query-query pair (qq,q),

$$cq = (y_1, y_2, \dots, y_{n4})$$

is the result of this step. A q_n in the query vector q represents the weight of term n of the query-term thesaurus QT within the query considered.

F: Based on the correlation coefficients cd, cq and a query-document correlation matrix cQ, that contains the correlation of queries and documents resulting from previous search processes, a system-proposal is computed. This proposal is represented by a list of correlation coefficients

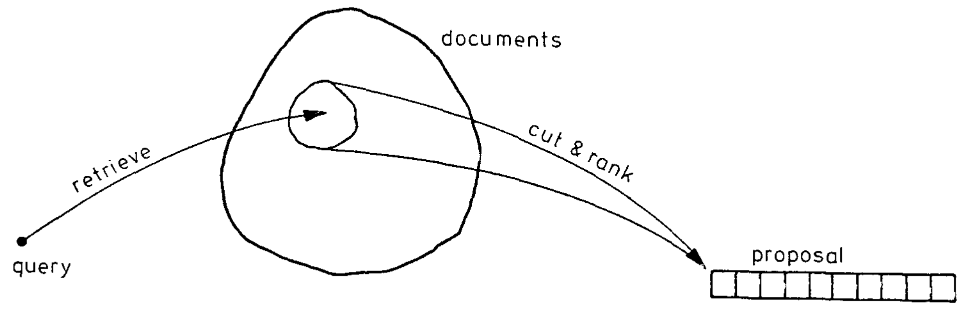
$$cv = (c_1, c_2, c_3, \dots, c_{n3}),$$

where a c_n denotes the relevance of document n to the query posed.

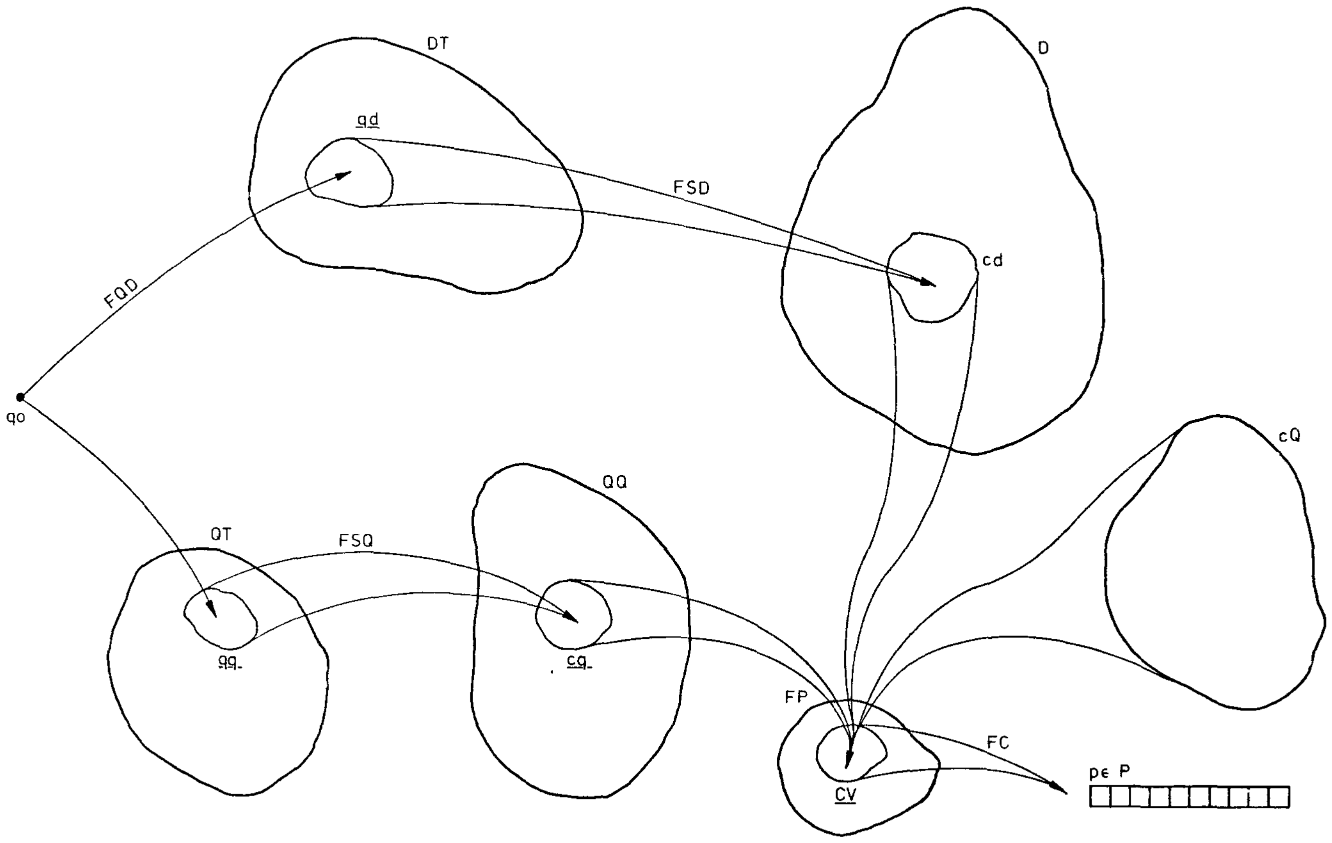
G: The documents are ranked in an decreasing order according to the correlation coefficients c_n in the vector cv of step F. The highest ranked documents are brought to the user's attention depending on the selection parameter (i.e. the cut value) chosen.

H: The user decides whether a document presented by the system is relevant or not. Next, the actual query, the relevance judgement and references to the relevant documents are put together into the proposal store P.
In this step the system dynamically performs self-adaption, in that the user's view on the stored documents is reported and will be usable in the following retrieval processes of this user. If he or she poses a query similar to the actual query afterwards to the system, the system uses the reported view to produce the answer to the new but similar query.

The classical retrieval process



The DYSTAIR retrieval process



3. The Calculation of a System-Proposal

As mentioned in Section 2, correlation coefficients of two different types are produced in a DYSTAIR retrieval process, namely query-document and 'query-query'-document correlations. The latter correlation coefficients are computed from the query-query correlation coefficients cq and the query-document correlation coefficients cQ of the proposals previously produced. The question now is how to compute a complete proposal p for a query q based on these coefficients.
For a current query q the calculation of the correlation values works as follows: In the first step, q is correlated with all the documents, resulting in a set of direct query-document correlations. In this process the query and the documents are compared according to their representations biven by the document-term thesaurus. In the next step the correlation values of q and the previously posed queries $q' \in Q$ are computed, producing a set of query-query correlations. Both sets may be written as vectors to make computations easier. From these query-query correlations and the correlations already computed between the previously posed queries and the documents, indirect correlations between q and the documents are calculated. In this step we get for each document a set of values denoting the indirect correlation between that document and the current query. The direct correlation values, computed from q and D without q', and the indirect correlation values, computed from q q' correlations, together define the set of correlation values between q and d.
We select (or compute) only one from this set of correlation values. If we do this for each document we get a set (or vector) of correlation values, where each value indicates the composite correlation between one document and the current query. The list of documents used as the system proposal is then constructed based on these correlation values.
To make this more precise and formal, the following definitions are used:

query-document correlation :

$$cd = (x_1, x_2, \ldots, x_{n3})$$

query-query correlation :

$$cq = (y_1, y_2, \ldots, y_{n4})$$

query-document correlation of proposals previously produced :

$$cQ = \begin{array}{l} z_{11}, z_{12}, \ldots, z_{1(n3)} \\ z_{21}, z_{22}, \ldots, z_{2(n3)} \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ z_{(n4)1}, z_{(n4)2}, \ldots, z_{(n4)(n3)} \end{array}$$

where:

x_n : query-document correlation of query q and document n
y_n : query-query correlation of query q and query n
$z_{(n4)(n3)}$: query-document correlation of query n4 and document n3 in the set of system-proposals P
cQ : matrix of query-document correlations of accepted proposals

correlation vector of query q :

$$cv = (c_1, c_2, \ldots, c_{n3})$$

c_n : correlation of document n and query q

$$\begin{array}{ll} cv = & (s(e\text{¤}x_1, y_1\text{¤}z_{11}, y_2\text{¤}z_{21}, \ldots, y_{n4}\text{¤}z_{(n4)1}), \\ & s(e\text{¤}x_2, y_1\text{¤}z_{12}, y_2\text{¤}z_{22}, \ldots, y_{n4}\text{¤}z_{(n4)2}), \\ & s(e\text{¤}x_3, y_1\text{¤}z_{13}, y_2\text{¤}z_{23}, \ldots, y_{n4}\text{¤}z_{(n4)3}), \\ & \ldots \\ & \ldots \\ & \ldots \\ & s(e\text{¤}x_{n3}, y_1\text{¤}z_{1(n3)}, y_2\text{¤}z_{2(n3)}, \ldots, y_{n4}\text{¤}z_{(n4)(n3)})) \end{array}$$

where:
e : experimentally selected factor
¤ : composition operator for two correlation coefficients (e.g. multiplication)
s : selection function (e.g. maximum, average)
x,y,z : as above

<u>Proposal of the system</u>

The complete proposal to a query q includes:

- System proposal sequence (documents in the order in which they are <u>retrieved)</u>

  $sps = (d_1, d_2, d_3, \ldots, d_{n3})$

  where d_n : document retrieved at step n

- System proposal (documents in the order in which they are <u>ranked)</u>

  $sp = (pd_1, pd_2, pd_3, \ldots, pd_{n3})$

  where pd_n : document ranked at position n

- System proposal correlation vector (correlation values according to the system proposal)

  $spcv = (c_{pd_1}, c_{pd_2}, \ldots, c_{pd_{n3}})$

  where c_{pd_n} : correlation of document ranked at position n

  and $c_{pd_1} < c_{pd_2} < c_{pd_3} < \ldots < c_{pd_{n3}}$

4. Effectiveness and efficiency of an information retrieval system

As extensions of standard evaluation measures (recall, precision, fallout, generality,...), see /Salt71/ and /Robe76/, two estimation procedures are introduced by the DYSTAIR system. Both procedures are based on relevance vectors and system-proposal vectors. Therefore, the relevance judgement is mapped onto a vector similar to the system-proposal vector. On the basis of these two vectors, the effectiveness and efficiency of an information retrieval system are defined in an implementation independent manner.

4.1 Effectiveness of an information retrieval system

Idea :

The difference between the ideal and the real effectivity of a system, represented by the ideal and the real effectiveness vectors of a system, can be taken to measure the effectiveness. Components of the ideal effectiveness vector are values describing the behavior of an ideal system which ranks the retrieved documents as the user does in making the relevance judgement.
The real effectiveness vector represents, in its components, values measuring the difference between the system proposal and the relevance judgement for each query. The correlation of both of these vectors is a measure of the effectiveness.

Definitions :

a. relevance judgement per query

$rj(q) = (r_1, r_2, r_3, ..., r_{n_3})$
where r_n : relevance value of document n for query q,
relevance scale example : $0 \le r_n \le 1$,
$r_n = 0$: document n is irrelevant to query q,
$0 < r_n < 1$: document n is relevant to query q but not satisfactory to the user,
$r_n = 1$: document n is relevant to query q and satisfactory to the user

b. correlation vector per query

$cv(q) = (c_1, c_2, c_3, ..., c_{n_3})$
where c_n : correlation value of document n and query q

c. system effectiveness vector

$sesv = (ses_1, ses_2, ses_3, ..., ses_{n_4})$

$ses_n = c(rj(n), sp(n))$
where ses_n : correlation of correlation vector and relevance judgement for query n,
c : correlation function on two vectors

d. ideal system effectiveness vector

$isesv = (ises_1, ises_2, ises_3, ..., ises_{n_4})$

$ises_n = c(rj(n), rj(n))$
where $ises_n$: correlation of the relevance judgement for query n to itself ,
c : correlation function on two vectors

<u>System Effectiveness</u> : $ses = c(sesv, isesv)$

4.2 Efficiency of an inforamtion retrieval system

Idea :

The most efficient information retrieval systems are those which retrieve the highest ranked (i.e. the most relevant) documents first. In line with this idea, the difference between the output of the real and the ideal retrieval processes, as represented by the difference between the real efficiency vector and an ideal efficiency vector, may be taken as a measure of the efficiency of an information retrieval system. An ideal efficiency vector represents, in its components, values describing the efficiency of an ideal system which retrieves the documents in the same order of relevance in which the user has ranked them. The real efficiency vector describes, in component n, the correlation between the relevance judgement sequence, i.e. the vector denoting the documents in the order of decreasing relevance, and the real system proposal sequence, i.e. the vector denoting the documents in the order they are retrieved.

Definitions :

a. relevance judgement sequence per query

$rjs(q) = (rd_1, rd_2, rd_3, ..., rd_{n3})$

where: relevance of document rd_n > relevance of document rd_{n+1}

b. system proposal sequence per query

$sps(q) = (d_1, d_2, d_3, ... d_{n3})$

d_n : n-th retrieved document

c. system efficiency vector

$seyv = (sey_1, sey_2, sey_3, ..., sey_{n4})$

$sey_n = c(sps(q), rjs(q))$
where sey_n : correlation of system proposal sequence and relevance judgement sequence for query n,
c : correlation function on two vectors

d. ideal system efficiency vector

$iseyv = (isey_1, isey_2, isey_3, ..., isey_{n4})$

$isey_n = c(rjs(n), rjs(n))$
where sey_n : correlation of the relevance judgement sequence for query n to itself,
c : correlation function on two vectors

<u>System Efficiency</u> : $sey = c(seyv, iseyv)$

Both the measurement of the effectiveness and the measurement of the efficiency of the system are totally independent of the implementation. The motivation for using these measures is to have only two estimation values of an information retrieval system. Our implementation-independent measures allow the comparison of the behavior of such systems on the basis of user-given relevance judgements.

5. Adaption of the system

A DYSTAIR system is dynamically self-adaptive with respect to the individual user. A user, working every day on an information retrieval system for a special set of documents, should have the opportunity to use his own "language" (i.e. his dialect) in wording the queries he wants to pose to the system. The dialect represents the user's knowledge and view of the topics considered, whereas the set of all queries posed previously to the current query reflect this knowledge, as does the set of accepted proposals.
In the DYSTAIR system, both the previously posed queries and the accepted documents are used to bring the system closer to the user. Both are stored together in the system to assist the retrieval process in finding the relevant documents. In an actual retrieval process, the current query and the previously posed queries are taken to produce a system proposal that accords with the the user's dialect. The current query and the preceding queries, all worded in terms of that dialect, are correlated, and documents relevant to queries similar to the current query are indicated by this process. This kind of an information retrieval process is based upon the wordings taken by the individual user, wheras classical retrieval processes work only on sentences worded in terms of the system thesauri, which in general do not represent knowledge or views of an individual user.
DYSTAIR captures the user's views also at a second level. After a complete retrieval process the user of the DYSTAIR system decides whether the proposed documents are relevant or not to the current query. This judgement represents also the knowledge and views of the individual user. Therefore, only documents judged as relevant are put into the proposal store, to be retrievable by subsequent search processes.
This has led to an information retrieval process, where in two different situations the knowledge and views of an individual user are taken into account, namely at the query-input and at the proposal-output stage. After an introductory phase, queries posed to the DYSTAIR system are more or less similar to each other, and the user has gained an overview of the stored information, and an understanding of actions of the system in terms of his own dialect.
The adaption process may be improved by a set of retrieval processes working on queries covering clusters of documents. This requires clustering of documents depending on user views, which is to be done by procedures working either as user support or automatically. In classical retrieval systems such as SMART, these procedures are at a highly elaborated stage. The DYSTAIR system therefore will adopt these procedures.
Based on the second level user knowledge input, the relevance judgement, document and query feedback is performed. In the case of document feedback the document space is transformed in such a way that relevant documents are moved towards the query vector in that space, while the irrelevant documents are moved in the opposite direction. After several document feedback processes the document space is transformed toward an user oriented representation of the documents.
If query feedback is done, the query is moved into the area of relevant documents. Here the wording and the rewording of a query are directed bydocuments judged as relevant or irrelevant. Hybrid feedback strategies, consisting of alternating document and query feedback, are most effective, as experiments have shown. A detailed description of both single and hybrid feedback procedures, as well as of the experiments performed on them is to be found in /Bock75/.
The query-proposal adaption and the feedback adaption are to be done either separately or jointly. The question of how they should be combined to achieve effective retrieval processes should be answered by the individual user depending on his own experiences with these procedures. It is assumed that query-proposal adaption is also useful in fuzzy retrieval systems.

6. Concluding remarks

The object of designing a DYSTAIR information retrieval system is to extend the classical information retrieval model as in SMART in such a way that the stepwise adaption of an information retrieval sytembecomes possible. To achieve this extension simple transformations and functions, as well as uncomplicated valuation functions, are added. The idea behind the DYSTAIR system model is that, in the life-time of an information retrieval system, the most recently posed queries are quite similar to the previously posed queries. One can therefore use the system-proposals related to similar queries to improve the actual retrieval process as well as to fit the resulting system-proposal.
A questioner using a DYSTAIR system more than once does not need to learn the viewpoints of the developer of such a system. On the contrary the systems gains the knowledge and the

views of the system user in each retrieval stage and throughout its entire application past. Information and documents newly stored in the system are processed also on that knowledge base and, if affected by retrieval processes, they are also subject to user-oriented adaption of the system.
The implementation of the DYSTAIR system is planed and it will be done in two steps. In step one an algebraic specification will be made which may then be transformed into programms by a compiler for abstaract algorithms and abstract data types. This will lead to a prototype of DYSTAIR. The full DYSTAIR may then be implented as an enrichment of an 'classical' information retrieval system, e.g. FAKYR.

Acknoledgement:

The author wishes to acknowledge J.L. Darlington for his critical remarks on earlier versions and J.L. Darlington, F. Gebhardt and B.S. Müller for editorial reading of the final paper.

7. References

/Bock75/ Bock, M.; Hausen, H.L.; Konrad, E.; Zuse, H.
FAKYR - an on-line Information Retrieval System
Proc. 1975 Conference on Information Sciences and Systems
Baltimore (Maryland), April 1975, pp 364-369

/Bock76/ Bock, M.; Hausen, H.L.; Zuse, H.
Konzept und Realisierung des on-line Information Retrieval Systems FAKYR
in: Hoßfeld, H. (Ed)
Proc. Praxis der Realisierung von Informationssystemen
Conference of the German Chapter of the ACM 1976
Applied Computer Science No. 2, 133-153
Hanser Verlag, München, 1976

/Doyl75/ Doyle, L.B.
Information Retrieval and Processing
Wiley, Los Angeles, 1975

/Konr71/ Konrad, E.
Dynamische Dokumenträume
Proc. GI Jahrestagung, Karlsruhe, 2.-4. Oktober 1972
Lecture Notes in Economics and Mathematical Systems, Vol.78, 499-502
Springer Verlag, Berlin-Heidelberg-New York, 1973

/Mare74/ Marek, W.; Lipski, W.; Pawlak, Z.
Information Storage and Retrieval - Mathematical Foundations
Part I & II, Computation Centre Polish Academiy of Sciences
Technical Reports 149 and 153, 1974

/Robe76/ Robertson, S.E.
A theoretical model of the retrieval characteristics of information retrieval systems
Ph.D. Thesis, University of London, 1976

/Salt71/ Salton, G. (Ed)
The SMART Retrieval System
Prentice Hall, Englewood Cliffs, N.J., 1971

/Salt72/ Salton, G.
Dynamic Document Processing
Comm.ACM 15, 1972, 658-668

'-------'